POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Aerospaziale

Tesi di Laurea Magistrale

Space Surveillance and Tracking Tool

Implementation and Test of New Methods

Relatore
Prof. Manuela Battipede

Correlatori:
Prof. Gunnar Tibert (KTH Stockholm)
Oscar Rodriguez Fernandez (Airbus Defence and Space)
Dr. Jens Utzmann (Airbus Defence and Space)

Giovanni Cirillo
matricola: s234860

Anno accademico 2018 – 2019

Contents

List of Tables

List of Figures

Abstract

In March 2019 the number of artificial objects bigger than 1 mm in orbit around the Earth is estimated to be more than 170 millions [2]. Only a small fraction of them (0.03 %) is catalogued. An impact of an operational satellite with one of these debris can damage the satellite and undermine its mission. So it is important to catalogue as many objects as possible in order to reduce the risk of a collisions. This is done by using the software tool Space Object Observations and Kalman Filtering (*SPOOK*), developed in *Airbus Defence and Space* in Friedrichshafen. The goal of this Master Thesis was to create new functionalities to this tool and improve the existing ones. In particular three main goals have been accomplished:

- a new model for the lighting ratio has been built to take into account the occultation of the Sun due to a covering body (for example the Earth or the Moon) and its influence on the solar radiation pressure, necessary to have a good model for orbit propagation;
- a tracklet building algorithm has been built to distinguish different tracklets (consecutive observations of the same object along its orbit) as a starting point for the association of different measurements belonging to the same object at distant epochs, necessary to update a catalogue of space objects;
- a model to take into account the process noise has been improved giving some suggestion on how to tune the different parameters for different kinds of orbit.

Chapter 1

Introduction

1.1 Space debris situation

Since the start of the Space Age in the late '50 the space around the Earth has been populated by human-made objects.

Not all the objects in space are operational satellites: most of them are non-operational satellites or debris caused by collisions or by loss of material.

Figure 1.1 shows the evolution in the number of catalogued objects from 1957 to 2019.

At 2019 March 9th there were about 40 000 catalogued objects, and only around 5% of them are active satellites. Another 15% are inactive satellites in orbit or decayed, while 80% are debris , as we can see in Table 1.1.

source	Payloads				Debris			All		
	On orbit	Decayed	Total	Active	On orbit	Decayed	Total	On orbit	Decayed	Total
All	5042	3383	8425	2221	14484	21157	35641	19526	24540	44066

Table 1.1. SATCAT boxscore [5].

Even if 40 000 seems to be a big number, it is only a small fraction of the total number of objects. According to the European Space Agency (*ESA*) [2] the total number of space debris objects in orbit is in the order of:

- 29000 bigger than 10 cm
- 670000 bigger than 1 cm
- More than 170 million bigger than 1 mm.

It is impossible to catalogue all these debris, because this number continuously increases and some of them are not detectable because too small, but progress in this field is necessary.

Furthermore, according to *ESA* [3] a millimetre-size object would create small surface pits, an object larger than 1 cm would cause a mission-critical damage and an impact with a 10 cm object on a spacecraft would likely cause a catastrophic disintegration of the target, due to the very high kinetic energy.

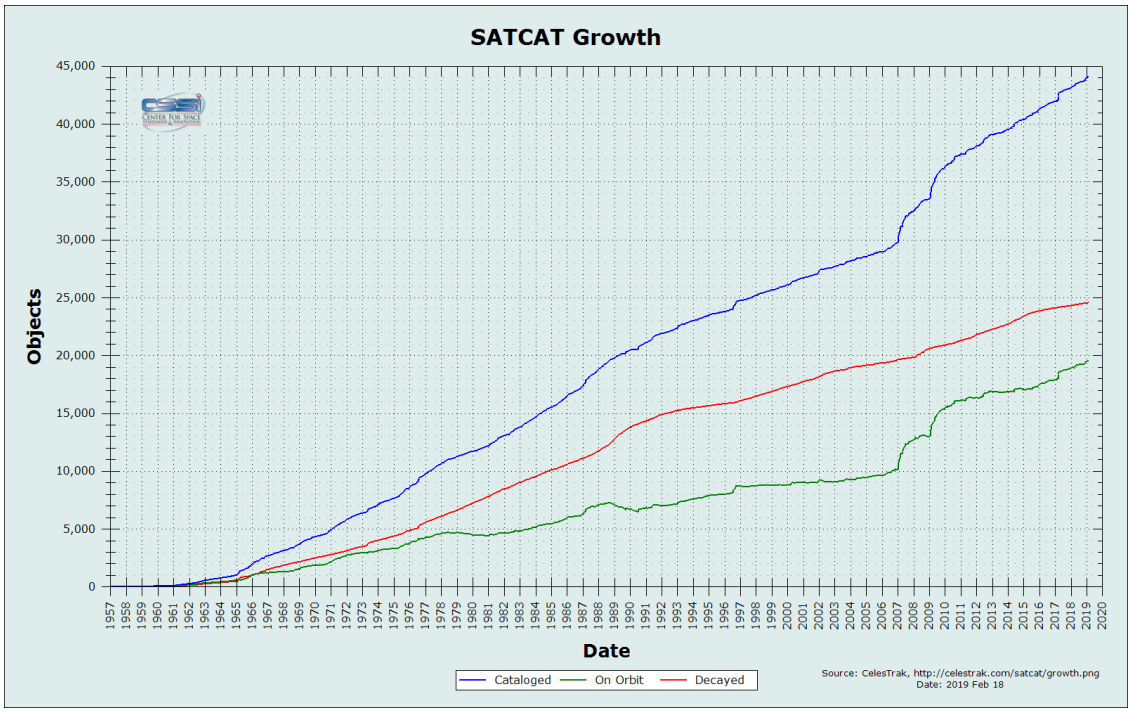

Figure 1.1. Historical SATCAT Growth, 1957 to Present, *CelesTrak* [4].

So it is very important to track and catalogue as many objects as possible, even the smallest ones.

Figure 1.2 shows the distribution of debris orbiting around the Earth. It is very clear that the distribution is not homogeneous, but most of the objects are in low orbit or in a thin layer which corresponds to geostationary orbits.

For this reason it is more important to focus the attention on these two regions, because in these regions the probability of collisions is higher.

SPOOK (SPace Object Observations and Kalman filtering) is a software tool developed by *Airbus Defence and Space* in Friedrichshafen (Germany) for the detection, cataloguing and orbit prediction analysis of Earth orbiting objects.

A space object can be detected using a telescope.

For the current project an optical telescope located in Spain is used, so most of the algorithms have been written in such a way they work properly for optical telescopes.

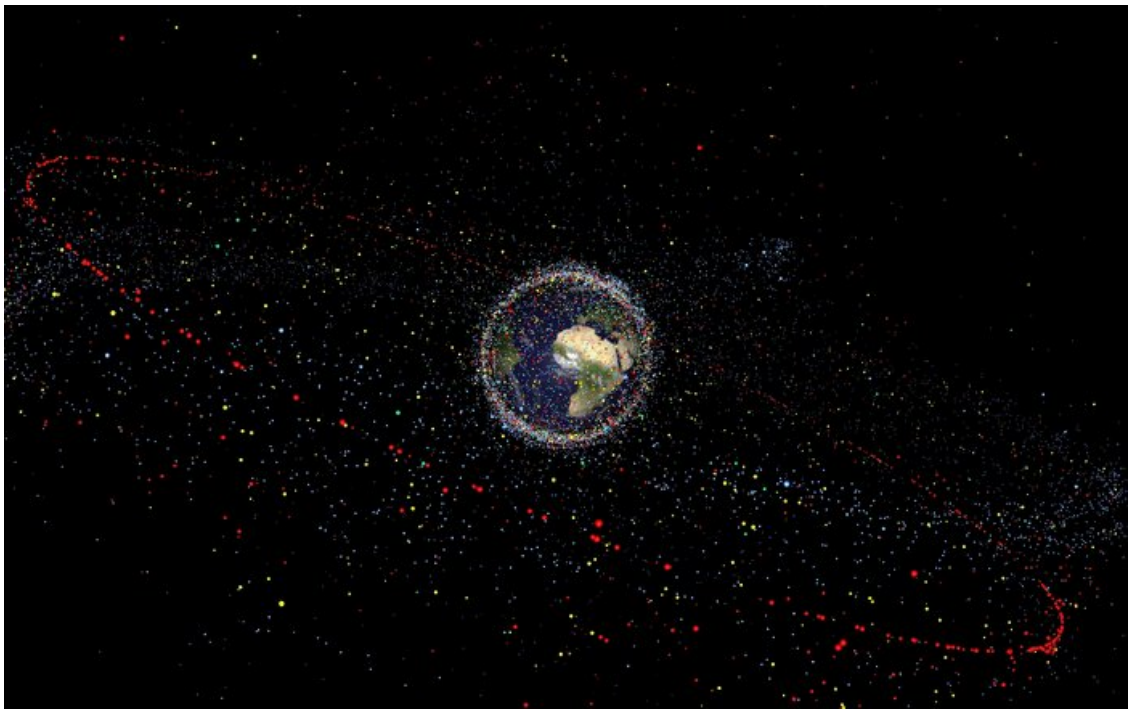

Figure 1.2. Distribution of debris orbiting around the Earth, picture by *ESA* [1].

1.2 Main goals

The software tool *SPOOK* mentioned earlier is the main software used in this Master Thesis project. This tool can be used in various modes:

- sensor calibration;
- sensor simulator;
- state and covariance propagation;
- orbit determination;
- correlation.

Since the tool is not yet in its release version, some functions inside the software need to be revisited or improved. Moreover every new function that is modified or created must be tested with a wide series of simulated data and, if possible, with real data.

Three are the main goals of this Thesis:

- The implementation of a function to compute the lighting ratio due to a total or partial eclipse. The lighting ratio affects the perturbation due to the solar radiation pressure, so it should be taken into account. A function that computes the lighting ratio is already present in the software, but some errors arise in some special configurations.

In this Thesis a new shadow function has to be implemented. This function has to be tested comparing the results obtained with simulated data with the results obtained with other tools, for example *Orekit* (a free library in *Java* providing accurate and efficient low level components for the development of flight dynamics applications).

- The implementation of an tracklet building function (to link measurements to various pieces of orbit tracks). A very raw function is already present in the software, but it takes into account only the timestep between two measurements, without any consideration on the dynamics of the problem. In this Thesis a new tracklet building algorithm has to be implemented. This algorithm has to be tested with simulated data (where we already know which measurements are linked to which object) and, if possible, also with a couple of real data.

- The implementation of the process noise in the propagation of the state vector and the covariance.

  Two models of the process noise were already implemented:

  – in the first model the process noise is modelled as a diagonal matrix which is simply added to the covariance, but this model is highly-dependent on the number of timesteps;

  – in the second model an analytical solution was implemented, but valid only in certain reference frames and if the timesteps are small (so good inside a tracklet, where the timesteps are a few seconds, but bad between a tracklet and the following, where the timestep could be several hours).

  Considering the limitations of the two previous models, new models have to be implemented to overcome these limitations. The new models have to be tested for some kinds of orbit using simulated data.

1.3 Structure of the Thesis

The structure of this Thesis can be summarised in this way:

- chapter 1 will give a brief description of the space debris problem, the software tool that has been used and the goals of this Master Thesis.

- chapter 2 will give some basic elements of theory for each topic related to the goals that we want to achieve. For some complex mathematical formulations, only the steps necessary for the comprehension of the topic will be shown. However, references to books are put for those who want to look further at these topics.

- chapter 3 will show the methods that have been used and the algorithms that have been implemented to achieve the different goals.

- chapter 4 will show the results obtained for each task, with the help of a relevant number of figures.

- chapter 5 will show what goals of the expected ones have been achieved. Moreover it will show what are the limitations of the models and algorithms that have been implemented and suggest possible improvements for future works.

Chapter 2

Background

2.1 Shadow function

2.1.1 General perturbation theory

To propagate the orbit it is necessary to know all the forces that act on the satellite. Newton demonstrated in his masterpiece *PhilosophiæNaturalis Principia Mathematica (*1687*)* that the motion of two point masses around the centre of mass of the system is a conic if we take into account only the gravitational force. For the typical values of energy, the orbit of an object around the Earth should be elliptic (the elliptic solution actually was found by Kepler at the beginning of the 17^{th} century, but only Newton gave a complete physical and mathematical formulation).

The equation that gives the elliptic solution is the following:

$$\ddot{\vec{r}}(t) = -\frac{\mu}{|\vec{r}(t)|^3}\vec{r}(t) \tag{2.1}$$

where $\vec{r}$ is the position vector, μ is the gravitational parameter (the gravitational constant G multiplied by the mass of the Earth M)

The real orbit is not perfectly elliptic because of various perturbations. Considering the perturbations, it is impossible to find an analytical solution and we can determine the orbit only numerically.

The complete equation, considering the perturbations, is the following:

$$\ddot{\vec{r}}(t) = -\frac{\mu}{|\vec{r}(t)|^3}\vec{r}(t) + \vec{f}(t, \vec{r}(t), \dot{\vec{r}}(t)) \tag{2.2}$$

The perturbations $\vec{f}$ (that are forces per unit mass) could be very complicated, so as said earlier the equation can be solved only approximately.

To solve the equation numerically, however, we need to evaluate these perturbations. Here a summary of the main perturbations:

Non-spherical gravity($\vec{f}_{\text{nsg}}$):

The Earth is not a perfect sphere and its mass is not evenly distributed within the Earth. So the point-mass gravity model is inaccurate for orbits around the Earth,

especially for low orbits. We can take into account the variations in gravitational potential, using a spherical harmonics model [12]. The sum of these harmonics gives the total perturbation:

$$\vec{f}_{\text{nsg}} = \sum_{n=2}^{\infty} \sum_{m=0}^{\infty} \vec{f}_{\text{n,m}} \tag{2.3}$$

where $\vec{f}_{\text{n,m}}$represents the contribution of the spherical harmonic of degree n and order m. The computation of each spherical harmonic requires not only the position vector, but also coefficients obtained experimentally (the formula to compute each harmonic can be found in [12]). Different models truncate the sum of harmonics up to a certain degree and a certain order. For example, one of the most used models (*EGM2008*) computes all the harmonics up to degree and order 259 [10].

Third-body perturbations ($\vec{f}_{\text{3body}}$):
Gravitational forces from third bodies can cause perturbations to an orbit. These forces are attractive central forces like the primary gravitational force, but their presence makes impossible to find an analytic solution to our differential equation. Since the gravitational force is directly proportional to the mass of the body and inversely proportional to the squared distance, it is quite obvious that the main perturbations are due to the Sun (because it is very massive) and the Moon (because it is very close). More complete models take into account also the perturbations due to the planets of the solar system.

Solar radiation pressure ($\vec{f}_{\text{srp}}$):
Solar radiation pressure is due to the fact that the light, made up of photons, exerts a certain pressure on a surface. This pressure, multiplied by the perpendicular area hit by the light, gives a perturbing force on the Keplerian orbit. This effect will be developed more deeply in subsection 2.1.2.

Propulsion ($\vec{f}_{\text{prop}}$):
For completeness also this effect is shown, even if for debris we don't have any propulsion system, while if we want to monitor also operational satellites this effect could be important. There are many different types of spacecraft propulsion. Rocket engines are one of the most widely used. The force of a rocket engine can be expressed in this way [15]:

$$\vec{F}_{\text{prop}} = -\dot{m}\vec{v}_{\text{e}} = -(\dot{m}\vec{v}_{\text{e,act}} + A_{\text{e}}(p_{\text{e}} - p_{\text{amb}})) \tag{2.4}$$

where $\dot{m}$ is the exhaust gas mass flow, $\vec{v}_{\text{e}}$ is the effective exhaust velocity, $\vec{v}_{\text{e,act}}$ is the actual jet velocity at nozzle exit plane, A_{e} is the flow area at nozzle exit plane, p_{e} is the static pressure at nozzle exit plane and p_{amb} is the atmospheric pressure. Dividing by the mass of the object we obtain the force per unit mass, so the perturbing acceleration.

Drag ($\vec{f}_{\text{drag}}$):
The effect of the atmospheric drag can be relevant for low orbits, where the air density

is high enough to slow down the object in a significant way. The equation, attributed to Lord Rayleigh, is:

$$\vec{F}_{\text{drag}} = -\frac{1}{2}\rho C_{\text{d}} A |\vec{V}|^2 \hat{u}_{\text{v}} \tag{2.5}$$

where $\vec{F}_{\text{drag}}$ is the force of drag, ρ is the density of the fluid (which above all depends on the height), C_{d} is the dimensionless drag coefficient (usually in the order of 1), A is the reference area, $\vec{V}$ is the velocity of the object relative to the fluid and $\hat{u}_{\text{v}}$ is the unit vector in the same direction as the velocity. For high orbits around the Earth the velocities could be in the order of some kilometers per second, but the density is extremely low and this makes this perturbation negligible. Also in this case we divide this force by the mass of the object to obtain the perturbing acceleration.

Other perturbations ($\vec{f}_{\text{other}}$):
Other small perturbations could be taken into account, like the magnetic perturbations due to the Earth magnetic field or the electrostatic perturbations due to charged particles in the ionosphere.

So the total perturbation can be summarized in this way:

$$\vec{f} = \vec{f}_{\text{nsg}} + \vec{f}_{\text{3body}} + \vec{f}_{\text{srp}} + \vec{f}_{\text{prop}} + \vec{f}_{\text{drag}} + \vec{f}_{\text{other}} \tag{2.6}$$

Putting all these contributions into Equation 2.2 we can solve the differential equation numerically.

2.1.2 Solar radiation perturbations

As said in subsection 2.1.1, one of the main perturbations on a Keplerian orbit is the solar radiation pressure. This pressure is extremely small, but for an object in orbit around the Earth, exposed to sunlight for months or years, this perturbation should be taken into account.

First of all we should define the solar constant G_{SC}. The solar constant measures the mean solar irradiance per unit area perpendicular to the solar rays at a distance of $1\,\text{AU} \approx 1.496 \cdot 10^8\,\text{km}$. Its value is dependent on the solar activity (so it is improperly defined as constant), and its mean value is approximately $1366\,\text{kW/m}^2$ [8].

Dividing the solar constant G_{SC} by the speed of light $c \approx 3 \cdot 10^8\,\text{m/s}$ we obtain the solar radiation pressure at the Earth's distance from the Sun (1 AU) for an absorbing surface perpendicular to the solar rays:

$$p_{\text{SR}} = \frac{G_{\text{SC}}}{c} \approx 4.5\,\mu\text{Pa} \tag{2.7}$$

For a sheet at an angle α to the solar rays, we have two contributions that depend on $\cos\alpha$: one is the reduction of the effective area A due to the fact that the surface is not perpendicular to the solar rays; the other is due to the fact that the pressure is produced only by the component of the force normal to the surface.

Moreover we are considering a perfectly absorbing surface. For a completely reflecting surface, on the contrary, the radiation the pressure that acts on the surface is doubled due to the reflected wave.

We should take into account also the distance from the Sun. To understand the relationship between the solar radiation pressure and the distance from the Sun, imagine two spheres centred on the Sun, one with radius 1 AU and the other with radius 2 AU. Ignoring the small fraction of photons that are absorbed or reflected between the two spherical surfaces, it is clear that the same quantity of photons goes through the two surfaces. But, since the second sphere has a radius double than the first, its surface is 4 times, so the mean density of photons on the second surface is reduced by a factor 4. The solar radiation pressure is directly proportional to the density of photons, so it is also reduced by a factor 4. If we consider a sphere with radius 3 AU the solar radiation pressure is reduced by a factor 9, and so on. So we can conclude that there is an inverse square law between the solar pressure and the distance.

Finally we obtain this formula for the solar radiation pressure:

$$p_{\mathrm{SR}} = C_{\mathrm{R}} \frac{G_{\mathrm{SC}}}{cR^2} \cos^2 \alpha \tag{2.8}$$

where R is expressed in Astronomical Units, C_{R} is a factor between 1 and 2 (1 means completely absorbing surface, 2 means completely reflecting surface), and the other quantities have been previously defined.

Since the pressure always acts normally to a surface, multiplying the pressure by the area of the sheet we obtain a force. Then, dividing this force by the mass of the sheet, we obtain the perturbation expressed as an acceleration.

Notice, however, that in order to account for the net effect of solar radiation on object, one would need to consider the total force (in the direction away from the sun) and not only the component normal to the surface, so one $\cos\alpha$ factor should be removed when we compute the perturbation. Moreover, calling $\hat{u}$ the unit vector in the direction from the space object to the Sun, we can write:

$$\vec{a}_{\mathrm{SRP}} = -\frac{C_{\mathrm{R}} A}{m} \frac{G_{\mathrm{SC}}}{cR^2} \cos\alpha \hat{u} \tag{2.9}$$

To extend this formula to a body with any shape, we have to introduce the cross-sectional area $A_\perp$, which corresponds to the projection of all the surfaces of the space object into a plane perpendicular to the solar rays (for the sheet case it is simply $A_\perp = A\cos\alpha$.

So the final formula valid for a space object with any shape is:

$$\vec{a}_{\mathrm{SRP}} = -\frac{C_{\mathrm{R}} A_\perp}{m} \frac{G_{\mathrm{SC}}}{cR^2} \hat{u} \tag{2.10}$$

in which $A_\perp$ may depend on the attitude of the object.

2.1.3 Shadow models

So far we have considered the surface as directly illuminated by the Sun, but what happens if another body (for example a planet or the Moon) passes in front of the object (satellite

or debris) that we are considering? In other words, how the presence of another body affects the solar radiation pressure?

According to the relative position of the Sun, the occulting body and our object, we can have one of these three cases:

1. the object is fully illuminated by the Sun;
2. the object is not illuminated by the Sun at all (shadow);
3. the object is illuminated only partially by the Sun (semi-shadow);

What we want to obtain at the end is a number that represents the fraction of the maximum solar radiation pressure that we can have for that object at that distance from the Sun with that attitude (varying the attitude of the object the surface normal to the solar rays can change, so it is important to consider the same attitude). This fraction is called *eclipse factor* or *lighting ratio*.

If the object is fully illuminated, it receives the maximum quantity of solar radiation, so the eclipse factor is 1. If the object is in shadow it doesn't receive light from the Sun, so the eclipse factor is 0. For the semi-shadow region the occulting body covers only partially the solar disc, so, depending on what percentage of the solar disc is covered, we have a value for the eclipse factor strictly between 0 and 1.

As we can see in Figure 2.1, we see how the shadow (umbra) and semi-shadow (penumbra) are originated. We draw the tangents between the Sun and the Earth and we can identify three regions: the white one is the illuminated region, the dark grey one is the umbra region and the light grey one is the penumbra region. First some considerations about the figure:

- The figure is not to scale. If we consider the dimension of the Earth as reference, the Sun should be much bigger and much farther, but in the way the figure is drawn the tangents and the angles are much clearer.
- The figure is in 2D, but if we use the approximation of spherical Sun and spherical Earth the figure is exactly the same cutting the Sun and the Earth with any plane passing through their centres.
- Even if the penumbra region is represented with the same shade of grey, it is not uniform, but it contains all the situations between the fully illuminated to the fully obscured.
- Increasing the distance from the Earth the umbra region is reduced more and more and eventually disappear, while the penumbra region gets larger and larger.

Now we want to determine the two angles α_{umbra} and α_{penumbra} and the position of the point P_1, where the shadow region ends, and P_2, where the semi-shadow region begins.

First let's define some distances:

- $\overline{SA} \equiv \overline{SC} \approx 6.95 \cdot 10^5\,\text{km}$ (radius of the Sun)
- $\overline{EB} \equiv \overline{ED} \approx 6.371 \cdot 10^3\,\text{km}$ (mean radius of the Earth)

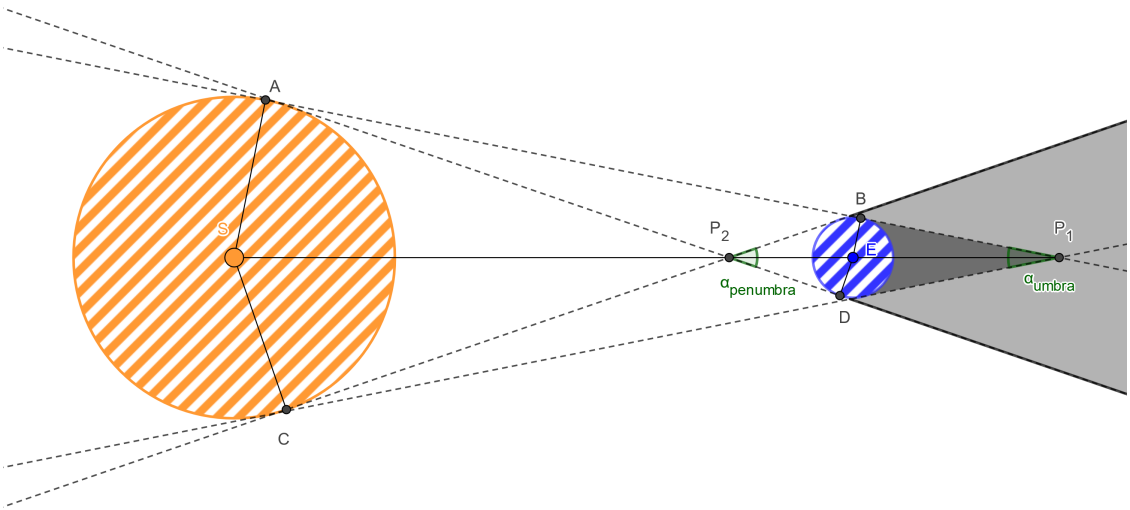

Figure 2.1. Umbra and penumbra, 2D model, not in scale.

- $\overline{SE} \approx 1.5 \cdot 10^8$ km (mean distance between the Sun and the Earth)
- $\overline{EP_1}$ (distance of the shadow cone vertex from the center of the Earth)
- $\overline{EP_2}$ (distance of the semi-shadow cone vertex from the center of the Earth)

The two triangles $\overset{\triangle}{SAP_1}$ and $\overset{\triangle}{EBP_1}$ are similar because they are both right-angled and they have an angle in common, so we can write:

$$\frac{\overline{SA}}{\overline{EB}} = \frac{\overline{SE} + \overline{EP_1}}{\overline{EP_1}} \Rightarrow \overline{EP_1} = \frac{\overline{SE}}{\frac{\overline{SA}}{\overline{EB}} - 1} \approx 1.39 \cdot 10^6 \text{ km} \tag{2.11}$$

The two triangles $\overset{\triangle}{SCP_2}$ and $\overset{\triangle}{EDP_2}$ are similar because they are both right-angled and they have an angle which is a half of two vertically opposite angles, so we can write:

$$\frac{\overline{SC}}{\overline{ED}} = \frac{\overline{SE} - \overline{EP_2}}{\overline{EP_2}} \Rightarrow \overline{EP_2} = \frac{\overline{SE}}{\frac{\overline{SC}}{\overline{ED}} + 1} \approx 1.37 \cdot 10^6 \text{ km} \approx \overline{EP_1} \tag{2.12}$$

Now we can calculate the two angles α_{umbra} and α_{penumbra}:

$$\alpha_{\text{umbra}} = 2 \arcsin \frac{\overline{EB}}{\overline{EP_1}} \approx 0.525 \text{ deg} \tag{2.13}$$

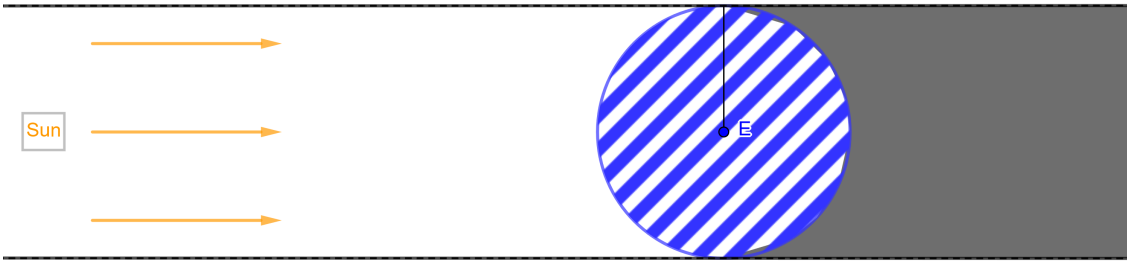

Figure 2.2. Umbra without penumbra, 2D simplified model.

$$\alpha_{\text{penumbra}} = 2 \arcsin \frac{\overline{ED}}{\overline{EP_2}} \approx 0.533 \,\text{deg} \tag{2.14}$$

We see that the two angles are quite similar and very small. Moreover the two distances $\overline{EP_1}$ and $\overline{EP_2}$ are much bigger than the normal distance of a satellite from the center of the Earth (for example a geostationary satellite is 30 times closer than the point P_1). Using this fact, since the vertex of the shadow cone is very far, one simplified model of the shadow approximates the shadow region to a cylinder. This approximation corresponds to the situation where we imagine that the Sun is at an infinite distance from the Earth, so that the tangents are parallel and there is not semi-shadow region, as we can see in Figure 2.2.

This simplified model, that at first sight looks quite good for the typical distances of a space object, has a drawback. The permanence in the penumbra region, that could last from a few seconds for a low orbit to a few minutes for a high orbit, is not taken into account at all. This effect, probably negligible for a single revolution, becomes important after thousands or tens of thousands of revolutions (typical values for a space mission), so a good model should consider this effect, especially for the orbit propagation problem (where we want to use the best model for the reference orbit).

Moreover, if we consider the Moon as occulting body, it is more probable to have an object in the semi-shadow region instead of the shadow region (as comparison, on the Earth surface it is more frequent to have a partial solar eclipse instead of a total one), so the cylindrical shadow model must be avoided.

2.2 Tracklet building

2.2.1 What is a tracklet?

The main aim of the SPOOK tool is to catalogue objects. Cataloguing an object means that we know where the object is at the current epoch, but we also know (with a certain margin of error) where that object will be at a future epoch. This is important not only because we can avoid possible collisions with that object, but also because we can point our telescope in that region of space where we expect to find the object and get new measurements that allow to have better orbit estimation of that object, reducing the errors.

To catalogue an object we need first to set an orbit. A preliminary step is the *Tracklet linking*. With tracklet linking we link together measurements belonging to the same object in a small portion of its orbit. The reason why we consider only small portions of the orbit lies in the way the measurements are obtained.

What the optical telescope does is to "take pictures" of the portion of sky in its field of view, then these pictures are elaborated to get the actual measurements that we need.

Unfortunately the telescope can't "see" an object forever: in the general case the angular velocity of the object and the ground telescope are different, so in a certain moment the object will exit the field of view of the telescope.

But even if the two velocities are the same (for example for a GEO satellite) the presence of the daylight makes impossible to have a continuous observation of that object.

Moreover, we don't want to see only one single object, but as many objects as possible.

For these reasons, what we see for one single object is a series of measurements for only a small portion of its orbit. This series of consecutive measurements is called *tracklet*.

The tracklets generated in this way have to be associated each other (determine which ones belongs to the same object) in order to set an orbit. The more tracklets we have, the better the estimation of the orbit is. This step, not treated in this Thesis, is the *Tracklet correlation*.

Even though the tracklet linking is a preliminary step, it is fundamental to improve the efficiency of the correlation.

Other kinds of sensors exist, which measure other observables, such as radars. Various combinations of observables are possible for a radar sensor:

- range, azimuth and elevation;
- range, azimuth, elevation, range rate;
- azimuth, elevation, range rate;
- only range;
- only range rate;
- range and range rate;

Azimuth and elevation are angles obtained considering the direction of the returning signal, the range is a distance obtained measuring the time for the signal to go and come back, while the range rate is obtained using the Doppler effect.

In this Thesis, the tracklet building algorithm is implemented to work properly for an optical sensor, but it has been extended to handle also the various combinations of observables for different kinds of radar sensors.

2.2.2 Noise in the measurements, missing points and fake detections

After the telescope "takes the pictures", the images are processed to get measurements.

A picture is made up of pixels. For a black and white picture a quantity called "brightness" can be associated to each pixel. An algorithm determines which pixels are bright enough to be considered as objects. Obviously some objects can cover more than 1 pixel, so the algorithm should associate close pixels with similar brightness to the same object and so return only a single measurement. Moreover this algorithm should remove, comparing with a stellar map, the pixels that actually represent stars and not space objects.

In these steps various things could go wrong. We can divide these problems into two categories and many subcategories:

- Problems due to the telescope:
  - the pointing of the telescope is not infinitely precise, so the measurement has an error;
  - the pixel is not a point, but it has a finite dimension, so the measurement has an error.
- Problems due to the algorithm:
  - an object with low brightness, for example because partially shadowed by another body, could not be detected at all;
  - a star could not be recognised in the stellar map and considered as an object;
  - an object very close to a star in the picture could be removed together with the star.

Irrespective of how these problems come from, we can distinguish three main kinds of problems:

noise in the measurement: a measurement is affected by an error;

missing point: a measurement that we should have, but we do NOT have;

fake detection: a measurement that we have, but we should NOT have.

It is very important to consider these problems in the development of the tracklet building algorithm.

2.3 Process noise

2.3.1 Introduction on the Kalman filter

In the orbit propagation problem we want to propagate the state vector over the time. First we define a state vector:

Definition 1 *A state vector is a vector that contains the components of position and velocity in a certain reference frame.*

In some cases we can extend the state vector, putting also other quantities, for example the ballistic coefficient, the solar radiation pressure coefficient or the accelerations.

Given the state vector of a space object at time t_0 and a model of the forces that act on that object over the time, it is possible to propagate the state vector for times $t > t_0$. This is our reference orbit.

The real orbit will be different from the reference orbit for two main reasons:

- There is an error in the initial state.
- There is an error in the model of the forces that act on the space object.

For the moment we ignore the error in the model and we focus on the error on the initial state.

The error in the initial state is due to the fact that we don't measure directly the quantities in the state vector but we measure some observables (for example for a optical telescope right ascension and declination) and then, using the relationship between the observables and the state vector we determine position and velocity of the space object.

However, the initial measurement is affected by an error, so this error is projected also on the initial state. So, together with the initial state, we have also the initial covariance, that represents the uncertainty in a statistical sense (the covariance matrix is a matrix that contains in the diagonal the variances of the elements of the state vector and off-diagonal the joint variabilities).

When we propagate, we propagate both the state and the covariance. Even if the initial covariance is quite small, if we don't have any other measurement and we propagate basing only on the first measurement, the covariance will grow more and more and after a certain time the covariance is so high that we could expect the object to be on one side or the other side of the Earth with the same probability.

Obviously we don't want that, and it is the reason why we need other measurements. When we have a new measurement, we have more information than before, so this measurement can be used to have a better estimation of the state vector.

This new better estimation can be obtained in different ways: one of the most used for aerospace applications is the Kalman filter. Using the Kalman filter for each new measurement we have a new estimation of the state and a new estimation of the covariance basing only on the previous estimation and the uncertainties in the measurements, what we call *measurement noise*.

Adding new measurements we have better and better estimations of the state, and the covariance will drop, so we are converging. But are we sure that we are converging to the true state?

Actually it depends. If our model well represents the reality the answer is yes. Unfortunately in general our model is not perfect, but it is affected by an error, what we call *process noise*, so it is possible to converge to a value different from the true state.

Someone could wonder "if we add new measurement that are related to the true state within a certain tolerance, how can we converge to a different value?"

The reason lies in the way the Kalman filter works (as we can see better in subsection 2.3.2): when the covariance becomes very low, we reach the saturation of the filter, so the filter becomes insensitive to new measurements. In other words, when the covariance is low we trust more on the previous estimation than on the new measurements, ignoring the new ones.

So it is very important to take into account not only the measurement noise, but also the process noise inside the Kalman filter.

2.3.2 Iterative process in the Kalman Filter

In this section the iterative process in the Kalman filter is explained. The process is explained in a more practical way, without going too deep in the math.

The flowchart in Figure 2.3 represents a very simple case, with a single measured value. Note that the dashed lines are used only in the first iteration of the Kalman Filter.

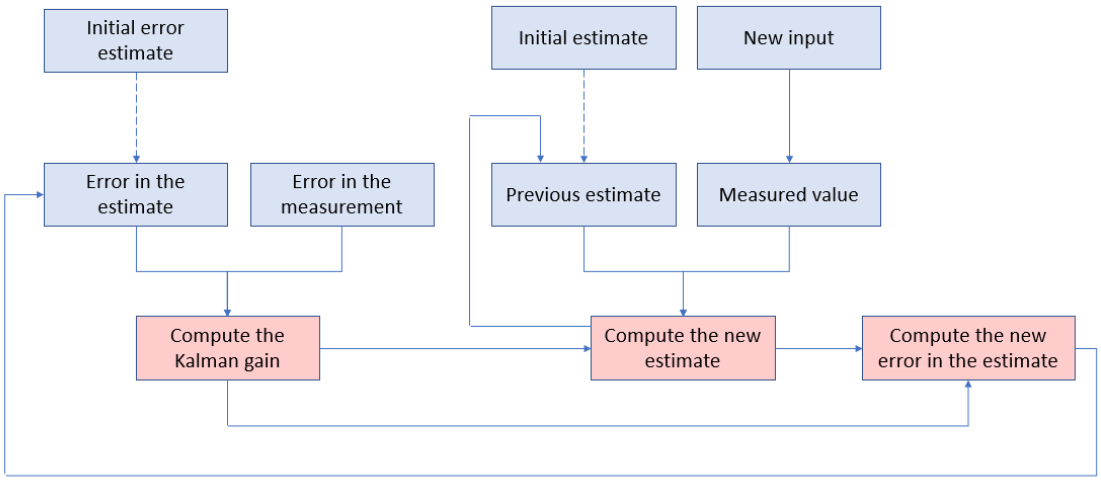

Figure 2.3. Flowchart of the iterative process in the Kalman Filter.

So we start with an initial estimate of the state and the uncertainty (the one that in our case is called covariance). Using the information of the initial error estimate and the uncertainty in the measurements, we compute the Kalman gain. The Kalman gain has an important meaning, and it says how much we trust on the previous estimate and how

much on the new measurement. It can be computed in this simple way:

$$\mathrm{KG} = \frac{\varepsilon_{\mathrm{est}}}{\varepsilon_{\mathrm{est}} + \varepsilon_{\mathrm{meas}}} \tag{2.15}$$

where KG is the Kalman gain, $\varepsilon_{\mathrm{est}}$ is the error estimate and $\varepsilon_{\mathrm{meas}}$ is the uncertainty in the measurements.

Its value can be between 0 and 1. When the Kalman gain is 1 it means that the uncertainty in the measurement is small compared to the error estimate, so we trust completely on the new measurement ignoring the previous estimation.

On the contrary, if the Kalman gain is 0 it means that the error estimate is small compared to the uncertainty in the measurement, so we trust completely on the previous estimate ignoring the new measurement (it is what in subsection 2.3.1 we called *saturation of the filter*).

The next step is to compute the new estimate of the state. For the new estimate, we need the previous estimate, the new measurement and the Kalman gain:

$$\hat{x}_t = \hat{x}_{t-1} + \mathrm{KG}(x_{meas} - \hat{x}_{t-1}) \tag{2.16}$$

where $\hat{x}_t$ is the new estimate of the state, $\hat{x}_{t-1}$ is the previous estimate of the state and x_{meas} is the new measurement. As said before, for the case $\mathrm{KG} = 0$ we obtain $\hat{x}_t = \hat{x}_{t-1}$, while for the case $\mathrm{KG} = 1$ we obtain $\hat{x}_t = x_{meas}$.

Finally we have to calculate the new error in the estimate:

$$\varepsilon_{\mathrm{est,t}} = \frac{\varepsilon_{\mathrm{meas}} \cdot \varepsilon_{\mathrm{est,t\text{-}1}}}{\varepsilon_{\mathrm{meas}} + \varepsilon_{\mathrm{est,t\text{-}1}}} = (1 - \mathrm{KG})\varepsilon_{\mathrm{est,t\text{-}1}} \tag{2.17}$$

so the error in the estimate is reduced.

This three steps are repeated whenever we have a new measurement.

This is the basic Kalman filter process. In the orbit determination problem, the input is multiple (we have a series of observables converted into a state vector) and the problem is not linear, so more complex versions of the Kalman filtered should be used, for example the extended Kalman filter or the unscented Kalman filter. The formulas are more complex, but the basic idea is the same: we compute a gain, we compute a new estimation of the state and a new error in the estimate (see [7], [13] and [16] for a more complete mathematical formulation).

2.3.3 Linearization of the orbit determination process

In the general orbit determination problem, the governing equations for the dynamics and the measurements are:

$$\begin{cases} \dot{\mathbf{X}} = \mathbf{A}(\mathbf{X}, t), & \mathbf{X}(t_k) = \mathbf{X_k} \\ \mathbf{Y_i} = \mathbf{B}(\mathbf{X_i}, t_i) + \varepsilon_{\mathbf{i}}, & i = 1, \ldots, l \end{cases} \tag{2.18}$$

where $\mathbf{X}$ is the state vector (the unknown), $\mathbf{A}(\mathbf{X}, t)$ is the matrix which represents the dynamical model, $\mathbf{Y_i}$ is the i-th measurement, $\mathbf{B}(\mathbf{X_i}, t_i)$ represents the relationship between the state and the measurement, and $\varepsilon_{\mathbf{i}}$ is the error of the i-th measurement.

If we assume that the dynamical model is perfect and that we know the exact value of $\varepsilon_{\mathbf{i}}$ for each measurement, what we obtain is the true state. Unfortunately the model is not perfect and the error is known only in a statistical sense, so what we obtain is not the true state, but an estimation of the true state. Since the equations are not linear, we can't apply the normal Kalman filter.

So we have to linearise the problem around a certain reference trajectory, applying a normal Kalman filter not on the state vector, but on the difference between the state vector and the reference trajectory [13].

If the reference trajectory is reasonably close to the true trajectory, we can expand in Taylor's series:

$$\begin{cases} \dot{\mathbf{X}}(t) \approx \mathbf{A}(\mathbf{X}, t) = \mathbf{A}(\mathbf{X}^*, t) + \left(\dfrac{\partial \mathbf{A}(t)}{\partial \mathbf{X}(t)}\right)^* (\mathbf{X}(t) - \mathbf{X}^*(t)) \\ \mathbf{Y_i} \approx \mathbf{B}(\mathbf{X_i}, t_i) + \varepsilon_{\mathbf{i}} = \mathbf{B}(\mathbf{X_i^*}, t_i) + \left(\dfrac{\partial \mathbf{B}}{\partial \mathbf{X}}\right)_i^* (\mathbf{X}(t_i) - \mathbf{X}^*(t_i)) + \varepsilon_{\mathbf{i}} \qquad i = 1, \dots, l \end{cases} \tag{2.19}$$

where $()^*$ indicates that the partial derivatives are evaluated on the reference solution and $\mathbf{X}^*(t)$ is the reference state vector as a function of time (the reference trajectory).

Adding the condition $\mathbf{X}^*(t) = \mathbf{A}(\mathbf{X}^*, t)$ and $\mathbf{Y_i^*} = \mathbf{B}(\mathbf{X_i^*}, t_i)$, and defining $\mathbf{x}(t) = \mathbf{X}(t) - \mathbf{X}^*(t)$ and $\mathbf{y_i} = \mathbf{Y_i}(t) - \mathbf{Y_i^*}(t)$ we obtain:

$$\begin{cases} \dot{\mathbf{x}}(t) = F(t)\mathbf{x}(t) \\ \mathbf{y_i} = H_i \mathbf{x_i} + \varepsilon_{\mathbf{i}}, \qquad i = 1, \dots, l \end{cases} \tag{2.20}$$

where $F(t) = \left(\dfrac{\partial \mathbf{A}(t)}{\partial \mathbf{X}(t)}\right)^*$ and $H_i = \left(\dfrac{\partial \mathbf{B}}{\partial \mathbf{X}}\right)_i^*$.

If we want to take into account the error in the model, we have to add another term in the previous equation:

$$\begin{cases} \dot{\mathbf{x}}(t) = F(t)\mathbf{x}(t) + G(t)\mathbf{q}(t) \\ \mathbf{y_i} = H_i \mathbf{x_i} + \varepsilon_{\mathbf{i}}, \qquad i = 1, \dots, l \end{cases} \tag{2.21}$$

where $\mathbf{q}(t)$ is the process noise and $G(t)$ is the matrix that maps the process noise into the state.

2.3.4 Pseudo-process noise

The easiest way to take into account the process noise is to "inflate" the covariance adding a constant diagonal matrix to the covariance to avoid the saturation of the filter.

$$P_{\text{tot}} = P + Q_{\text{pseudo}} \tag{2.22}$$

where P is the covariance due to the errors in the measurements and Q_{pseudo} is the pseudo-process noise matrix. This approach has two main drawbacks:

- We are adding some values that don't have a clear physical meaning, so the choice of the right values is not so immediate.

- the result is highly dependent on the timesteps: if for example we double the number of timesteps in the same timespan, we add twice the same constant quantity, so we obtain a result that is completely different from the previous case.

For these reasons this approach, developed in [11] and implemented in *SPOOK*, can't be applied in a general case.

So other two approaches are used:

- State Noise Compensation (SNC)
- Dynamical Model Compensation (DMC)

2.3.5 State Noise Compensation

In the State Noise Compensation the process noise q is modelled as a white noise ([13, eq. 4.9.2]):

$$\begin{cases} E[q(t)] = 0 \\ E[q(t)q(\tau)^T] = Q_{\text{proc}}(t)\delta(t-\tau) \end{cases} \tag{2.23}$$

where $E[*]$ is the expectation operator, $\delta(t-\tau)$ is the Dirac Delta (a function whose value is 0 everywhere except in the origin $t = \tau$, where it is $+\infty$) and Q_{proc} is a diagonal matrix which contains the squared standard deviations of the perturbations $q(t)$:

$$Q_{\text{proc}} = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix} \tag{2.24}$$

Considering the measurement noise and the process noise as completely uncorrelated, we can consider the two covariance matrices due to the two effects separately, and simply adding them using the superposition of effects.

Using Equation 2.21 and Equation 2.23, after many mathematical steps we obtain this differential equation to compute the process noise covariance matrix Q ([13, eq. 4.9.35]):

$$\dot{Q}(t) = F(t)Q(t) + Q(t)F^T(t) + G(t)Q_{\text{proc}}(t)G^T(t) \tag{2.25}$$

This differential equation has semi-analytical solution only for some special cases (considering only small timesteps, for example inside a single tracklet [11]), but not valid in the general case (for example between two tracklets). For this reason, a numerical integration of Equation 2.25 is suggested.

As said earlier, this matrix Q is added to the covariance due to the measurement noise to obtain the total covariance used in the Kalman Filter:

$$P_{\text{tot}} = P + Q \tag{2.26}$$

same formulation as for the pseudo-process noise, but in this case the matrix Q has a clear physical meaning and the final covariance is not dependent on the timesteps (Q is not constant, so reducing the timestep also the impact of the matrix Q on the total covariance becomes smaller).

2.3.6 Dynamical Model Compensation

The State Noise Compensation belongs to a more complete class of processes, called Gauss-Markov processes.

The general expression of the n^{th} order Gauss-Markov process, considering the perturbation only in the accelerations, is the following:

$$\frac{\mathrm{d}^n a_{\text{unmod}}}{\mathrm{d}t^n} = q(t) + \sum_{i=0}^{n-1} c_i \frac{\mathrm{d}^i a_{\text{unmod}}}{\mathrm{d}t^i} \tag{2.27}$$

where c_i are constant coefficients and $q(t)$ is a white random variable with the same properties as in Equation 2.23.

From this general formulation it is quite clear that the state noise compensation is a 0^{th} order Gauss-Markov process:

$$a_{\text{unmod}} = q(t) \tag{2.28}$$

We wonder what happens if we consider an order higher than 0. In this case the process is called in general Dynamical Model Compensation.

For example with $n = 1$ we have the 1^{st} order Gauss-Markov process:

$$\frac{\mathrm{d}a_{\text{unmod}}}{\mathrm{d}t} = q(t) - \beta a_{\text{unmod}} \tag{2.29}$$

For $n = 2$ we have the 2^{nd} order Gauss-Markov process:

$$\frac{\mathrm{d}^2 a_{\text{unmod}}}{\mathrm{d}t^2} = q(t) - \beta a_{\text{unmod}} - \alpha \frac{\mathrm{d}a_{\text{unmod}}}{\mathrm{d}t} \tag{2.30}$$

and so on.

We focus of the 1^{st} order. Equation 2.29 can be solved and we obtain this expression for one generic component of the unmodelled acceleration [13, eq. 4.9.57]:

$$\begin{cases} E[a_{\text{unmod,i}}(t)] = a_{\text{unmod,i}}(t_0)\mathrm{e}^{-\beta(t-t_0)} \\ E[(a_{\text{unmod,i}}(t) - E[a_{\text{unmod,i}}(t)])(a_{\text{unmod,i}}(\tau) - E[a_{\text{unmod,i}}(\tau)])^T] = \dfrac{\sigma_i^2}{2\beta}\left(1 - \mathrm{e}^{-2\beta(t-\tau)}\right) \\ E[(a_{\text{unmod,i}}(t) - E[a_{\text{unmod,i}}(t)])(a_{\text{unmod,j}}(\tau) - E[a_{\text{unmod,j}}(\tau)])^T] = 0, \qquad i \neq j \end{cases} \tag{2.31}$$

where the first part is the deterministic term and the other two parts are the covariance (the stochastic term).

From the stochastic part we can compute Q_{proc}:

$$Q_{\text{proc}} = \frac{1 - \mathrm{e}^{-2\beta(t-t_0)}}{2\beta} \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix} \tag{2.32}$$

So it is possible to compute the process noise covariance matrix Q even in this case substituting in Equation 2.25 the Q_{proc} just calculated.

But there is another important difference between state noise compensation and dynamical model compensation: in the state noise compensation the unmodelled accelerations are assumed as stochastic variables but with zero mean and a certain standard deviation, while in the dynamical model compensation the mean is not zero, so it can be estimated as the position and the velocity and added into the state vector.

2.3.7 Autocorrelation function

In the dynamical model compensation we have a parameter called β, which can be defined as the inverse of the correlation time. To understand the meaning of this parameter, we should introduce another concept: the autocorrelation function, which measures the correlation between a signal x and a delayed copy of itself:

$$R(t,\tau) = E[x(t)x(\tau)^T] \tag{2.33}$$

As shown in [13, eq. 4.9.57], the autocorrelation function for a 1^{st} order Gauss-Markov process is:

$$R(t,\tau) = E[a_{\text{unmod,i}}(t)a_{\text{unmod,i}}(\tau)^T] = \mathrm{e}^{-\beta(t-\tau)}\left(a^2_{\text{unmod,i}}(t_0)\mathrm{e}^{-2\beta(\tau-t_0)} + \frac{\sigma_i^2}{2\beta}(1-\mathrm{e}^{-2\beta(\tau-t_0)})\right) \tag{2.34}$$

Choosing $\tau = t_0$ we can simplify a lot the expression:

$$R(t,t_0) = \mathrm{e}^{-\beta(t-t_0)}\left(a^2_{\text{unmod,i}}(t_0) + \frac{\sigma_i^2}{2\beta}\right) \tag{2.35}$$

This autocorrelation function has a finite value for $t = t_0$ and then decays exponentially. Here we can understand the meaning of β: after a time $\Delta t = 1/\beta$ the value is reduced to $\mathrm{e}^{-1} \approx 37\%$ of its initial value.

Here we notice another difference with respect to the state noise compensation. In the state noise compensation, as said in Equation 2.23:

$$R(t,t_0) = E[a_{\text{unmod,i}}(t)a_{\text{unmod,i}}(\tau)^T]\sigma_i^2\delta(t-\tau) \tag{2.36}$$

so the autocorrelation is infinite for $t = t_0$ and 0 everywhere else.

The autocorrelation function is useful to show the difference between the two approaches:

- In the state noise compensation not only the perturbations in the different directions are uncorrelated, but also the perturbations in the same direction at different times ($R = 0$).
- In the dynamical model compensation the perturbations in the different directions are uncorrelated, but the perturbations in the same direction at different times are exponentially correlated (R decays exponentially). In other words, a perturbation "depends" on the perturbation at previous times, when the time difference is not so big.

It looks clear that the dynamical model compensation is more realistic to model the process noise, because in reality there is a correlation of the perturbations at different times.

Chapter 3

Methods

3.1 Shadow function

3.1.1 Shadow algorithm

The algorithm computes the eclipse factor. It has been written using *Fortran*. For simplicity, we will refer to a space object as *satellite* even though it could be a space debris, and we will refer to the occulting body as *Earth* even though it could be another body.

Now a list of quantities and variables that are used in the algorithm is shown, with a brief explanation of what they represent:

$\rho_{\mathbf{sun}}$**:** mean radius of the Sun ($695.5 \cdot 10^3$ km);

$\rho_{\mathbf{earth}}$**:** mean radius of the occulting body (if it is the Earth, its value is $\approx 6.371 \cdot 10^3$ km);

$\vec{r}_{\mathbf{earth,sun}}$**:** relative position of the Sun with respect to the Earth;

$\vec{r}_{\mathbf{earth,sat}}$**:** relative position of the satellite with respect to the Earth;

$\vec{r}_{\mathbf{sat,sun}}$**:** relative position of the Sun with respect to the satellite;

$\theta_{\mathbf{sun}}$**:** Sun visual cone semiangle (seen from the satellite);

$\theta_{\mathbf{earth}}$**:** Earth visual cone semiangle (seen from the satellite);

α**:** angle between the Sun and the Earth visual cone axes (note that after the projection this angle becomes a segment);

$\Omega_{\mathbf{sun}}$**:** solid angle of the Sun visual cone, projected on a plane (it is a circle);

$\Omega_{\mathbf{earth}}$**:** solid angle of the Earth visual cone, projected on a plane (it is a circle);

$\Omega_{\mathbf{int}}$**:** solid angle of the intersection of the two visual cones, projected on a plane (it is the intersection of two circles);

$\alpha_{\mathbf{sun}}$**:** angle between the Sun visual cone axis and the radical axis (note that after the projection this angle becomes a segment);

$\alpha_{\mathbf{earth}}$**:** angle between the Earth visual cone axis and the radical axis (note that after the projection this angle becomes a segment);

$\beta_{\mathbf{sun}}$**:** semiangle between the intersection points and the centre of the projection of the Sun;

$\beta_{\mathbf{earth}}$**:** semiangle between the intersection points and the centre of the projection of the Earth;

ν**:** the eclipse factor.

In Figure 3.1, Figure 3.2 and Figure 3.3 the meaning of the different quantities is shown graphically.

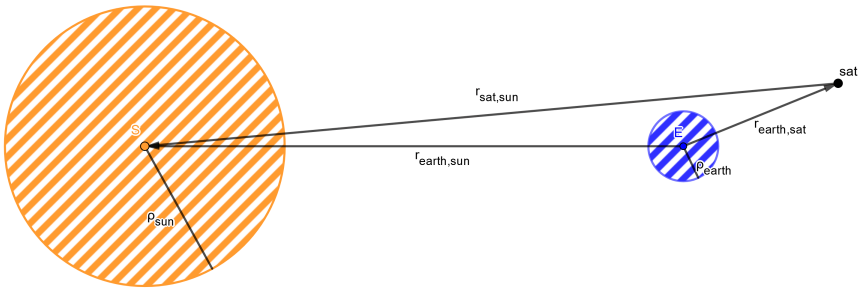

Figure 3.1. Known initial quantities for the shadow function algorithm.

The only information that we have at the beginning are ρ_{sun}, ρ_{body}, $\vec{r}_{\text{body, sun}}$ and $\vec{r}_{\text{body, sat}}$. All the other quantities can be obtained from these 4 quantities using the following algorithm, in order to compute the eclipse factor ν:

1. Compute $\vec{r}_{\text{sat,sun}}$:

$$\vec{r}_{\text{sat,sun}} = \vec{r}_{\text{sun}} - \vec{r}_{\text{sat}} \tag{3.1}$$

2. Return an error if the position of the satellite is inside the Sun or the Earth, so if:

$$|\vec{r}_{\text{sat,sun}}| \leq \rho_{\text{sun}} \quad \text{or} \quad |\vec{r}_{\text{earth,sat}}| \leq \rho_{\text{earth}} \tag{3.2}$$

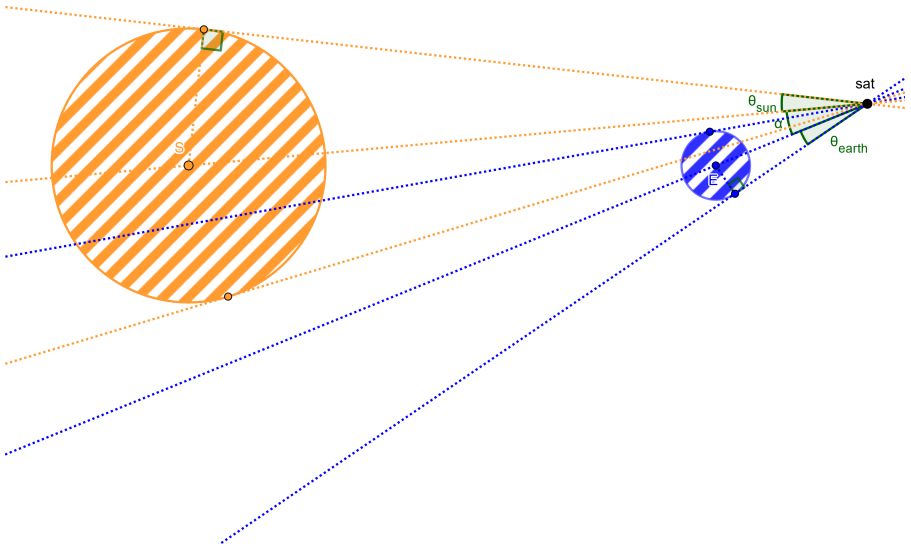

Figure 3.2. Quantities related to the visual cones.

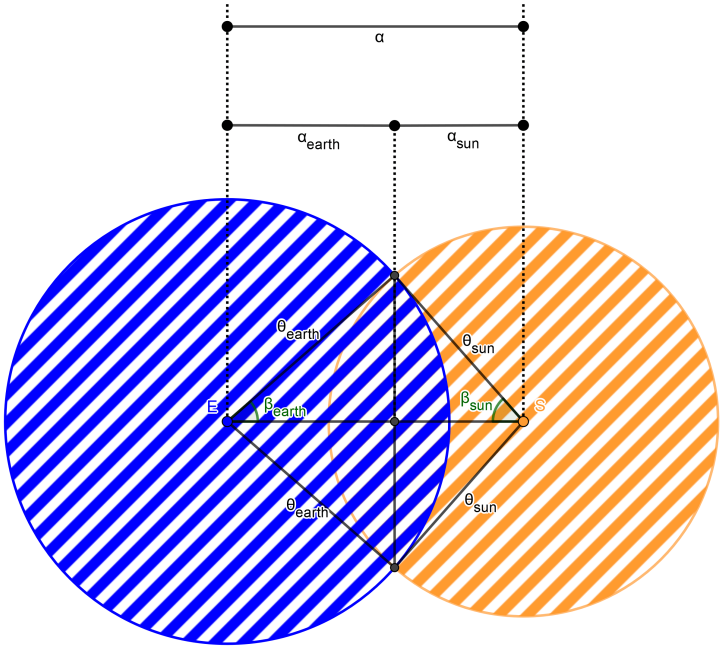

Figure 3.3. Quantities related to the projection of the visual cones into a plane.

3. Compute the angle α between the two visual cone axes (since the angle is seen from the satellite, we have to compute the angle between $\vec{r}_{\text{sat,sun}}$ and $\vec{r}_{\text{sat,earth}} = -\vec{r}_{\text{earth,sat}}$):

$$\vec{r}_{\text{sat,sun}} \cdot (-\vec{r}_{\text{earth,sat}}) = |\vec{r}_{\text{sat,sun}}||\vec{r}_{\text{sun}}| \cos\alpha \Rightarrow \alpha = \arccos \frac{\vec{r}_{\text{sat,sun}} \cdot (-\vec{r}_{\text{earth sat}})}{|\vec{r}_{\text{sat,sun}}||\vec{r}_{\text{sun}}|} \tag{3.3}$$

4. Compute the visual cone semiangles θ_{sun} and θ_{earth} using a bit of trigonometry:

$$\theta_{\text{sun}} = \arcsin \frac{\rho_{\text{sun}}}{|\vec{r}_{\text{sat,sun}}|} \tag{3.4}$$

$$\theta_{\text{earth}} = \arcsin \frac{\rho_{\text{earth}}}{|\vec{r}_{\text{earth,sat}}|} \tag{3.5}$$

5. The two visual cones define two solid angles. In common cases θ_{sun} and θ_{earth} are small, so also the corresponding solid angles are small. As a consequence, we can project the two solid angles into a plane tangent to a reference sphere, so that they become circles. When α is small we can imagine to project into the same plane even if the two planes are slightly different. Obviously α is not necessarily small, so in this case we should compute the intersection of the two visual cones without projecting them. However it is quite obvious that if α is too big there is no intersection between the visual cones (the solution is trivial in this case) so computationally only the case with small α is of interest, and the approximation valid for small angles is good. The radii of the two circles depend on the reference sphere that we consider. If we consider the unit sphere, for small angles we can approximate an arc with a segment with the same arc length, so θ_{sun} and θ_{earth} are the radii of the two circles and α is the distance between the centres of the two circles.

   According to the relation between θ_{sun}, θ_{earth} and α, 3 cases can be distinguished:

   (a) $\theta_{\text{sun}} + \theta_{\text{earth}} \leq \alpha$ (Figure 3.4):

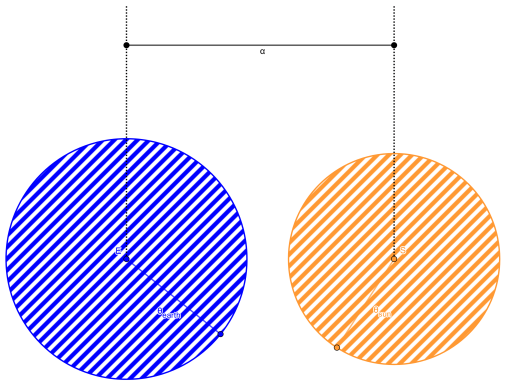

Figure 3.4. $\theta_{\text{sun}} + \theta_{\text{earth}} \leq \alpha$: the two visual cones don't have intersection, so the satellite is fully illuminated.

$$\nu = 1 \tag{3.6}$$

(b) $|\theta_{\text{sun}} - \theta_{\text{earth}}| \geq \alpha$: the two visual cones are one inside the other.

According to which cone is inside the other and the relative position of the Sun, the Earth and the satellite, we can distinguish 3 different subcases:

i. $|\vec{r}_{\text{sat,sun}}| < |\vec{r}_{\text{earth,sat}}|$ (Figure 3.5):

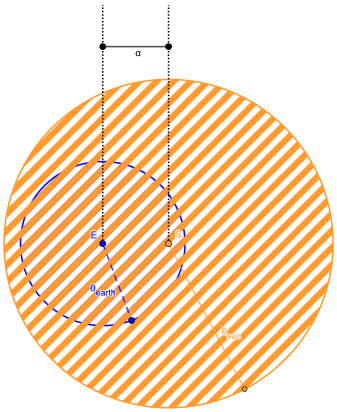

Figure 3.5. $|\theta_{\text{sun}} - \theta_{\text{earth}}| \geq \alpha$ and $|\vec{r}_{\text{sat,sun}}| < |\vec{r}_{\text{earth,sat}}|$: the Sun is between the satellite and the Earth, so the satellite is fully illuminated.

$$\nu = 1 \tag{3.7}$$

ii. $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$ and $\theta_{\text{sun}} < \theta_{\text{earth}}$ (Figure 3.6):

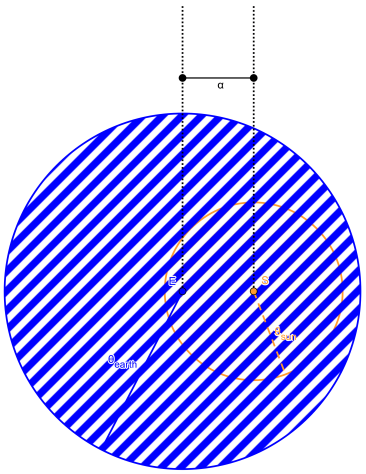

Figure 3.6. $|\theta_{\text{sun}} - \theta_{\text{earth}}| \geq \alpha$ and $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$ and $\theta_{\text{sun}} < \theta_{\text{earth}}$: the Earth is between the satellite and the Sun and covers completely the Sun (total eclipse).

$$\nu = 0 \tag{3.8}$$

iii. $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$ and $\theta_{\text{sun}} \geq \theta_{\text{earth}}$ (Figure 3.7):

$$\nu = 1 - \frac{\Omega_{\text{earth}}}{\Omega_{\text{sun}}} = 1 - \frac{\pi {\theta_{\text{earth}}}^2}{\pi {\theta_{\text{sun}}}^2} = 1 - \frac{{\theta_{\text{earth}}}^2}{{\theta_{\text{sun}}}^2} \tag{3.9}$$

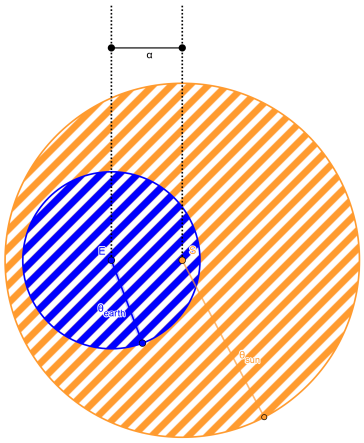

Figure 3.7. $|\theta_{\text{sun}} - \theta_{\text{earth}}| \geq \alpha$ and $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$ and $\theta_{\text{sun}} \geq \theta_{\text{earth}}$: the Earth is between the satellite and the Sun and covers partially the Sun (annular eclipse). In this case the ratio between the the smaller and the bigger circle gives the occultated fraction, so its complementary gives the fraction of the Sun that illuminates the satellite.

(c) $\theta_{\text{sun}} + \theta_{\text{earth}} > \alpha$ and $|\theta_{\text{sun}} - \theta_{\text{earth}}| < \alpha$: the two visual cones intersect but they are not one inside the other (partially overlapping). In this case we have 2 subcases:

i. $|\vec{r}_{\text{sat,sun}}| < |\vec{r}_{\text{earth,sat}}|$ (Figure 3.8):

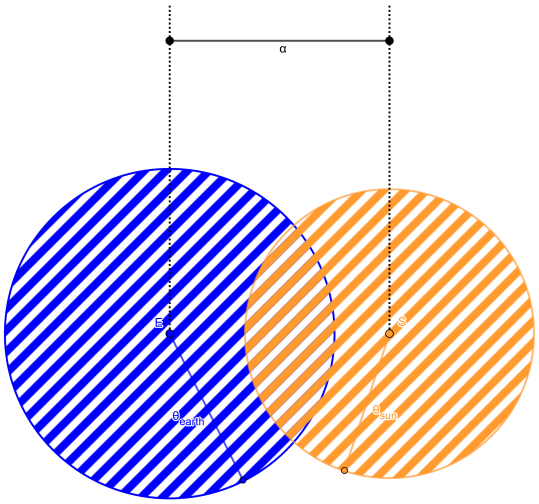

Figure 3.8. $\theta_{\text{sun}} + \theta_{\text{earth}} > \alpha$ and $|\theta_{\text{sun}} - \theta_{\text{earth}}| < \alpha$ and $|\vec{r}_{\text{sat,sun}}| < |\vec{r}_{\text{earth,sat}}|$: the Sun is between the satellite and the Earth, so the satellite is fully illuminated.

$$\nu = 1 \tag{3.10}$$

ii. $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$ (Figure 3.9):

$$\alpha_{\text{sun}} = \frac{\alpha^2 - {\theta_{\text{earth}}}^2 + {\theta_{\text{sun}}}^2}{2\alpha} \tag{3.11}$$

$$\alpha_{\text{earth}} = \frac{\alpha^2 + {\theta_{\text{earth}}}^2 - {\theta_{\text{sun}}}^2}{2\alpha} \tag{3.12}$$

$$\beta_{\text{sun}} = \arccos \frac{\alpha_{\text{sun}}}{\theta_{\text{sun}}} \tag{3.13}$$

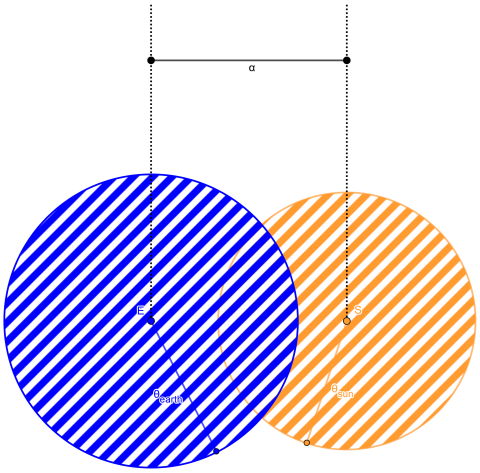

Figure 3.9. $\theta_{\text{sun}} + \theta_{\text{earth}} > \alpha$ and $|\theta_{\text{sun}} - \theta_{\text{earth}}| < \alpha$ and $|\vec{r}_{\text{sat,sun}}| \geq |\vec{r}_{\text{earth,sat}}|$: the Earth is between satellite and the Sun, so the satellite is partially illuminated (partial eclipse).

$$\beta_{\text{earth}} = \arccos \frac{\alpha_{\text{earth}}}{\theta_{\text{earth}}} \tag{3.14}$$

After some mathematical steps, we obtain the following formula for the intersection of the two circles:

$$\Omega_{\text{int}} = \frac{{\theta_{\text{earth}}}^2}{2} (2\beta_{\text{earth}} - \sin 2\beta_{\text{earth}}) + \frac{{\theta_{\text{sun}}}^2}{2} (2\beta_{\text{sun}} - \sin 2\beta_{\text{sun}}) \tag{3.15}$$

and the value of the eclipse factor is:

$$\nu = 1 - \frac{\Omega_{\text{int}}}{\Omega_{\text{sun}}} \tag{3.16}$$

3.2 Tracklet building

3.2.1 Shape of the tracklets

Before showing the algorithm, it is quite important to define the shape of a tracklet, because basing on that we can decide if a measurement can be linked to a tracklet or not.

First we imagine to have an inertial observer in the center of the Earth looking at an object orbiting around the Earth in a Keplerian orbit. Since the plane of the orbit is always the same and the observer is inertial, we expect that right ascension and declination have a periodic behaviour. In particular in one orbit we expect a single complete oscillation for declination, while a growing or decreasing trend for right ascension (depending on the direction of rotation of the object) with a jump between 360° and 0°.

However, the observer is not located in the center of the Earth: considering a ground telescope the reference frame is oriented in the same way, but its center moves together with the Earth in a circular motion with period 1 day. Since the rotation period of the Earth is very big, we expect that even in this case right ascension and declination vary

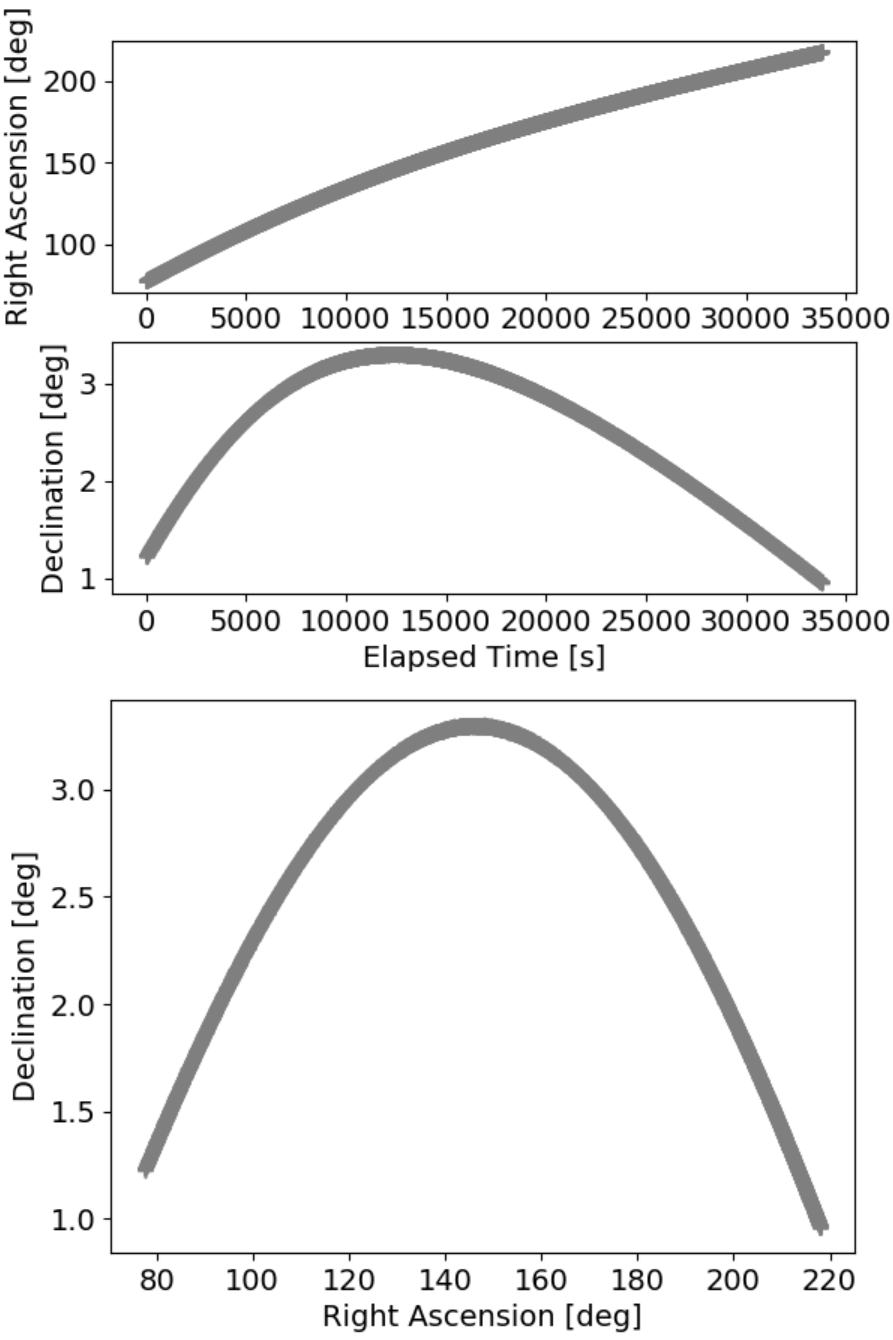

Figure 3.10. Shape of a "virtual" very long tracklet.

with a similar behaviour as for the Earth-centered observer case. This expectation is true, as we can see in Figure 3.10.

In the picture we can see a portion of a simulated GEO-transient with eccentricity 0.2. A GEO-transient has been chosen because, since its motion is almost synchronous with the Earth, the object remains in the field of view of the telescope for a long time. However, with the daylight the object can't be seen, and it is the reason why it is not shown the entire orbit, but more or less half of it. Note also that the presence of a quite high eccentricity makes the curve not perfectly sinusoidal (it is not symmetric), but a sine-like curve.

So declination is a periodic oscillating (sine-like) function of time and right ascension is growing/decreasing (almost linear with a variable small curvature in case of eccenticity) function of time.

However, this is not the right approach: we are considering the behaviour of right ascension and declination in one entire orbit, but a tracklet lasts for a short time compared with an entire revolution, so we are more interested on their local behaviour.

For right ascension the local behaviour can be considered linear or quadratic, while for the declination a more complete analysis is necessary. Let's consider the easiest periodic oscillating function, so a sine function, and consider its Taylor expansion around a point x_0:

$$\sin(x_0 + \Delta x) = \sin x_0 + \cos x_0 \Delta x - \frac{1}{2}\sin x_0 (\Delta x)^2 + o((\Delta x)^2) \tag{3.17}$$

Now let's consider two extreme cases: 0° and 90°:

$$\sin(0° + \Delta x) = \Delta x + o((\Delta x)^2) \tag{3.18}$$

$$\sin(90° + \Delta x) = 1 - \frac{1}{2}(\Delta x)^2 + o((\Delta x)^2) \tag{3.19}$$

So, depending on the initial point x_0, the behaviour can be purely linear or purely quadratic or, in the general case, a mix of both (so a complete quadratic function with also a linear term).

3.2.2 Properties of tracklets

Basing on subsection 2.2.1 this definition of tracklet can be given:

Definition 2 *A tracklet is a series of at least* 3 *consecutive observations of the same object along its orbit (usually it corresponds to a very small fragment of that orbit) made by the same observer.*

This definition is probably too generic, so we should define other properties of a tracklet, basing on what said in subsection 2.2.2 and subsection 3.2.1:

- The shape of a tracklet is linear or quadratic.
- The same measurement cannot belong to more than 1 tracklet.
- In a tracklet no more than 4 observations in consecutive pictures are missing.

- In a tracklet the time between two consecutive measurements is not more than 3 minutes.

These values are only references values, so anyone else can choose other values. These values have been chosen because otherwise the computational time would increase too much in presence of many fake detections.

Now other definitions that are used in this chapter are given:

- A closed tracklet is a tracklet that is already finished (so it contains at least 3 measurements, but it can't be extended further because of the last two properties of a tracklet).

- An established tracklet is a tracklet that MAY be extended further.

- A non-established tracklet is a series of 2 measurements that MAY belong to a tracklet (but it is not yet a tracklet because it contains less than 3 measurements). If at least another measurement is added, it becomes an established tracklet, otherwise it is discarded and deleted.

- A single measurement is a measurement not associated to any tracklet.

3.2.3 Number of combinations

To estimate the number of combinations to be computed by the algorithm, we consider first a simple case.

Imagine to have N objects observed from time t_0 to time t_m and to see all the objects in all the frames. Imagine also for simplicity that the tracklets are linear.

- At time t_0 we have N single measurements.

- At time t_1 we try to combine the N new measurements with time t_1 with the N single measurements with time t_0. Since there is always a straight line between two measurements, at time t_1 we obtain N^2 non-established tracklets (so each one with 2 measurements).

- At time t_2 we try to combine the N new measurements with time t_2 with the N^2 non-established tracklets obtained at time t_1, so the maximum number of combinations would be N^3. However, even if we have to test N^3 combinations, with the assumption of linear tracklet, only N of them give us estabished tracklets. Since the same measurement cannot belong to more than 1 tracklet and all the measurements at the first three times belong to tracklets, we can discard all the non-established tracklets.

- At time t_3 we try to combine the N new measurements with time t_3 with the N established tracklets obtained at time t_2. We have to test N^2 combinations, but only N of these combinations extend the N established tracklets.

- For the following times up to t_m the previous step is repeated.

So, except for time t_2 when we have N^3 combinations to test, for all the other times we have to test only N^2 combinations. Anyway, the number of combinations increases in a low-degree polynomial way with respect to the number of objects.

Unfortunately in reality the things are not so easy. First of all it is possible that a tracklet starts at a time different from t_0 and finishes at a time different from t_m. This facts makes the situation more difficult, but doesn't change too much the computational time. In fact, as expressed in the properties of a tracklet, if no measurements are added to a tracklet for more than 4 consecutive frames or more than 3 minutes, that tracklet is closed. In the same way, if a measurement cannot be linked to any previous tracklet, it means that it is the first measurement of a new tracklet.

Then there is the impact of the three main problems shown in subsection 2.2.2: the presence of noise in the measurements and missing points doesn't affect the computational time, while the presence of fake detections has an enormous impact on the computational time. As a matter of fact, we don't know at the beginning what are real measurements and what are fake detections. Moreover with fake detections we don't create established tracklets, so the number of possible non-established tracklets grows very fast. For this reason it is important to define from the beginning clear properties of a tracklet, in order to discard the bad combinations as soon as possible, without testing them.

To be more precise, with typical values of accepted tolerances, even fake detections MAY create "established" fake tracklets that we shouldn't consider. In subsection 3.2.6 a simple method is shown to determine which ones are probably fake tracklets in order to discard them.

3.2.4 Tracklet building algorithm

The tracklet building algorithm can be summarized in a series of steps:

- Read measurements.
- Try to link each single measurement to a tracklet. In order of priority:
  - to an established tracklet;
  - to a non-established tracklet;
  - to a single measurement.
- At the end of each time block (series of measurements with the same time) save tracklets that are closed and discard the bad non-established tracklets and the bad single measurements.

A measurement contains all the information that we need for the following steps. In particular, we get:

- The name of the observer (if known);
- The kind of observer;
- The time of each measurement;

- A list of observables for each measurement.

The name and the kind of the observer are very important for a first subdivision of the measurements, because we know that two measurements obtained by different observers can't belong to the same tracklet.

The linking is the core of the algorithm, so we should define some clear rules and properties to link a measurement to a tracklet.

In the linking we give a priority first to the established tracklets to reduce the computational time (when there are many fake detections the number of non-established tracklets can become many order of magnitude higher than the established tracklets, so we try to extend first the tracklets that probably are real).

Also the final step when we discard bad tracklets is important in order to reduce the computational time.

Figure 3.11 summarizes the algorithm using a flowchart.

In the flowchart there is a loop iterating on each "region". The explanation of what a region is and why we iterate in different regions is given in subsection 3.2.7.

In the flowchart there is also a condition "fast==True && N>max_comb". It is related to a mode of the algorithm called *fast mode*: when the *fast mode* is activated and the number of combinations is too high, all the non-established tracklets are deleted in order to reduce the number of combinations for the following steps and so to reduce the computational time.

3.2.5 Tracklet linking

This sections shows how a measurement is linked to an established or a non-established tracklet.

First the measurements are divided into time blocks (series of measurements with the same time). This division reduces a lot the computational time, because we still know that two measurements in a tracklet can't have the same time.

Then we have to decide what is the function to minimize, that is to say the quantity that we call "error". The linking error of a measurement to a tracklet is defined as the "distance" between the measurement and both the previous linear and quadratic estimation of that tracklet evaluated at the measurement time, choosing the smallest one. Depending on the kind of observables, this "distance" is measured in different ways.

For the linking, two different approaches are possible: either we can try to link each tracklet to all the measurements in a time block and choose the best one if the error is below a certain tolerance, or we can try to link each measurement in a time block to all the tracklets and choose the best one if the error is below a certain tolerance.

The first approach has a drawback: the same measurement may be linked to different tracklets, but it should belong only to one, so which one to choose? We should choose the one with the lowest fitting error, but that means that we have to save all the errors first in order to compare them. In the second approach this risk is avoided, so this one is used in the algorithm.

We call $\mathbf{r}$ the vector of observables. We start with the easiest case: radar sensor with only range ρ (one observable per measurement) First we examine a non-established tracklet (so with only two points). We have $\mathbf{r}(t_0) = \rho_0$ and $\mathbf{r}(t_1) = \rho_1$. With only two points we

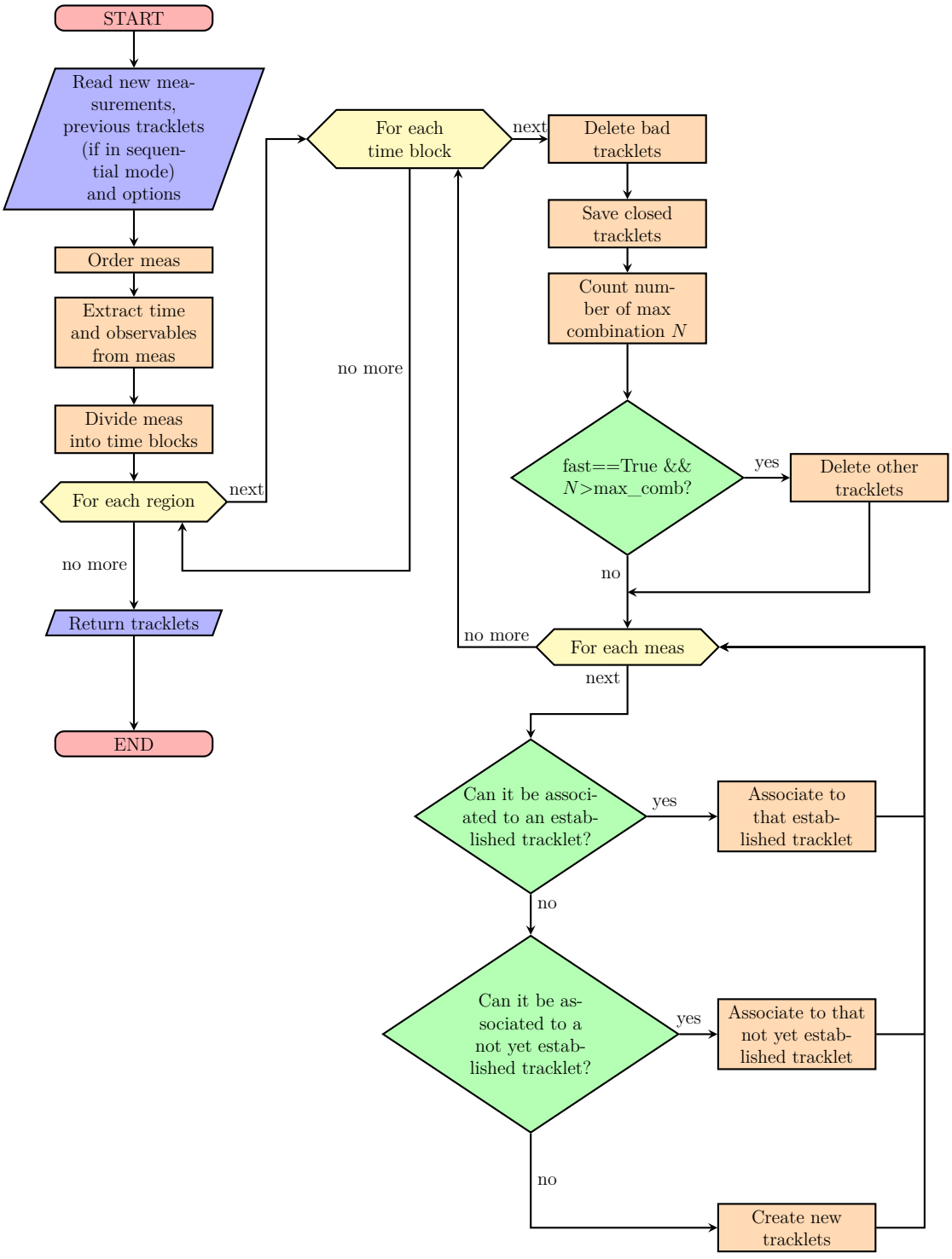

Figure 3.11. Flowchart of the tracklet building algorithm.

can have only a linear fit, so:

$$\hat{\mathbf{r}}_\mathbf{l}(t) = m \cdot t + b \qquad \text{where} \qquad \begin{cases} m = \dfrac{\rho_1 - \rho_0}{t_1 - t_0} \\ b = \rho_0 - m \cdot t_0 \end{cases} \tag{3.20}$$

where the "^" means that this is an estimation and the subscript "l" means that it is the linear estimation. For this kind of telescope, the "distance" is simply the absolute value of the difference between the measured value and the estimated value:

$$\varepsilon = \varepsilon_l = |\rho_n - \hat{\mathbf{r}}_\mathbf{l}(t_n)| \tag{3.21}$$

With three or more points we can assume a linear or a quadratic variation. We consider both the linear and quadratic estimation and compute both the errors choosing the smallest one:

$$\begin{cases} \varepsilon_l = |\rho_n - \hat{\mathbf{r}}_\mathbf{l}(t_n)| \\ \varepsilon_q = |\rho_n - \hat{\mathbf{r}}_\mathbf{q}(t_n)| \end{cases} \Rightarrow \varepsilon = \min(\varepsilon_l, \varepsilon_q) \tag{3.22}$$

For an optical sensor we have two observables: right ascension α and declination δ. The only differences are that the coefficients of the polynomials are vectors instead of scalars and we have to define the "distance" in another way. The first idea is to use the Cartesian norm:

$$\varepsilon = \sqrt{(\alpha_n - \hat{\alpha}(t_n))^2 + (\delta_n - \hat{\delta}(t_n))^2)} \tag{3.23}$$

but these observables represents points on the Celestial sphere. The best way to compute the distance between two points on a sphere is the use the Great-circle distance. The easiest formula is given by the spherical law of cosines:

$$\varepsilon = \arccos\left(\sin\delta_n \cdot \sin\hat{\delta}(t_n) + \cos\delta_n \cdot \cos\hat{\delta}(t_n) \cdot \cos\left(\alpha_n - \hat{\alpha}(t_n)\right)\right) \tag{3.24}$$

According to [14], this formula is not very good for small angles because of very large rounding errors (the cosine of a very small angle is 0.999...), so a formula that uses the sine instead of the cosine is better conditioned. He suggests to use the haversine formula for small angular distances. Using the trigonometric identity $\text{hav}(\theta) = \sin^2(\theta/2)$, we obtain the haversine formula in this form:

$$\varepsilon = 2\arcsin\sqrt{\sin^2\frac{\hat{\delta}(t_n) - \delta_n}{2} + \cos\delta_n \cdot \cos\hat{\delta}(t_n) \cdot \sin^2\frac{\hat{\alpha}(t_n) - \alpha_n}{2}} \tag{3.25}$$

For a radar sensor with range, azimuth and elevation we have three observables (one linear and two angular). We want to find the distance between two points, but we cannot mix linear and angular quantities. So first we measure the angle $\Delta\theta$ between the two points using the same haversine formula used in the optical telescope (assuming constant range), then multiplying by the range of the first measurement ρ_A we obtain the arc length. Since we are working with small angles, we can approximate the chord with the arc. Looking at Figure 3.12, we can apply the pythagorean theorem to compute first the linear distance, and then the angular distance dividing by the range.

$$\rho_A \cdot \varepsilon = \sqrt{(\rho_A \cdot \Delta\theta)^2 + (\Delta\rho)^2} \Rightarrow \varepsilon = \sqrt{(\Delta\theta)^2 + \left(\frac{\Delta\rho}{\rho_A}\right)^2} \tag{3.26}$$

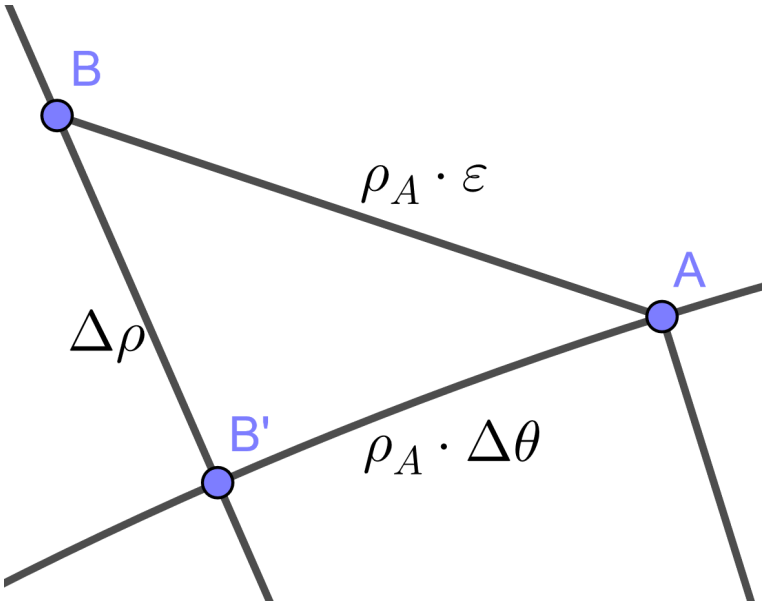

Figure 3.12. Distance between measurements Doppler radar with range, azimuth and elevation.

After a measurement is linked to a tracklet, two new fitting functions are computed (a linear one and a quadratic one) considering also the last added point, in order to have a better estimation for the following steps (the more points we consider in the fit, the better the estimation is).

3.2.6 Distinguish between real and fake tracklets

After trying to link all the measurements to tracklets, at the end we obtain a list of tracklets. Among these tracklets, how can we distinguish between real and fake tracklets?

Unfortunately we cannot be completely sure if a tracklet is real or fake, but we can determine with a certain confidence level (in a qualitative way) if a tracklet is real or not.

This can be explained better with an example: define a tolerance $\varepsilon_{\max}$ and imagine to draw n random points on a sheet of paper and to see if there are m points which lie on a line (it can be a straight line or a parabola in our case) within our defined tolerance. The probability to find m points with this property depends on various variables:

- the number of points n;
- the tolerance $\varepsilon_{\max}$;
- the number m;
- the dimension of the sheet of paper.

Increasing the first two variables the probability increases, while increasing the last two variables the probability decreases. It is clear the analogy with the tracklet building

problem: n is the total number of fake measurements, $\varepsilon_{\max}$ is the tolerance that we can accept in the fit, m is the minimum length to define a tracklet, the dimension of the sheet of paper is the field of view of the telescope. Since we said that the points are random, we want to decrease this probability, otherwise we create fake tracklets. We cannot change the total number of measurements and the field of view of the telescope, because they are given.

So, to decrease the probability to create fake tracklets, we can decrease the tolerance or increase the minimum length to define a tracklet. Both these changes have a drawback:

- if we decrease the tolerance we risk not to identify the real tracklets, because the measurement are always affected by an error and because the tracklets are not perfectly linear or parabolic (the linear or parabolic assumptions are approximations, the real shape could be obtained solving numerically the orbital dynamics equations);
- if we increase the minimum length to define a tracklet, we risk to miss the very short real tracklets.

The default value used for the tolerance have been obtained experimentally using simulated and real data, trying to reduce the number of fake tracklets without deleting real tracklets. Moreover, the best value depends also on the kind of orbit that we are considering, as we can see in Table 3.1.

	High-MEO/GEO/HEO	Medium-MEO	LEO/Low-MEO
tol	$1 \cdot 10^{-4}$ rad	$4 \cdot 10^{-4}$ rad	$2 \cdot 10^{-3}$ rad

Table 3.1. Default tolerances for different kinds of orbit.

The values are higher for lower orbits because the duration of a tracklet compared with the orbital period is higher, so the linear and quadratic approximation could not be valid anymore. To compensate this effect, we increase the tolerance in the fit.

As regards the number of measurement in a tracklet, we said in subsection 3.2.2 that the minimum number of measurements in a tracklet is 3. However, considering real cases or simulated cases with many fake detections, we obtain a large number of fake tracklets with 3 measurements, very few number of fake tracklets with 4 measurements and no fake tracklets with more than 4 measurements, so 5 can be chosen as default value for the minimum length so that we are "almost sure" that it is a real tracklet.

3.2.7 Division into regions

In Figure 3.11 the algorithm iterates not only on time blocks, but also on regions. We define a region fixing the angular velocity bounds. The angular velocity can be a good estimator of the kind of orbit that we are considering.

$$\omega = \sqrt{\frac{GM}{r^3}} \tag{3.27}$$

For a very low orbit (ex. $r = 6500\,\text{km}$), we obtain $\omega \approx 0.07\,\text{deg/s}$. This is valid for an inertial Earth-centered frame.

In the topocentic reference frame of the telescope, the relative angular velocity can be much higher, in the order of 1 deg/s.

The values shown in Table 3.2 represent approximately the range of angular velocities for different kinds of orbit (values obtained with simulated data).

	High-MEO/GEO/HEO	Medium-MEO	LEO/Low-MEO
lower bound	$5 \cdot 10^{-4}$ deg/s	$5 \cdot 10^{-3}$ deg/s	$5 \cdot 10^{-2}$ deg/s
upper bound	$6 \cdot 10^{-3}$ deg/s	$6 \cdot 10^{-2}$ deg/s	$6 \cdot 10^{-1}$ deg/s

Table 3.2. Lower and upper bound of angular velocities in a topocentric reference frame for different kinds of orbit.

Why the division into regions?

Using real data, we can notice that most of the fake tracklets that are created have a very high relative angular velocity (corresponding to the LEO region). If we define only one big region the fake tracklets could interfere and be an obstacle for the creation of real tracklets, since the same point cannot belong to more than one established tracklet. A possible solution would be to define a single region which excludes the LEO region, but in this way we would miss all the tracklets in low orbit, and we don't want that. So the adopted solution has been to divide into regions with different angular velocity bounds, giving a priority (from the first to the last). By default the priority is given for growing angular velocities, because the "slowest" tracklets are more probably real, as said in subsection 3.2.6.

Moreover, dividing into regions we can define different tolerances for the fit for each region, as explained in subsection 3.2.6.

Why three regions by default?

The number of regions is arbitrary, but for every region we have to iterate for all the measurements that were not linked to any tracklet in the previous regions, so if we increase too much the number of regions we only slow down the execution of the code without any change in the results.

Why the regions partially overlap?

Since in our approximation the observables in a tracklet change in a linear or a quadratic way, the angular velocity between two consecutive points of a tracklet is almost constant or changes very slowly, but if the regions don't overlap at all, it could happen that the angular velocity in the same tracklet goes beyond the limits of a region and enters another region, creating two separate tracklets. If we accept a partial overlapping of the regions we guarantee that a tracklet that goes beyond a limit is completely included in another region, so that the tracklet can be identified.

3.2.8 Sensitivity analysis

As said in subsection 2.2.1, one of the problem in the tracklet building algorithm is the presence of noise in the measurements: in fact the values of the observables of every single measurement are not perfect, but they are affected by an error. The presence of high noise in the measurement doesn't affect the computational time, but it can affect the final results given by the algorithm.

We can distinguish two different kinds of errors: bias errors and random errors.

- Bias errors could be only upward or only downward, so they affect the measurements in only one way. These errors are not so problematic in the tracklet building approach used in this Thesis, because they produce only a shift in the tracklets, but not a change of shape.
- Random errors could be both signs and, according to the magnitude of these errors, they can be very problematic for the tracklet building algorithm. If we don't have any other information, the best assumption is to assume a Gaussian distribution for each observable of each measurement with a mean μ and a variance σ^2.

Now we should define typical values for σ, which is the square root of the variance and it is called standard deviation. For the telescope in Spain used in this project the values for the standard deviations for both right ascension and declination are about $0.5\,\text{arcsec} \approx 2.4 \cdot 10^{-6}\,\text{rad}$, which is various orders of magnitude lower than the default values chosen for the tolerances in the fit (that means that the non-linearity of the tracklet is a much more relevant effect). This order of magnitude for errors in the measurements is typical only of optical sensors. Radar sensors, but also old optical sensors, could have much higher standard deviations (the same order of magnitude of the default values chosen for the fit), so this effect could be very relevant.

In fact, according to the Rayleigh criterion, the minimum angular distance that can be resolved is:

$$\Delta\theta = 1.220\frac{\lambda}{D} \tag{3.28}$$

where λ is the wavelength of the signal and D is the diameter of the lens.

For radars the wavelength is much higher than for optical sensors. This effect is partially but not totally compensated by the bigger dimensions of a radar, so the angular resolution is less precise.

For example, for a 80-meter radar working with a wavelength of 1 cm (small value in the radio spectrum) :

$$\Delta\theta = 1.220\frac{0.01\,\text{m}}{80\,\text{m}} \approx 2 \cdot 10^{-4}\,\text{rad} \approx 30\,\text{arcsec} \tag{3.29}$$

Notice also that the standard deviation is not the maximum possible error in the measurements. We can use the three-sigma rule [6] (68.3% of values are within 1σ from the mean, 95.5% of values are within 2σ from the mean and 99.7% of values are within 3σ from the mean), so almost all the measurements are within 3σ from the mean, but higher errors are possible (we call this measurements "outliers").

At first sight we may think that we can simply ignore the outliers, because the probability to have an outlier is very small and because the information given by an outlier is not very useful (the real value and the measured value are very far each other).

Unfortunately, the presence of an outlier in a tracklet can be very dangerous, especially if it is at the beginning of a tracklet, when it is not yet completely established, and the timespan between the first two points is small if compared with the distance between the other points.

For simplicity we consider a single observable, but this argument can be simply extended to more than one observable. A graphical explanation can be sufficient to understand what happens.

We consider various tracklets and we assume that the tracklets are perfectly linear, so we have to take into account only the effect of the errors in the measurements. We assume that all the measurements are without error (the real value is the mean value), except for one point, which is an outlier (actually we consider a point exactly at 3σ from the mean, so where the outlier region begins).

For example let's consider a high-orbit tracklet, with a constant angular velocity of $5 \cdot 10^{-3}$ deg/s in the topocentric reference frame. The results can be extended also for other values of angular velocities, but this orbit has been chosen as example because the plots are clearer.

Now we consider three different values of σ for the measurement noise: a low value, a medium value and a high value, seeing what happens.

	low error	medium error	high error
σ	0.5 arcsec	5 arcsec	30 arcsec

Table 3.3. Different values of standard deviation for different qualities of telescopes.

We consider two different cases: the first case in which the outlier is found when the tracklet is established, while the second case in which the outlier is found at the beginning of the tracklet.

In Figure 3.13 the tracklet is made up of 11 measurements, from $t = 0$ s to $t = 10$ s with an interval of 1 s. All the points in blue are on the same line, except the the red point at $t = 7$ s, which is the outlier.

In this case the outlier is not a big issue. Using the default values for the tolerances, we can see that for low and medium σ the outlier is within the tolerance, so it is accepted in the tracklet, while for high *sigma* the outlier is outside the tolerance, so it is not accepted in the tracklet. However, since the tracklet was already established, that measurement is simply ignored and doesn't affect the addition of other future measurements to that tracklet.

A different situation is when the outlier is at beginning of a tracklet, especially when there are a lot of missing points. So in Figure 3.14 we have the same situation as before, except for the fact that the outlier is at $t = 1$ s and the measurements between $t = 2$ s and $t = 5$ s are missing.

The orange dashed line represents the first estimation of the tracklet after the first two points. When the measurement error is quite high, this estimation is very bad, so the points from $t = 6$ s are not associated to that tracklet.

However, from $t = 6$ s we start another tracklet, so in this situation we lose only the first two measurements, but we don't lose the whole tracklet.

However, if we have another outlier at $t = 7$ s as in the first example, that measurement is not anymore in the middle of the tracklet, but at the beginning, so the same problem occurs.

In conclusion, the presence of outliers can affect the creation of established tracklets, with the risk to miss the initial part of a tracklet if it is long or the whole tracklet if it is

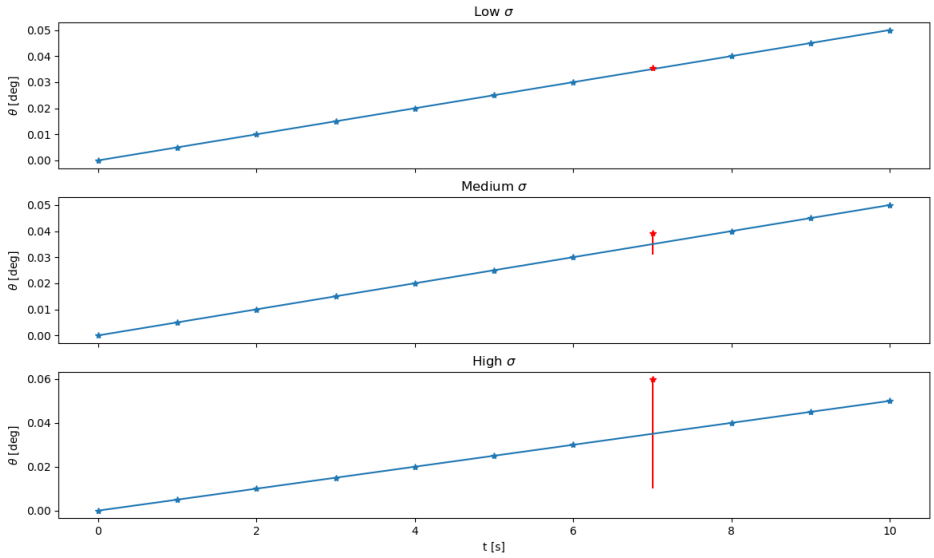

Figure 3.13. Outlier in an established tracklet, for different values of measurement noise.

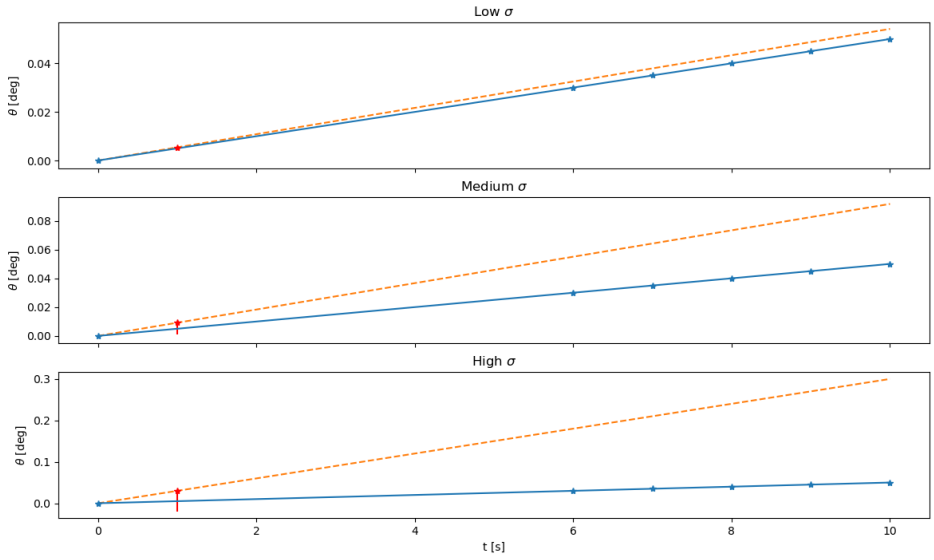

Figure 3.14. Outlier at the beginning of a tracklet, for different values of measurement noise.

short.

Luckily, for low values of measurement noise, as in the modern telescopes, the measurement noise is highly within the tolerances, and the effect of the measurement noise is small if compared with other effects (for example the error due to the linear or quadratic approximation for long tracklets).

3.3 Process noise

3.3.1 Computation of the partial derivative matrix

To compute the matrix Q we have to solve the differential Equation 2.25, that we show again:

$$\dot{Q}(t) = F(t)Q(t) + Q(t)F^T(t) + G(t)Q_{\text{proc}}(t)G^T(t) \tag{3.30}$$

To solve it, first we have to calculate $F(t)$ and $G(t)$. From the formula:

$$\dot{\mathbf{x}}(t) = F(t)\mathbf{x}(t) + G(t)\mathbf{q}(t) \tag{3.31}$$

using the total derivative theorem we obtain:

$$\begin{cases} F(t) = \left(\dfrac{\partial \dot{\mathbf{X}}(t)}{\partial \mathbf{X}(t)}\right)^* \\ G(t) = \left(\dfrac{\partial \dot{\mathbf{X}}(t)}{\partial \mathbf{q}(t)}\right)^* \end{cases} \tag{3.32}$$

where $()^*$ indicates that the partial derivatives are evaluated on the reference solution.

In this and in the following formulas the capital letters mean that we are considering the elements of the total state vector and not the difference between the state vector and the reference state.

In this subsection let's focus on the matrix F:

$$F = \left[\begin{array}{ccc|ccc} \frac{\partial \dot{X}}{\partial X} & \frac{\partial \dot{X}}{\partial Y} & \frac{\partial \dot{X}}{\partial Z} & \frac{\partial \dot{X}}{\partial U} & \frac{\partial \dot{X}}{\partial V} & \frac{\partial \dot{X}}{\partial W} \\ \frac{\partial \dot{Y}}{\partial X} & \frac{\partial \dot{Y}}{\partial Y} & \frac{\partial \dot{Y}}{\partial Z} & \frac{\partial \dot{Y}}{\partial U} & \frac{\partial \dot{Y}}{\partial V} & \frac{\partial \dot{Y}}{\partial W} \\ \frac{\partial \dot{Z}}{\partial X} & \frac{\partial \dot{Z}}{\partial Y} & \frac{\partial \dot{Z}}{\partial Z} & \frac{\partial \dot{Z}}{\partial U} & \frac{\partial \dot{Z}}{\partial V} & \frac{\partial \dot{Z}}{\partial W} \\ \hline \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial Y} & \frac{\partial \dot{U}}{\partial Z} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial V} & \frac{\partial \dot{U}}{\partial W} \\ \frac{\partial \dot{V}}{\partial X} & \frac{\partial \dot{V}}{\partial Y} & \frac{\partial \dot{V}}{\partial Z} & \frac{\partial \dot{V}}{\partial U} & \frac{\partial \dot{V}}{\partial V} & \frac{\partial \dot{V}}{\partial W} \\ \frac{\partial \dot{W}}{\partial X} & \frac{\partial \dot{W}}{\partial Y} & \frac{\partial \dot{W}}{\partial Z} & \frac{\partial \dot{W}}{\partial U} & \frac{\partial \dot{W}}{\partial V} & \frac{\partial \dot{W}}{\partial W} \end{array}\right] \tag{3.33}$$

So in the partial derivative matrix we are computing the derivatives of the velocities and the accelerations (derivative of the state vector) with respect to the positions and the velocities (the state vector). We have 4 blocks:

- derivatives of the velocities with respect to the positions;

- derivatives of the velocities with respect to the velocities;
- derivatives of the accelerations with respect to the positions;
- derivatives of the accelerations with respect to the velocities;

The last two terms depend on the model that we are considering and the different components can be coupled (for example the gravitational acceleration depends on the distance, so on the combination of the three components of the position), while the first two terms are easy to compute.

The velocities are not affected by the position, so the derivatives are 0, while the derivative of one component of the velocity with respect to itself is 1 and with respect to another component is 0. So we can write the partial derivative matrix in this form:

$$F = \left[\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial Y} & \frac{\partial \dot{U}}{\partial Z} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial V} & \frac{\partial \dot{U}}{\partial W} \\ \frac{\partial \dot{V}}{\partial X} & \frac{\partial \dot{V}}{\partial Y} & \frac{\partial \dot{V}}{\partial Z} & \frac{\partial \dot{V}}{\partial U} & \frac{\partial \dot{V}}{\partial V} & \frac{\partial \dot{V}}{\partial W} \\ \frac{\partial \dot{W}}{\partial X} & \frac{\partial \dot{W}}{\partial Y} & \frac{\partial \dot{W}}{\partial Z} & \frac{\partial \dot{W}}{\partial U} & \frac{\partial \dot{W}}{\partial V} & \frac{\partial \dot{W}}{\partial W} \end{array}\right] \tag{3.34}$$

This is for the State Noise Compensation case.

For the Dynamical Model Compensation, we add the deterministic part of the unmodelled accelerations into the state vector:

$$F = \left[\begin{array}{ccc|ccc|ccc} \frac{\partial \dot{X}}{\partial X} & \frac{\partial \dot{X}}{\partial Y} & \frac{\partial \dot{X}}{\partial Z} & \frac{\partial \dot{X}}{\partial U} & \frac{\partial \dot{X}}{\partial V} & \frac{\partial \dot{X}}{\partial W} & \frac{\partial \dot{X}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{X}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{X}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{Y}}{\partial X} & \frac{\partial \dot{Y}}{\partial Y} & \frac{\partial \dot{Y}}{\partial Z} & \frac{\partial \dot{Y}}{\partial U} & \frac{\partial \dot{Y}}{\partial V} & \frac{\partial \dot{Y}}{\partial W} & \frac{\partial \dot{Y}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{Y}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{Y}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{Z}}{\partial X} & \frac{\partial \dot{Z}}{\partial Y} & \frac{\partial \dot{Z}}{\partial Z} & \frac{\partial \dot{Z}}{\partial U} & \frac{\partial \dot{Z}}{\partial V} & \frac{\partial \dot{Z}}{\partial W} & \frac{\partial \dot{Z}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{Z}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{Z}}{\partial a_{\mathrm{un,Z}}} \\ \hline \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial Y} & \frac{\partial \dot{U}}{\partial Z} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial V} & \frac{\partial \dot{U}}{\partial W} & \frac{\partial \dot{U}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{U}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{U}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{V}}{\partial X} & \frac{\partial \dot{V}}{\partial Y} & \frac{\partial \dot{V}}{\partial Z} & \frac{\partial \dot{V}}{\partial U} & \frac{\partial \dot{V}}{\partial V} & \frac{\partial \dot{V}}{\partial W} & \frac{\partial \dot{V}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{V}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{V}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{W}}{\partial X} & \frac{\partial \dot{W}}{\partial Y} & \frac{\partial \dot{W}}{\partial Z} & \frac{\partial \dot{W}}{\partial U} & \frac{\partial \dot{W}}{\partial V} & \frac{\partial \dot{W}}{\partial W} & \frac{\partial \dot{W}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{W}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{W}}{\partial a_{\mathrm{un,Z}}} \\ \hline \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial X} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial Y} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial Z} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial U} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial V} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial W} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial X} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial Y} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial Z} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial U} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial V} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial W} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{a}_{\mathrm{un,Y}}}{\partial a_{\mathrm{un,Z}}} \\ \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial X} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial Y} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial Z} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial U} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial V} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial W} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial a_{\mathrm{un,X}}} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial a_{\mathrm{un,Y}}} & \frac{\partial \dot{a}_{\mathrm{un,Z}}}{\partial a_{\mathrm{un,Z}}} \end{array}\right] \tag{3.35}$$

In this cases other 5 blocks are added:

- derivatives of the velocities with respect to the deterministic unmodelled accelerations;
- derivatives of the accelerations with respect to the deterministic unmodelled accelerations;
- derivatives of the deterministic unmodelled jerk with respect to the positions;
- derivatives of the deterministic unmodelled jerk with respect to the velocities;
- derivatives of the deterministic unmodelled jerk with respect to the deterministic unmodelled accelerations;

The first term is 0 because the velocities are not directly dependent on the deterministic unmodelled accelerations.

The second term is 1 because the accelerations are the sum of three terms:

- modelled accelerations;
- deterministic part of the unmodelled accelerations;
- stochastic part of the unmodelled accelerations;

but only the second term is different from zero (it is the derivative of a variable with respect to itself).

The third and the fourth term are 0 because the deterministic unmodelled accelerations are not directly dependent on the positions and the velocities.

For the fifth term, considering the first component, we can write:

$$\frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial a_{\mathrm{un,X}}} = \frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial t} \cdot \frac{\partial t}{\partial a_{\mathrm{un,X}}} = \frac{\frac{\partial \dot{a}_{\mathrm{un,X}}}{\partial t}}{\frac{\partial a_{\mathrm{un,X}}}{\partial t}} = \frac{\frac{\partial^2 a_{\mathrm{un,X}}}{\partial t^2}}{\frac{\partial a_{\mathrm{un,X}}}{\partial t}} = \frac{\beta^2 a_{\mathrm{unmod}}(t_0)\mathrm{e}^{-\beta(t-t_0)}}{-\beta a_{\mathrm{unmod}}^2(t_0)\mathrm{e}^{-\beta(t-t_0)}} = -\beta \tag{3.36}$$

So, putting together all the terms, we obtain:

$$F = \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial Y} & \frac{\partial \dot{U}}{\partial Z} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial V} & \frac{\partial \dot{U}}{\partial W} & 1 & 0 & 0 \\ \frac{\partial \dot{V}}{\partial X} & \frac{\partial \dot{V}}{\partial Y} & \frac{\partial \dot{V}}{\partial Z} & \frac{\partial \dot{V}}{\partial U} & \frac{\partial \dot{V}}{\partial V} & \frac{\partial \dot{V}}{\partial W} & 0 & 1 & 0 \\ \frac{\partial \dot{W}}{\partial X} & \frac{\partial \dot{W}}{\partial Y} & \frac{\partial \dot{W}}{\partial Z} & \frac{\partial \dot{W}}{\partial U} & \frac{\partial \dot{W}}{\partial V} & \frac{\partial \dot{W}}{\partial W} & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & -\beta & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\beta \end{array}\right] \tag{3.37}$$

3.3.2 Computation of the process noise mapping matrix

The mapping matrix G is constructed similarly to F, but in this case the derivatives are computed not with respect to the state vector, but with respect to the process noise vector q (which contains the standard deviations of the unmodelled accelerations):

$$G = \left[\begin{array}{ccc} \frac{\partial \dot{X}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{X}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{X}}{\partial \sigma_{\ddot{Z}}} \\ \frac{\partial \dot{Y}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{Y}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{Y}}{\partial \sigma_{\ddot{Z}}} \\ \frac{\partial \dot{Z}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{Z}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{Z}}{\partial \sigma_{\ddot{Z}}} \\ \hline \frac{\partial \dot{U}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{U}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{U}}{\partial \sigma_{\ddot{Z}}} \\ \frac{\partial \dot{V}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{V}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{V}}{\partial \sigma_{\ddot{Z}}} \\ \frac{\partial \dot{W}}{\partial \sigma_{\ddot{X}}} & \frac{\partial \dot{W}}{\partial \sigma_{\ddot{Y}}} & \frac{\partial \dot{W}}{\partial \sigma_{\ddot{Z}}} \end{array}\right] \tag{3.38}$$

In a way similar to the computation of the F matrix, the derivatives of the velocities with respect to the process noise vector are 0, while the derivatives of the accelerations with respect to the process noise vector are 1. In fact, as in the previous case, the accelerations are the sum of three terms:

- modelled accelerations;
- deterministic part of the unmodelled accelerations;
- stochastic part of the unmodelled accelerations;

and in this case the third term is different from zero (it is the derivative of a variable with respect to itself).

So we can write:

$$G = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \tag{3.39}$$

For the Dynamical Model Compensation, we add also the derivatives of the deterministic

unmodelled jerks with respect to q, which are 0. So we obtain:

$$G = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \tag{3.40}$$

.

In both cases the part relative to the accelerations is the identity matrix. This is true when the process noise is expressed in *ICRF*. However, in most common cases, the perturbations have one or two main directions in *RTN*, so it is better to express the q vector in *RTN* instead of *ICRF*. However, since the state vector is expressed in *ICRF*, we need a change of coordinates that we can include in the mapping matrix:

$$G_{\text{RTN}} \cdot q_{\text{RTN}} = G_{\text{ICRF}} \cdot q_{\text{ICRF}} = G_{\text{ICRF}} \cdot C_{\text{RTN2ICRF}} \cdot q_{\text{RTN}} \Rightarrow G_{\text{RTN}} = G_{\text{ICRF}} \cdot C_{\text{RTN2ICRF}} \tag{3.41}$$

where C_{RTN2ICRF} is the rotation matrix from *RTN* to *ICRF* and the symbol "$\cdot$" is used for matrix multiplication even though not necessary (only for clarity).

So, considering the special structure of G_{ICRF}, we obtain for State Noise Compensation:

$$G_{\text{RTN}} = \begin{bmatrix} O_{3\times3} \\ C_{\text{RTN2ICRF}} \end{bmatrix} \tag{3.42}$$

and for Dynamical Model Compensation:

$$G_{\text{RTN}} = \begin{bmatrix} O_{3\times3} \\ C_{\text{RTN2ICRF}} \\ O_{3\times3} \end{bmatrix} \tag{3.43}$$

3.3.3 Process noise for ballistic coefficient and solar radiation pressure coefficient

How matrices F and G change if we introduce in the state vector also the ballistic coefficient and the solar radiation pressure coefficient?

For simplicity we consider a 1D case, so the state vector has this form:

$$\mathbf{x} = \begin{bmatrix} \Delta X \\ \Delta U \\ \Delta C_{\text{drag}} \\ \Delta C_{\text{SRP}} \\ \Delta a_{\text{un}} \end{bmatrix} \tag{3.44}$$

The matrix F will have this form:

$$F = \begin{bmatrix} \frac{\partial \dot{X}}{\partial X} & \frac{\partial \dot{X}}{\partial U} & \frac{\partial \dot{X}}{\partial C_{\text{drag}}} & \frac{\partial \dot{X}}{\partial C_{\text{SRP}}} & \frac{\partial \dot{X}}{\partial a_{\text{un}}} \\ \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial C_{\text{drag}}} & \frac{\partial \dot{U}}{\partial C_{\text{SRP}}} & \frac{\partial \dot{U}}{\partial a_{\text{un}}} \\ \frac{\partial \dot{C}_{\text{drag}}}{\partial X} & \frac{\partial \dot{C}_{\text{drag}}}{\partial U} & \frac{\partial \dot{C}_{\text{drag}}}{\partial C_{\text{drag}}} & \frac{\partial \dot{C}_{\text{drag}}}{\partial C_{\text{SRP}}} & \frac{\partial \dot{C}_{\text{drag}}}{\partial a_{\text{un}}} \\ \frac{\partial \dot{C}_{\text{SRP}}}{\partial X} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial U} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial C_{\text{drag}}} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial C_{\text{SRP}}} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial a_{\text{un}}} \\ \frac{\partial \dot{a}_{\text{un}}}{\partial X} & \frac{\partial \dot{a}_{\text{un}}}{\partial U} & \frac{\partial \dot{a}_{\text{un}}}{\partial C_{\text{drag}}} & \frac{\partial \dot{a}_{\text{un}}}{\partial C_{\text{SRP}}} & \frac{\partial \dot{a}_{\text{un}}}{\partial a_{\text{un}}} \end{bmatrix} \tag{3.45}$$

If we assume dynamical model compensation for the two new elements that we have introduced in the state vector, we obtain the same expression as for the unmodelled acceleration, but in a scalar form. Obviously the three phenomena (unmodelled accelerations, perturbations in the ballistic coefficient and in the solar radiation pressure coefficient) are independent, so different values of σ and β should be used.

At the end we obtain this form:

$$F = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ \frac{\partial \dot{U}}{\partial X} & \frac{\partial \dot{U}}{\partial U} & \frac{\partial \dot{U}}{\partial C_{\text{drag}}} & \frac{\partial \dot{U}}{\partial C_{\text{SRP}}} & 1 \\ 0 & 0 & -\beta_{\text{drag}} & 0 & 0 \\ 0 & 0 & 0 & -\beta_{\text{SRP}} & 0 \\ 0 & 0 & 0 & 0 & -\beta_{\ddot{X}} \end{bmatrix} \tag{3.46}$$

where $\frac{\partial \dot{U}}{\partial C_{\text{drag}}}$ and $\frac{\partial \dot{U}}{\partial C_{\text{SRP}}}$ depend on the reference model that we are considering.

A similar analysis can be done for the mapping matrix G:

$$G = \begin{bmatrix} \frac{\partial \dot{X}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{X}}{\partial \sigma_{\text{SRP}}} & \frac{\partial \dot{X}}{\partial \sigma_{\ddot{X}}} \\ \frac{\partial \dot{U}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{U}}{\partial \sigma_{\text{SRP}}} & \frac{\partial \dot{U}}{\partial \sigma_{\ddot{X}}} \\ \frac{\partial \dot{C}_{\text{drag}}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{C}_{\text{drag}}}{\partial \sigma_{\text{SRP}}} & \frac{\partial \dot{C}_{\text{drag}}}{\partial \sigma_{\ddot{X}}} \\ \frac{\partial \dot{C}_{\text{SRP}}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial \sigma_{\text{SRP}}} & \frac{\partial \dot{C}_{\text{SRP}}}{\partial \sigma_{\ddot{X}}} \\ \frac{\partial \dot{a}_{\text{un}}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{a}_{\text{un}}}{\partial \sigma_{\text{SRP}}} & \frac{\partial \dot{a}_{\text{un}}}{\partial \sigma_{\ddot{X}}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{\partial \dot{U}}{\partial \sigma_{\text{drag}}} & \frac{\partial \dot{U}}{\partial \sigma_{\text{SRP}}} & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.47}$$

The ballistic coefficient and the solar radiation pressure coefficient that we estimate in the state vector are only the deterministic part. The stochastic part affects the accelerations in the same way as the deterministic part, so we can write $\frac{\partial \dot{U}}{\partial \sigma_{\text{drag}}} = \frac{\partial \dot{U}}{\partial C_{\text{drag}}}$ and $\frac{\partial \dot{U}}{\partial \sigma_{\text{SRP}}} = \frac{\partial \dot{U}}{\partial C_{\text{SRP}}}$.

3.3.4 Choice of the tuning parameters

The choice of the tuning parameters is one of the main problems when we consider the process noise. For both SNC and DMC the standard deviations of the unmodelled accelerations are tuning parameters. Moreover, for DMC we add another tuning parameters, which is β.

Let's focus on the SNC, so we have to tune only the standard deviations (which correspond to the vector q).

Unfortunately, the tuning of these parameters is not easy because it depends on the model that we are considering: the more the model is similar to the reality, the lower values we have for the standard deviations (because the unmodelled accelerations are in a smaller range). The only thing that we can do is an estimation of these parameters, determining the order of magnitude.

To compare the performances using different values of the tuning parameters, we use simulated data: the "real" orbit is determined considering the propagation of the state vector using the "best" model to compute the accelerations, while for orbit determination we use a "worse" model adding process noise to handle the uncertainties in the model.

Choosing proper values of the tuning parameters we expect:

- a reduction of the "error" (difference between real and estimated) with respect to the case where we don't take into account the process noise;
- a covariance that estimates properly the "error" (with neither underestimation nor overestimation, otherwise it is useless).

The "best" values of the tuning parameters used in section 4.3 have been obtained experimentally only for those two cases, showing the advantages of using the process noise.

However, an optimization problem to choose automatically the best specific tuning parameters for each situation would be better (not treated in this Thesis).

Chapter 4

Results

4.1 Shadow function

The shadow function algorithm has been implemented in *Fortran* and tested with random values of the relative distances, comparing the results with the *Java* tool *Orekit*, specific for aerospace applications. The results for the *Fortran* code and the *Orekit* tool are the same within a certain margin of tolerance due to numerical errors.

Instead of considering random points, if we fix a certain number of distances from the center of the Earth along the shadow cone axis and then move radially on a plane perpendicular to the cone axis we can compute the lighting ratio and see how large the shadow and the semi-shadow regions are for various distances.

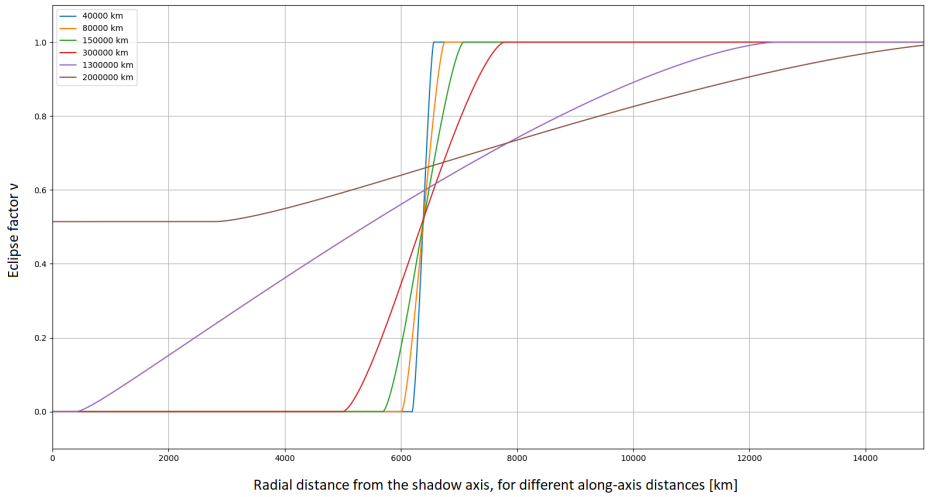

Figure 4.1. Eclipse factor as a function of the radial distance from the shadow axis of the Earth, for several along-axis distances.

As we can see from Figure 4.1, the semi-shadow region is very small if compared with the shadow region for typical distances of satellites and space debris (for example the blue line corresponds to a geostationary distance and all the other lines correspond to higher distances, that are unrealistic and in practice useless for common cases). So we could think that the cylindrical shadow model would be good for common practical cases and much easier and computationally faster.

However, we don't have to forget that a space object could be occulted not only by the Earth, but also by the Moon. Using Equation 2.11 with the radius of the Moon (2000 km) instead of the Earth one, we obtain a distance of $\approx 3.74 \cdot 10^5$ km between the center of the Moon and its shadow cone vertex. This value is between the minimum and the maximum distance between the Earth and the Moon. So for typical orbits of space objects the shadow region produced by the Moon can be absent or very small if compared with the semi-shadow region. So, for the case when the Moon is between the Earth and the Sun, and a space object is between the Earth and the Moon, we can plot the eclipse factor for several distances from the Earth (assuming a fixed distance of $3.85 \cdot 10^5$ km between the Earth and the Moon).

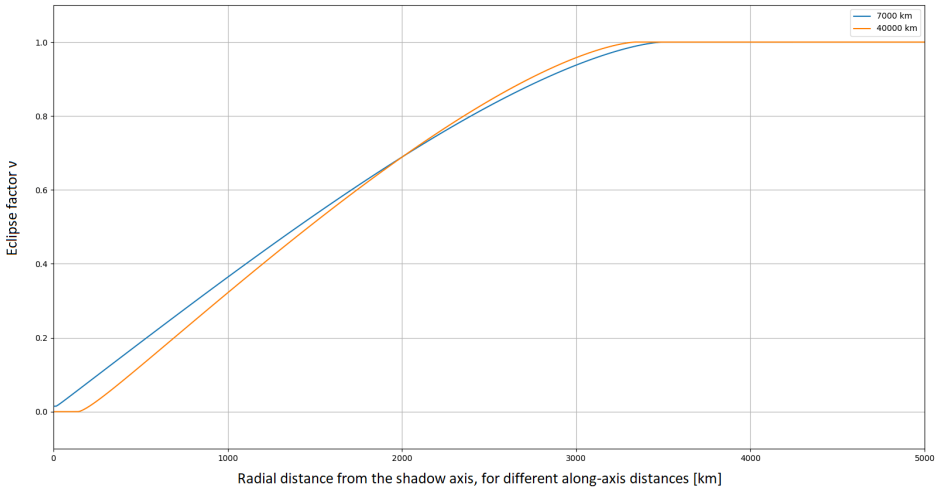

Figure 4.2. Eclipse factor as a function of the radial distance from the shadow axis of the Moon, for several along-axis distances (in the legend the distance from the Earth is indicated to understand what kind of orbit we are considering).

As we can see from Figure 4.2, the semi-shadow region is quite large for two typical kinds of orbit: a low-Earth orbit and a geostationary orbit.

4.2 Tracklet building

This sections shows various results obtained using the tracklet building algorithm, for both simulated and real data:

- Each figure contains some subfigures: all the angles separately as a function of time (for the radar also the range), and a summary plot of one angle as a function of the other.
- Each tracklet is represented with all the measurements as circles with the same color and a line connecting them. Since the default colour palette contains a limited number of colours, different tracklets can have the same colour, but they can easily distinguished visually, because they are not connected by a line.
- As said in subsection 3.2.6, we plot only the tracklets with length at least 5, in order to discard the fake tracklets.
- The points not associated to any tracklet because fake detections are represented as small gray pluses.
- In Figure 4.5 a long line goes from one side to the other side of the graph. The algorithm can distinguish a tracklet even if there is a switch in the value of the angle due to periodicity, so the problem is only graphical.
- For some cases the tracklets are very close each other, so the whole figure makes impossible to distinguish some of the tracklets. Various zooms have been used to solve this problem.
- For simulated data we know from the start how many objects there are, but the number of tracklets is higher, because each object is detected more times in different parts of its orbit. Anyway the output of the algorithm is a list of tracklets, not of space objects, so at the moment we don't care if the tracklets are originated from the same object or from different objects. As a matter of fact, for real data we don't know from the beginning how many objects are detected, so only the number of tracklets is indicated.
- Figure 4.5 and Figure 4.9 are two challenging example, respectively for the simulated case and the real case, because there are many fake detections. In these two cases the results are good, even though the computational time is a bit higher compared to the other cases.

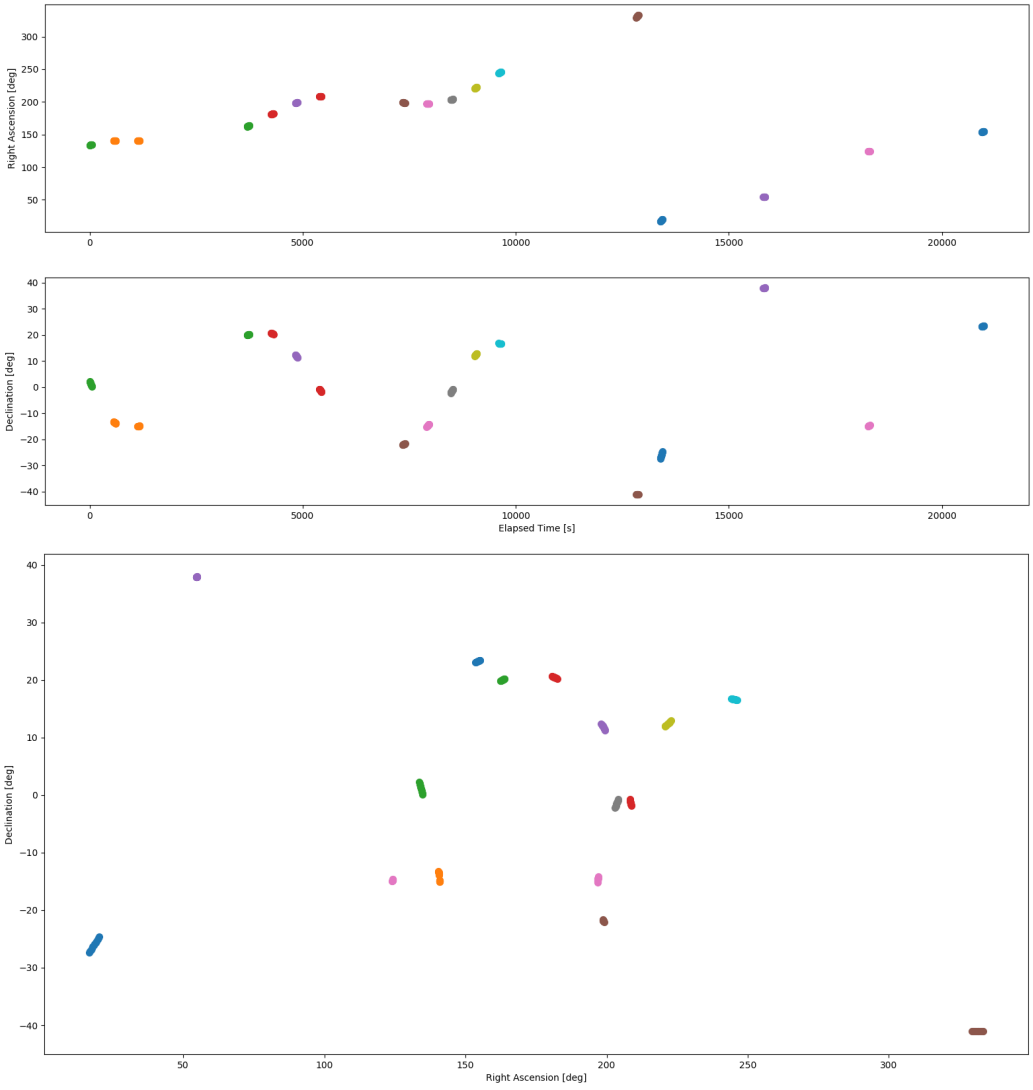

Figure 4.3. Tracklet building of simulated data: 1 MEO object observed by a space optical observer in LEO. Simulation time = 6 hours. Total number of tracklets = 17.

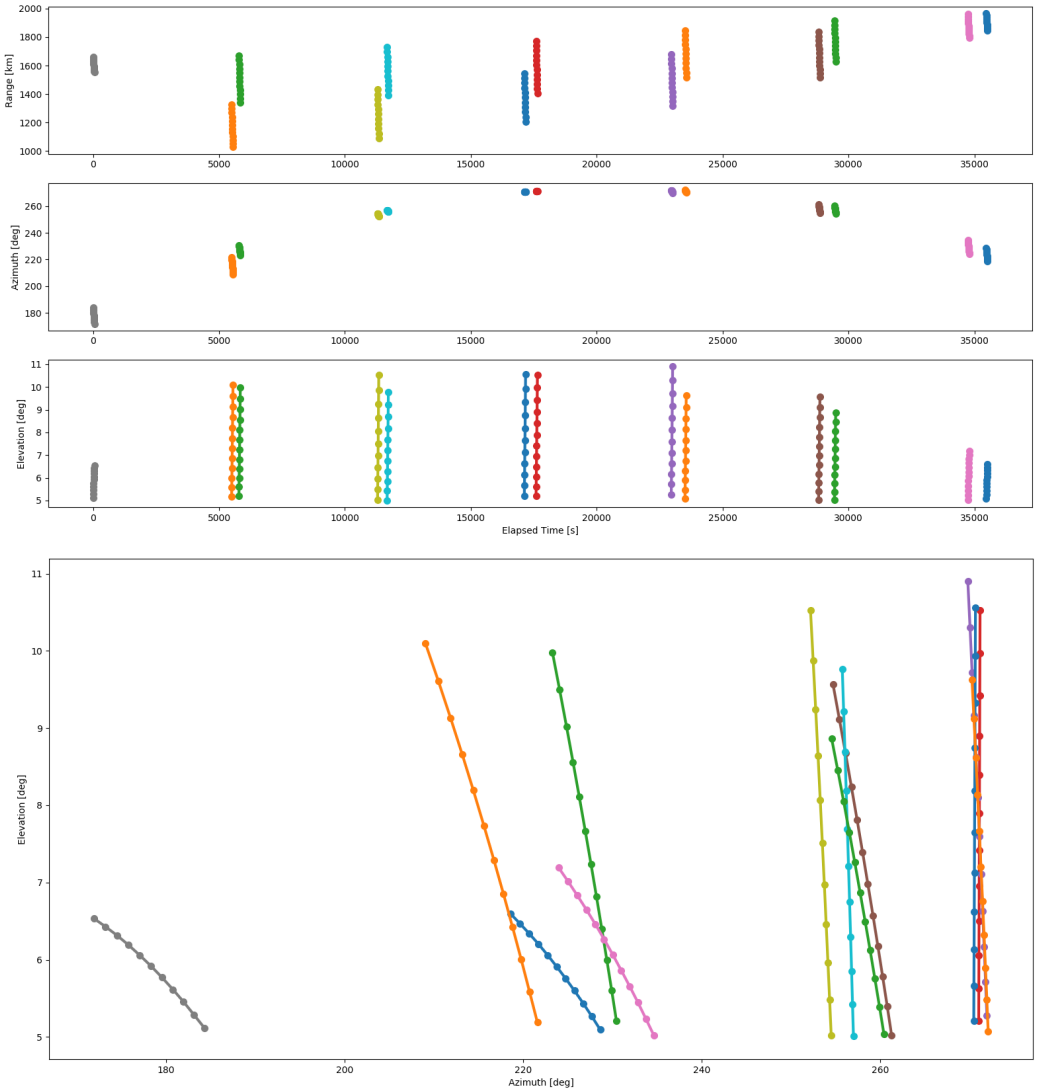

Figure 4.4. Tracklet building with simulated data: 2 objects observed by a radar telescope with range, azimuth and elevation (2 LEO objects very close each other). Simulation time = 10 hours. Total number of tracklets = 13.

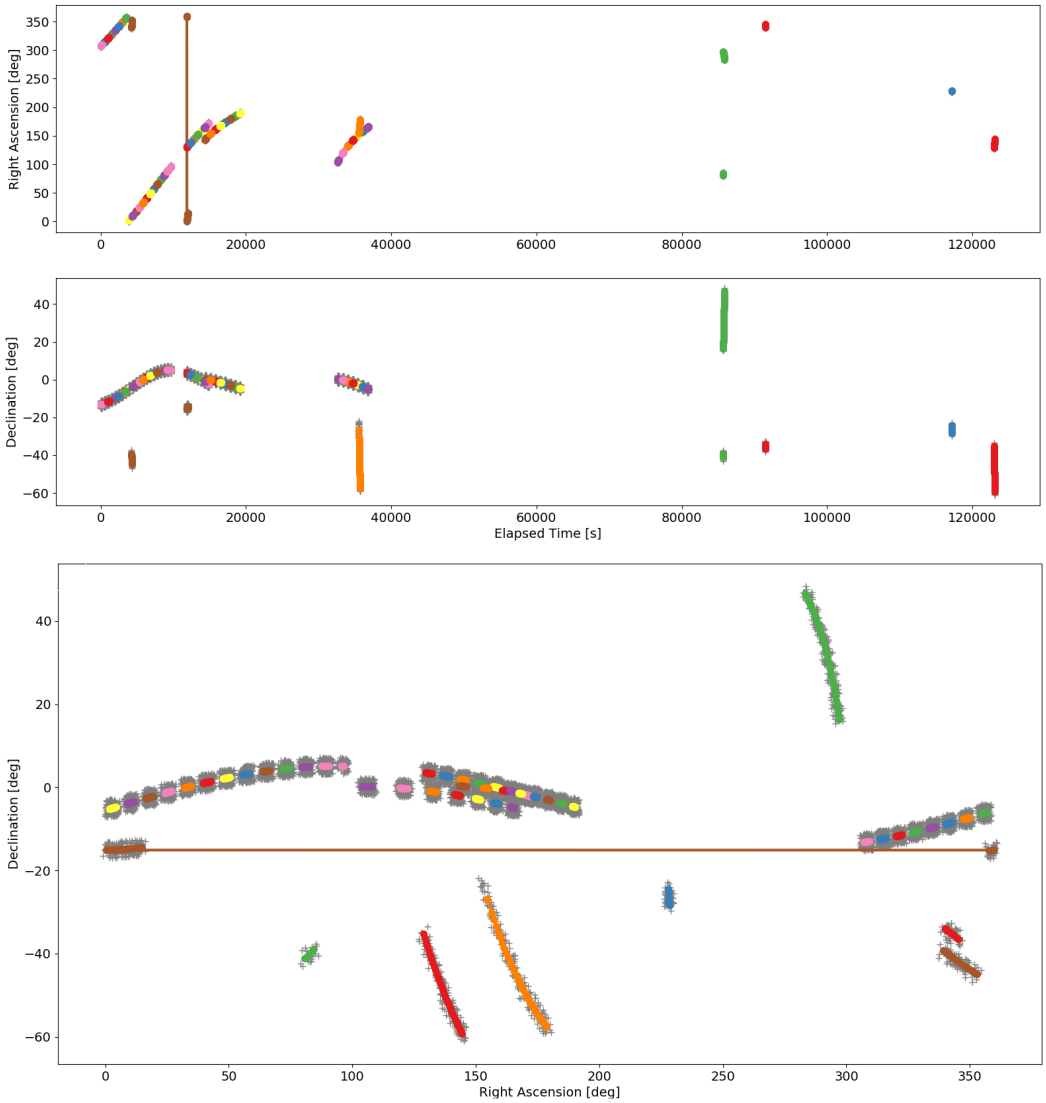

Figure 4.5. Tracklet building with simulated data with 5 close fake points per measurement: 4 objects observed by a ground optical telescope (1 LEO with low inclination, 1 LEO with high inclination, 1 MEO with low inclination, 1 LEO transient with low inclination and high eccentricity). Simulation time = 1.5 days. Total number of tracklets = 51.

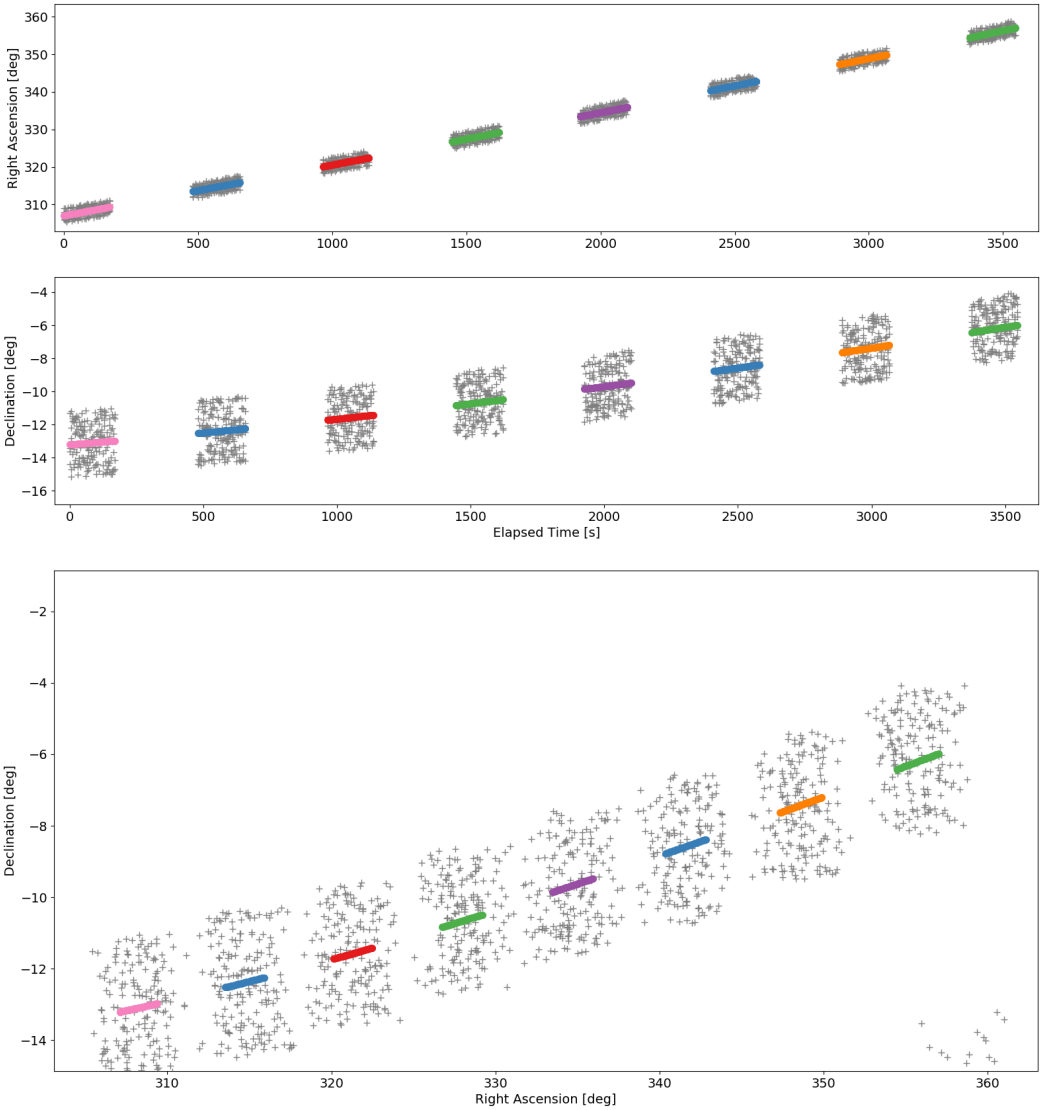

Figure 4.6. First zoom of Figure 4.5. The different tracklets can be distinguished more easily, but the single points of the tracklets cannot be distinguished.

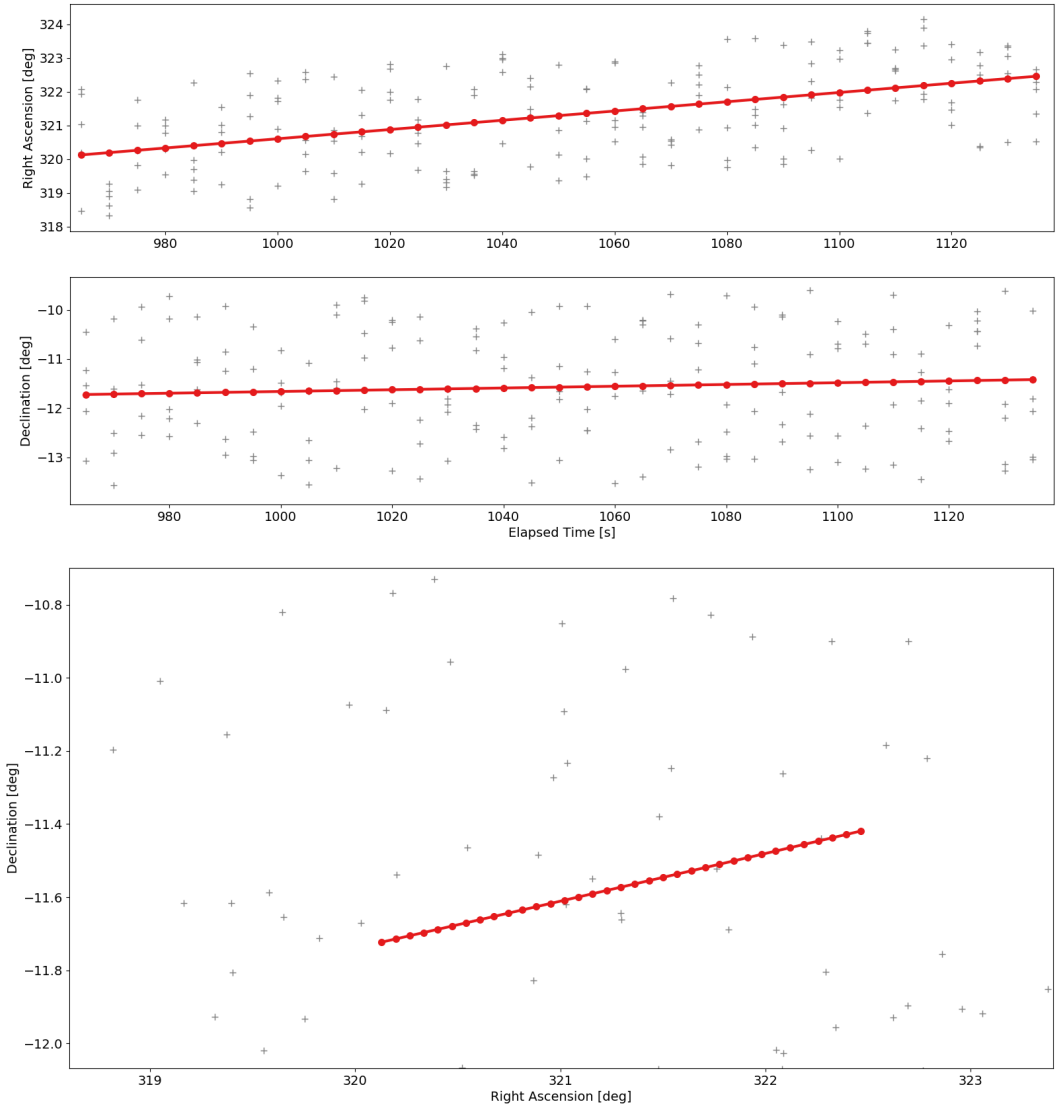

Figure 4.7. Second zoom of Figure 4.5. All the points belonging to a single tracklet can be distinguished.

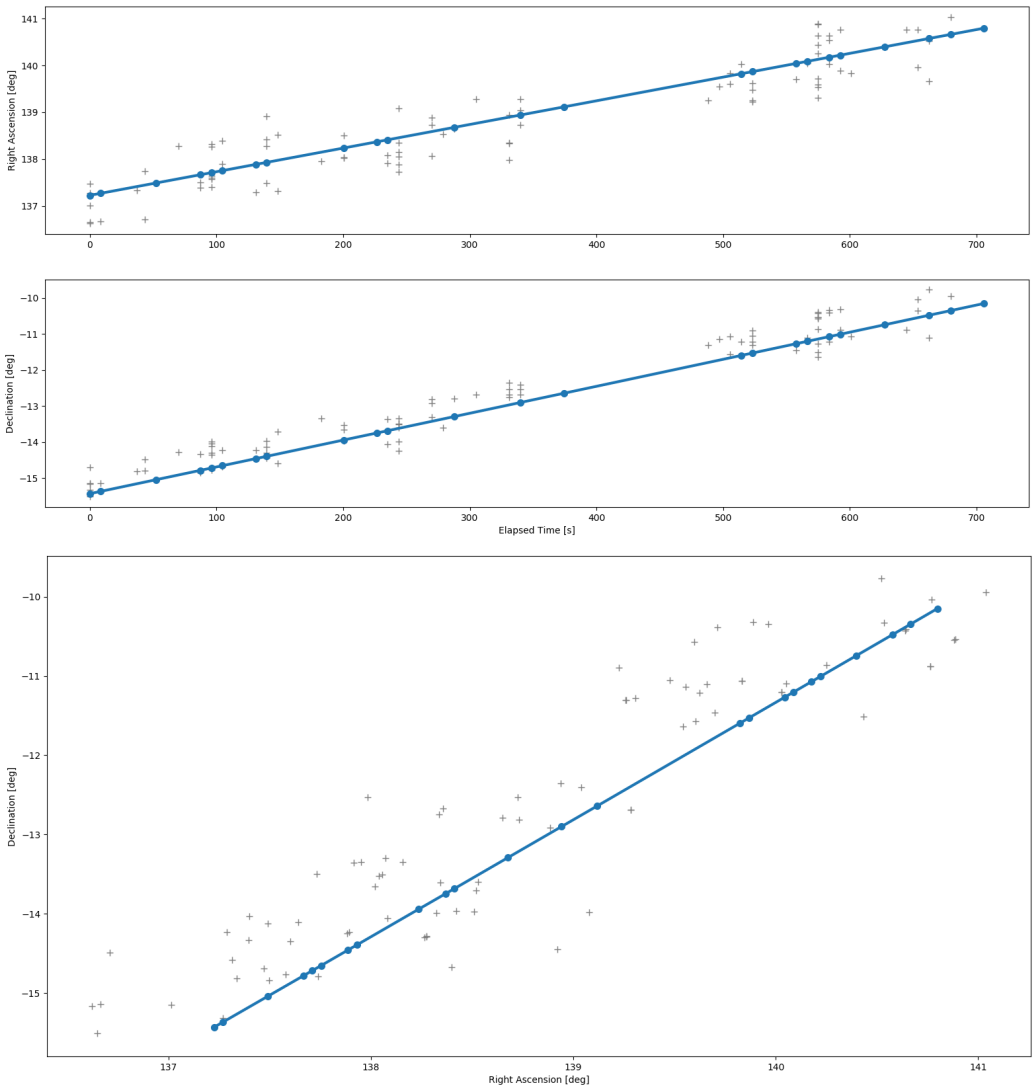

Figure 4.8. Tracklet building with real data, ground optical telescope, simple case (not many fake detections). Total number of tracklets = 1.

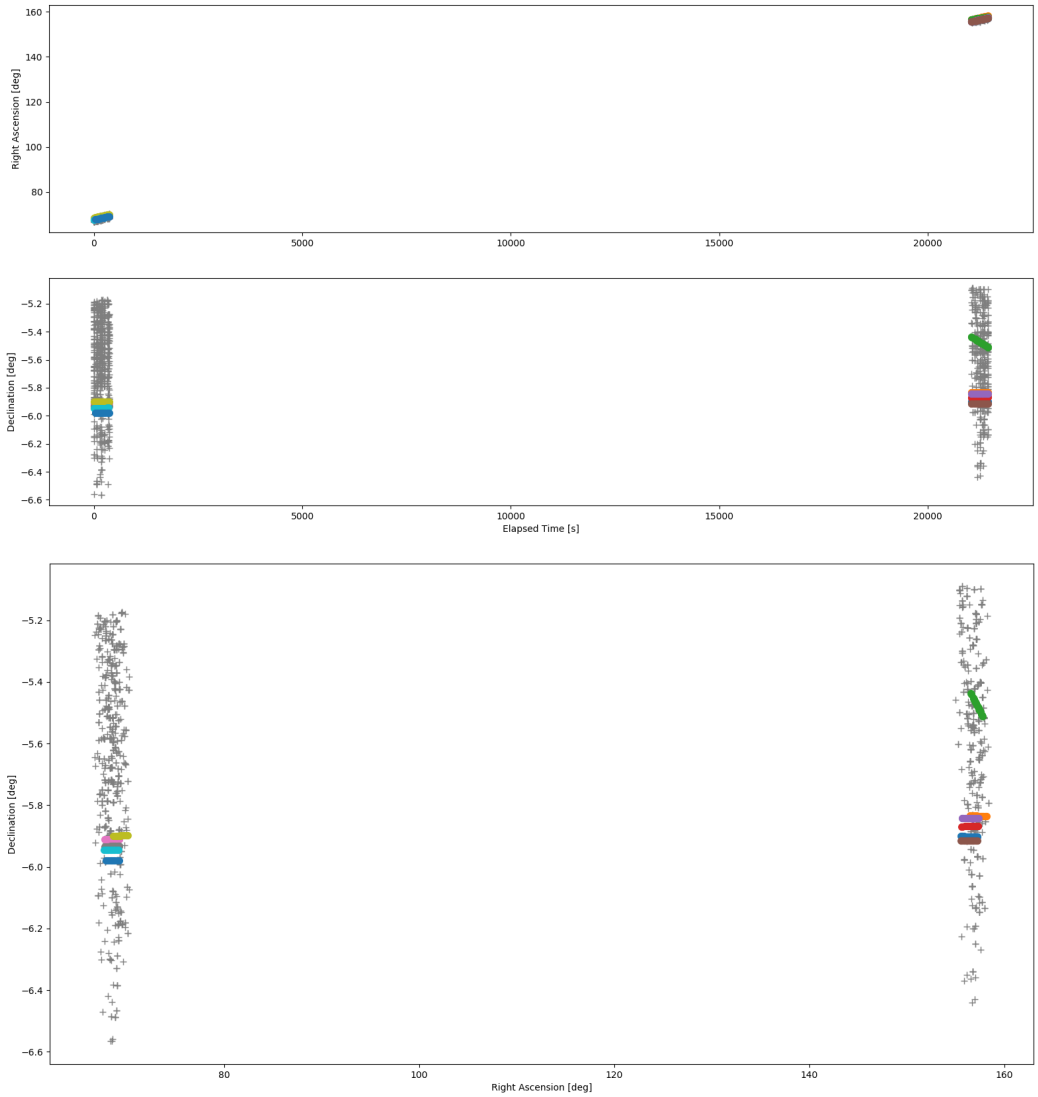

Figure 4.9. Tracklet building with real data, ground optical telescope, difficult case (many fake detections) Total number of tracklets = 11.

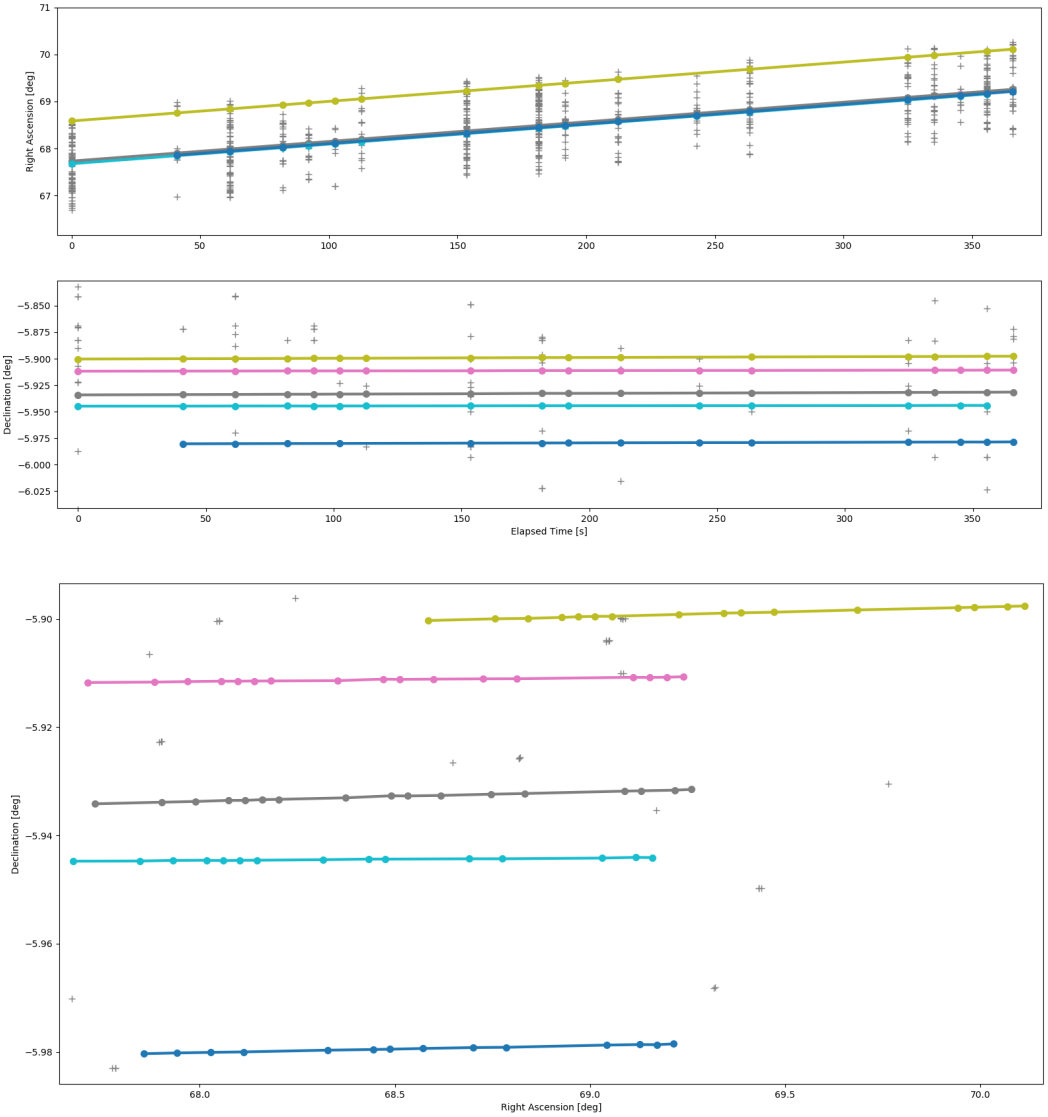

Figure 4.10. First zoom of Figure 4.9. 5 tracklets are shown in this zoom.

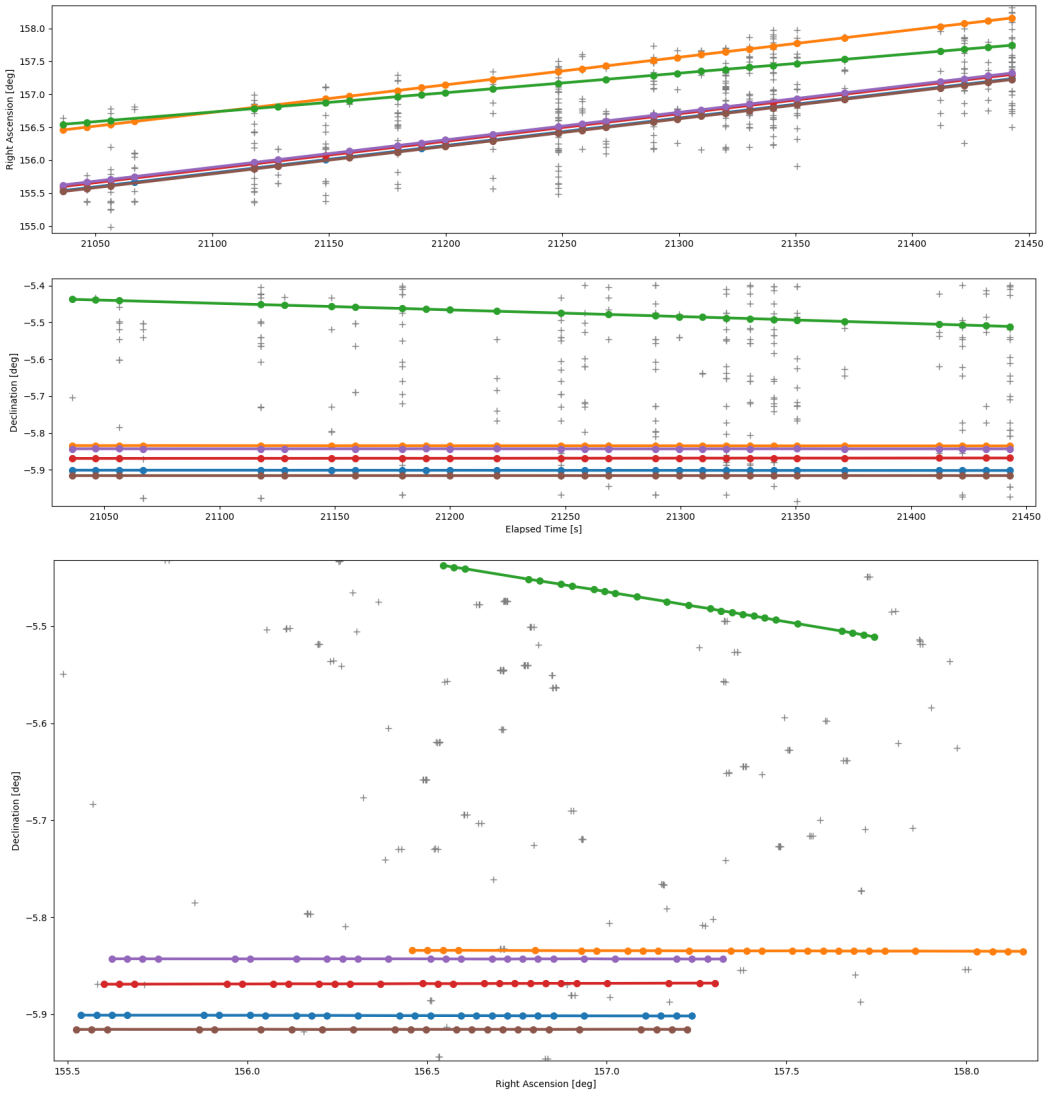

Figure 4.11. Second zoom of Figure 4.9. 6 tracklets are shown in this zoom.

4.3 Process noise

This sections shows various results obtained using the state noise compensation for two examples:

- a geostationary orbit;
- a low-Earth orbit.

For the two examples the inaccuracies in the model are treated as unmodelled perturbations, using proper tuning parameters obtained experimentally. The differences between the reference model, assumed to be as "real", and the model used for orbit determination are shown in Table 4.1.

	Reference	OD (GEO)	OD (LEO)
Gravity	EGM2008 30×30	2-body	EGM2008 30×30
Drag	NRLMSISE-00	NRLMSISE-00	None

Table 4.1. Differences between the reference model and the models used for orbit determination.

The initial orbital parameters used in the two examples are summarized in Table 4.2.

	a[km]	e	i[deg]	ν[deg]	ω[deg]	Ω[deg]
GEO	42000	0.001	40	180	0	0
LEO	7000	0.01	20	180	0	60

Table 4.2. Initial orbital elements for GEO and LEO.

Here a legend for the colours used in the following figures:

purple: real error without taking into account the process noise;

blue: real error taking into account the process noise with proper tuning parameters;

green: real error taking into account the process noise with bad tuning parameters;

yellow: 3-σ (3 standard deviations) without taking into account the process noise;

orange: 3-σ (3 standard deviations) taking into account the process noise with proper tuning parameters;

red: 3-σ (3 standard deviations) taking into account the process noise with bad tuning parameters.

The 3-σ shown in the figures don't have to be confused with the σ used as tuning parameter.

- The σ shown in the figure for each component is the square root of the corresponding element in the diagonal of the covariance matrix. This covariance matrix is the sum of two terms: one due to the measurement noise and the other due to the process noise.

- The σ used as tuning parameter is an estimation of the order of magnitude of the unmodelled accelerations. When we tune the parameter we tune the values of this σ in the three directions. The value of the σ shown in the figure is modified consequently because the covariance matrix changes.

Figure 4.12 shows the GEO example in a timespan of 3 days without taking into account the process noise. After the initial covariance determination, the 3-σ relative to each component of the position and the velocity is reduced with time because new measurements are available for the Kalman Filter. However, the real error is 3-4 order of magnitude higher than the 3-σ, so the covariance is underestimating the real error.

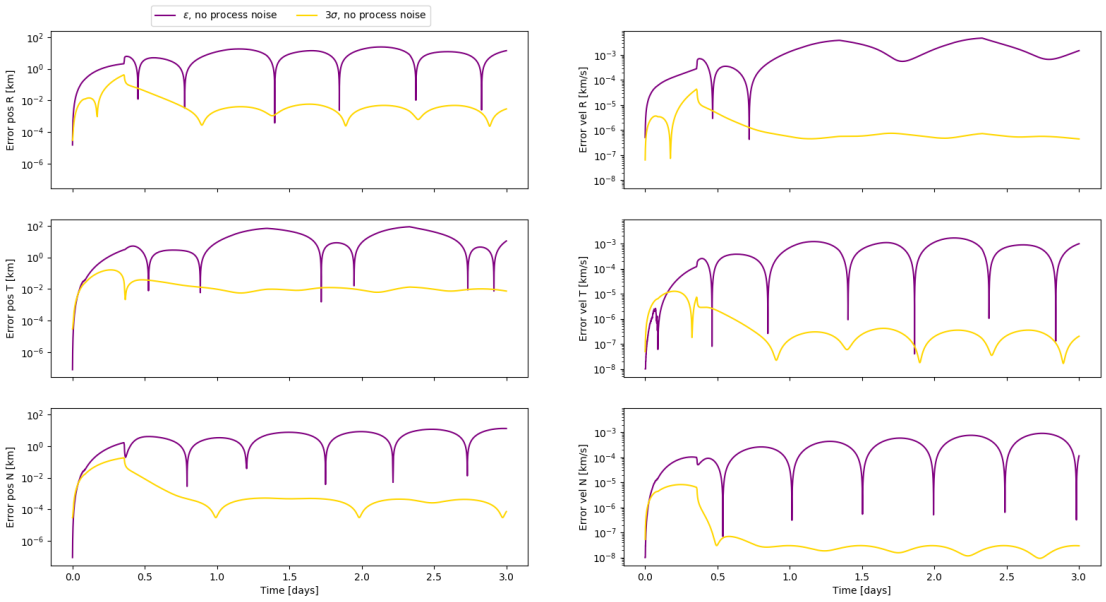

Figure 4.12. GEO example: no process noise.

In Figure 4.13 we see a comparison between the case without process noise and with a proper choice of the tuning parameters. What we notice is that the 3-σ for each component is higher than in the previous case (we are "inflating" the covariance to avoid the saturation of the filter). The advantage is that we are giving a higher weight to the new measurements with a reduction of the real error. Choosing proper values we notice also that the error and the 3-σ have the same order of magnitude, so the error is well estimated by the covariance. The values have been choosen taking into account what are the "perturbations" in our model, so the inaccuracy in the gravitational model. In our model we are ignoring the fact that the Earth is not perfectly spherical and not homogeneous, so the gravitational force has also a component in the tangential and in the normal direction. The normal

component is dominant, as we can see in the choice of the tuning parameter for the normal direction one order of magnitude higher than for the other directions.

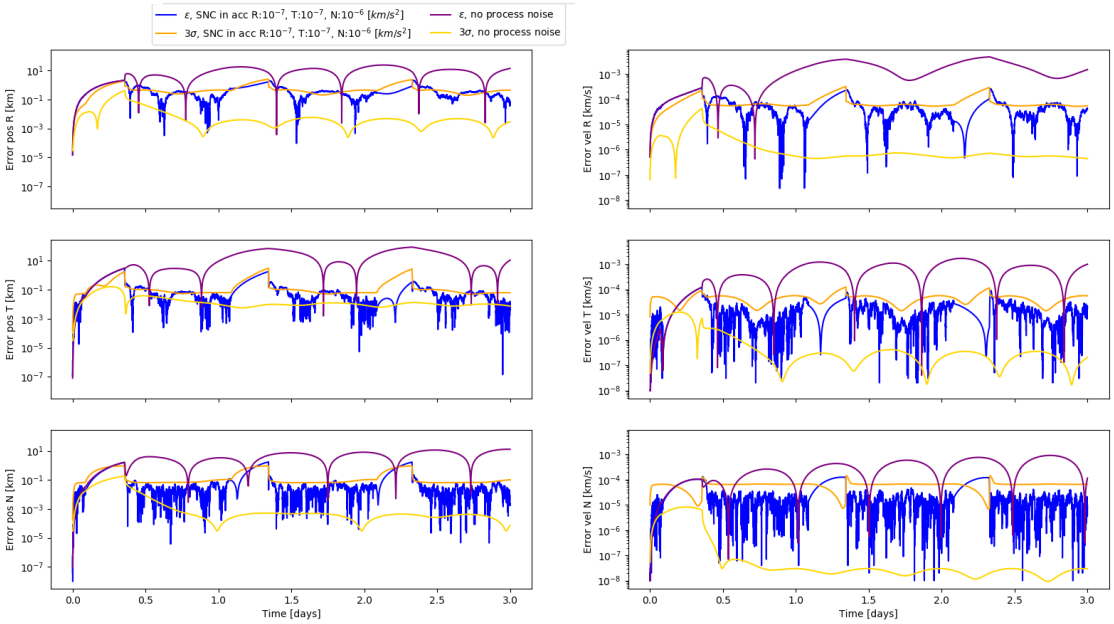

Figure 4.13. GEO example: no process noise vs proper choice of the tuning parameters.

But what happens if we choose too high values of the tuning parameters? In this case we are overestimating the perturbations, so we are "inflating" the covariance too much and giving too much importance to the new measurements and too small importance to the previous estimate. The consequence is a covariance that overestimates the real error, and a real error that is bigger than for the previous case (see Figure 4.14).

A similar analysis can be done for the LEO example. Without taking into account the process noise (Figure 4.15) the covariance is underestimating the real error.

The choice of the best values for the tuning parameters has been done experimentally, considering that the drag perturbation is mainly in the direction of the velocity, so on the orbital plane (the effect in the normal direction is much smaller). For this reason, the tuning parameter in the radial direction has been chosen one order of magnitude smaller than for the other directions (Figure 4.16).

Even in this case, choosing too large tuning parameters the covariance overestimates the real error, and also the real error has increased (Figure 4.17).

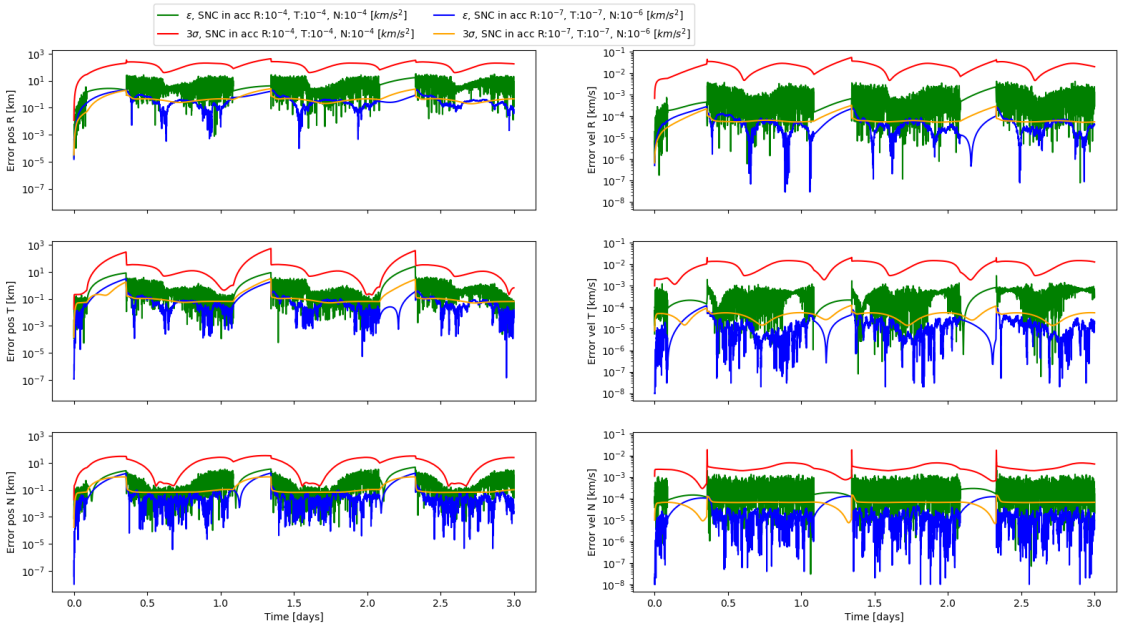

Figure 4.14. GEO example: proper choice vs bad choice of the tuning parameters.

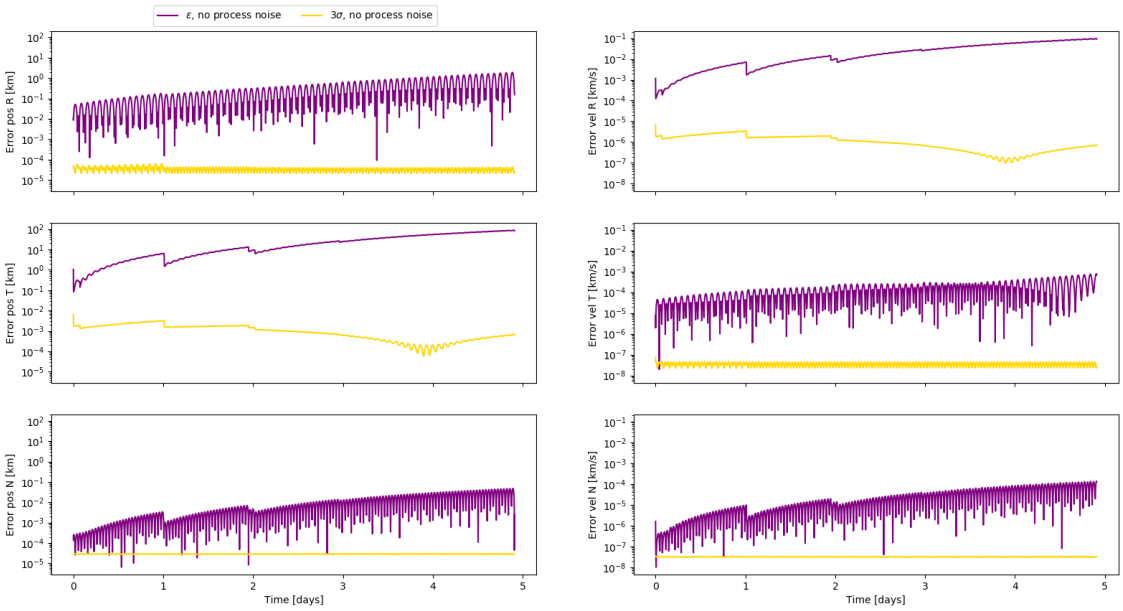

Figure 4.15. LEO example: no process noise.

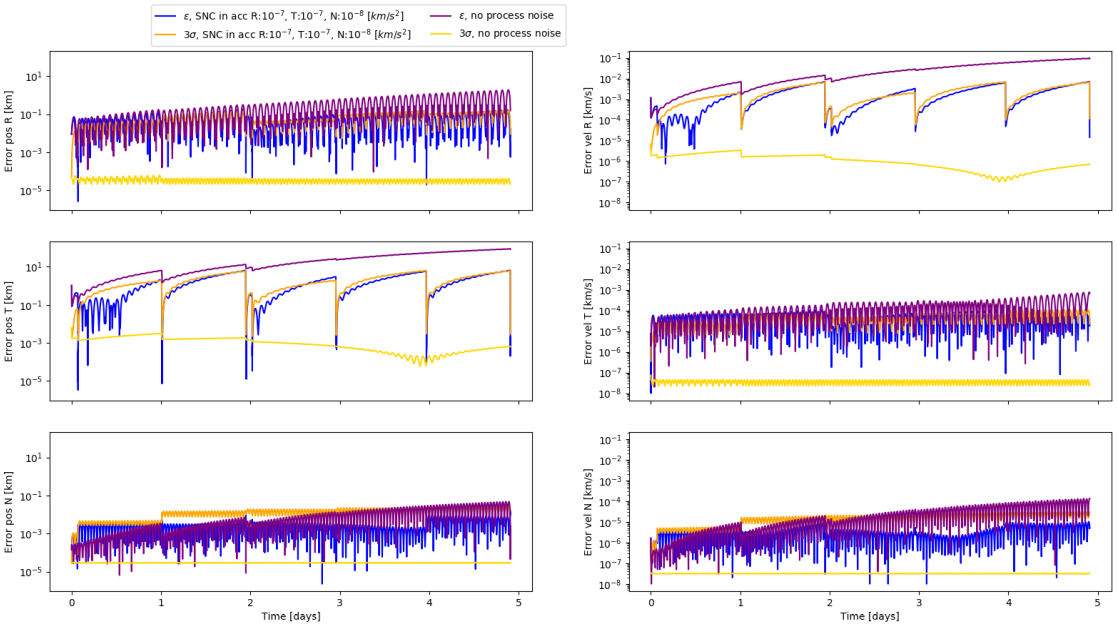

Figure 4.16. LEO example: no process noise vs proper choice of the tuning parameters.

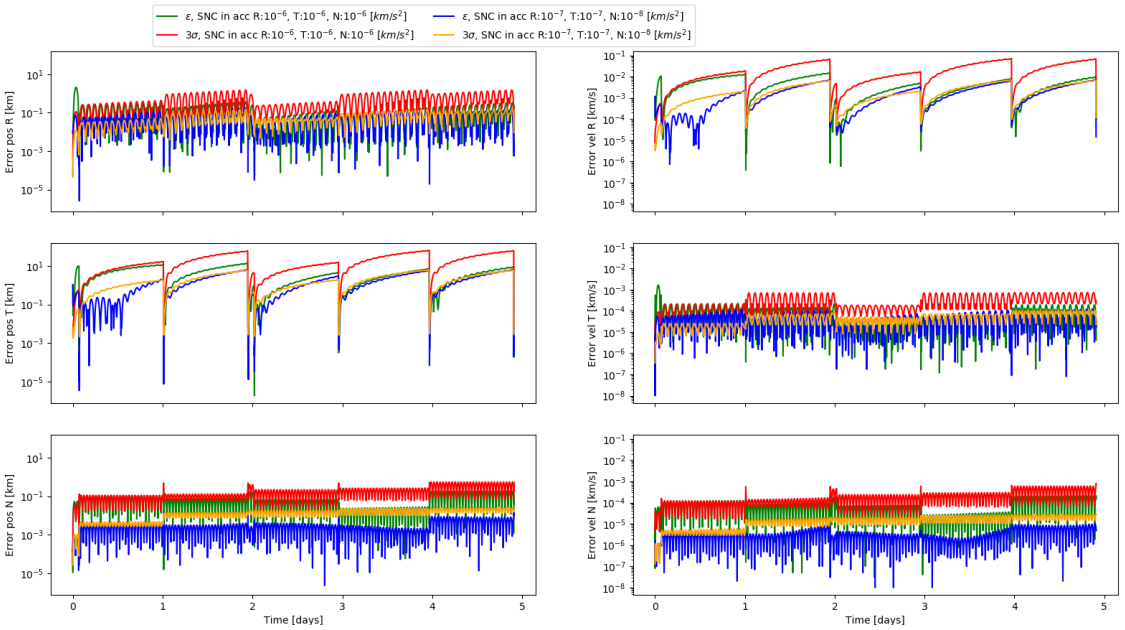

Figure 4.17. LEO example: proper choice vs bad choice of the tuning parameters.

Chapter 5

Conclusions

The three main goals of the Thesis have been accomplished:

- An improved shadow function to compute the lighting ratio has been tested with *Orekit* and implemented in *SPOOK*, considering not only the Earth but also other bodies (for example the Moon) as occulting body.
- A tracklet building algorithm has been written from scratch and tested with several combinations of observables and several kinds of orbit, both with simulated and real data.
- A State Noise Compensation algorithm to take into account the process noise has been implemented and tested for two typical orbits (a GEO and a LEO), choosing the proper values of the tuning parameters. A Dynamical Model Compensation has been implemented but not fully tested.

Obviously the models that have been implemented are not perfect, so it is crucial to highlight what are the limitations in these models showing suggestions for future improvements.

The model used for the shadow function has these limitations:

- The Sun and the Earth are assumed as perfect spheres. For the Sun this approximation is extremely good, while for the Earth is quite good (the polar radius is more or less 0.3% smaller than equatorial ratio). Taking into account the non-sphericity of the Earth, the Earth visual cone projected on the unit sphere would an ellipse and not a circle. While with circles we can assume a central symmetry, with ellipse this symmetry is lost. This would make our model dependent on the reference frame, while the model presented in this Thesis is independent on the reference frame (as long as all the quantities are expressed in the same reference frame).
- We are considering intersections of circles on a plane instead of intersections of solid angles. Since the distances are very big compared with the dimensions of the bodies, this approximation is very good. However, a more precise approach using intersections of solid angles is possible (see [9]).

- The refraction due to the Earth atmosphere is ignored.
- The albedo, i.e. the reflection of the solar rays due to "reflecting" surfaces on the Earth, like the snow or the clouds, is ignored.

Besides the limitations on the shadow function itself, there are other limitations which affect the final value of the perturbing acceleration due to the solar pressure.

- The solar radiation pressure is not perfectly isotropic and depends on the solar activity (the value of the solar constant G_{SC} is not really constant, but it varies within a certain range).
- The space object can change its attitude and in this way, unless we have some symmetries, the value of the cross-sectional area $A_{\perp}$ and the coefficient of reflectivity C_{R} can vary.
- We assume that the solar rays reach instantly the object, otherwise we should take into account also that the position of the Sun relative to the object at the time when the solar rays are originated and at the time when they reach the object is different.

As regards the tracklet building algorithm, to find an algorithm that works perfectly in all the cases is very complicated because many factors can contribute (for example if the number of fake detections is much higher than the number of real detections). Moreover, it is important to find a balance between the correctness of the solution and the execution time. It is quite useless an algorithm that finds a perfect solution in one million years. In the algorithm a fast mode has been implemented. If the program runs in fast mode, when the number of possible combinations becomes too big, all the non-established tracklets are deleted. In this way we reduce the execution time at the expense of a solution that could be imperfect. In fact this mode deletes indistinctly good and bad non-established tracklets, so it is quite risky. This mode is necessary especially if the algorithm is run in real time, where the time between one picture and the following could be 1-10 s, so the execution time per each time block shouldn't be longer than that time, otherwise the pictures and the code execution are not synchronized.

Another limitation is related to the maximum number of missing points. If we increase too much this number, the number of possible combinations can become very large, while if we reduce too much this number we risk to divide a tracklet into subtracklets.

Another limitation is due to the error in the measurements (see subsection 3.2.8).

The algorithm could be improved in various ways:

- We could analyse the brightness of the points for optical sensors, or the RCS (Radar Cross-Section) for radars (if present). This information could be very useful but it could "confuse" the algorithm in some cases. In fact, we expect that this quantity (either the brighness or the RCS) of an object remains almost constant and so it is an extra information that we can use to distinguish two tracklets, but that assumption is not always true. If the object is very irregular and spins at very high angular velocity around its axis, it reflects the light in an unpredictable way and changes the cross-section area, so this quantity is not constant at all. So this information should be used with caution. If this quantity is almost constant, the confidence level of that

tracklet gets higher, but if it is not we can say nothing about the correctness of the solution (we can base only on the other observables).

- If we don't care about the priority of the various regions, we could parallelize the code and divide the execution of each regions into different threads that run simultaneously.
- We could use different tolerances for different kinds of observables, and make them dependent on the errors in the measurements (for the typical values of the errors in the modern telescopes this is not necessary, because the default accepted tolerances are much higher than the typical error in the measurements, see subsection 3.2.8).
- We could use a weighted least square method instead of a classical one, giving different weights to the various points of a tracklet. It is not so obvious the choice of the weights to use in the regression, so a classical least square method has been used.

As regards the process noise algorithm, the main limitation is the choice of the "best" tuning parameters, that at the moment is not automatised (depending on the initial conditions we compute experimentally proper tuning parameters). Moreover the State Noise Compensation assumes white noise with zero mean in the three directions, an approximation that is not completely right.

Some improvements can be made:

- The automation in the choice of the tuning parameters.
- The test of first-order Gauss-Markov process both for unmodelled accelerations and for the ballistic coefficient and the solar radiation pressure coefficient.
- The implementation of higher-order Gauss-Markov processes (for example second order).

Bibliography

[1] European Space Agency. *Distribution of debris*. Last access: 09 March 2019. 2013. URL: https://www.esa.int/spaceinimages/Images/2013/04/Distribution_of_debris.

[2] European Space Agency. *How many space debris objects are currently in orbit?* Last update: 25 July 2013, Last access: 09 March 2019. URL: https://www.esa.int/Our_Activities/Space_Engineering_Technology/Clean_Space/How_many_space_debris_objects_are_currently_in_orbit.

[3] European Space Agency. *Hypervelocity impacts and protecting spacecraft*. Last update: 1 November 2018, Last access: 09 March 2019. URL: https://www.esa.int/Our_Activities/Operations/Space_Safety_Security/Space_Debris/Hypervelocity_impacts_and_protecting_spacecraft.

[4] CelesTrack. *Historical SATCAT Growth, 1957 to Present*. Last access: 09 March 2019. URL: https://celestrak.com/satcat/growth.png.

[5] CelesTrack. *SATCAT Boxscore*. Last access: February 2019. URL: https://celestrak.com/satcat/boxscore.php.

[6] Shu-Ho Dai and Ming-O Wang. *Reliability analysis in engineering applications*. Van Nostrand Reinhold, 1992.

[7] Bruce P Gibbs. *Advanced Kalman filtering, least-squares and modeling: a practical handbook*. John Wiley & Sons, 2011.

[8] Huashan Li et al. "Solar constant values for estimating solar radiation". In: *Energy* 36 (Mar. 2011), pp. 1785–1789. DOI: 10.1016/j.energy.2010.12.050.

[9] Oleg Mazonka. "Solid angle of conical surfaces, polyhedral cones, and intersecting spherical caps". In: *arXiv preprint arXiv:1205.1396* (2012).

[10] Nikolaos K Pavlis et al. "An earth gravitational model to degree 2160: EGM2008". In: *EGU General Assembly* (2008).

[11] Julian Rodríguez-Villamizar. "Dynamical Compensation Methods in Uncertain Systems Applied to Orbit Determination". MA thesis. Technical University of Munich, 2017.

[12] Carlos M. Roithmayr. "Contributions of Spherical Harmonics to Magnetic and Gravitational Fields". In: (2004).

[13] Bob Schutz, Byron Tapley, and George H Born. *Statistical orbit determination*. Elsevier, 2004.

[14] R. W. Sinnott. "Virtues of the Haversine". In: *Sky and Telescope* 68 (1984), p. 159.

[15] George P Sutton and Oscar Biblarz. "Rocket propulsion elements". In: (2001).

[16] David A Vallado. *Fundamentals of astrodynamics and applications*. Vol. 12. Springer Science & Business Media, 2001.