Line of Sight Extraction Algorithm for Stand-Alone CubeSats in Deep-Space

Optical Image Generation, Attitude Determination and Identification of Planets at Unresolved Scale

POLITECNICO
MILANO 1863

Department of Aerospace Science and Technology

Author

Salvatore Andrea Bella
Space Engineering MSc

Supervisor

Francesco Topputo
Politecnico di Milano

Co-supervisor

Vittorio Franzese
Politecnico di Milano

Wednesday 28th April, 2021

Sommario

Affinchè i CubeSat possano navigare autonomamente nello spazio profondo, è necessario che essi siano dotati di strumenti in grado di calcolarne lo stato. Gli algoritmi finora sviluppati sono in grado di determinare l'assetto di un satellite senza la necessità di un intervento umano, anche a partire da condizioni *lost-in-space*. Tuttavia, i metodi attualmente disponibili per determinarne la posizione dipendono da operazioni effettuate dal *ground-segment*, non permettendo ai CubeSat di soddisfare i requisiti per essere definiti *stand-alone*.

Questa tesi propone un metodo, basato sulla conoscenza della direzione dei pianeti rispetto ad un satellite, per calcolare la posizione del CubeSat. Essendo nota la posizione di ogni pianeta del sistema solare in un dato istante, la posizione di un satellite può essere determinata a patto che sia possibile ricavare la *line-of-sight* di un pianeta sfruttando gli strumenti a bordo.

L'argomento principale di questo scritto è lo sviluppo di un algoritmo per l'estrazione della *line-of-sight* basato su dispositivi ottici, come gli *star tracker*, capace di ricavare in modo autonomo la direzione di un pianeta partendo da un'opportuna stima iniziale della posizione del satellite.

Inoltre, è stato messo a punto il modello di uno *star tracker* e l'immagine generata in questo modo è stata processata attraverso un algoritmo di *centroiding*. In seguito, l'assetto è stato ricostruito grazie all'implementazione di un metodo di determinazione basato su un processo di identificazione delle stelle opportunamente strutturato. Il codice è stato testato su un database distribuito omogeneamente in modo da ottenere risultati rappresentativi che siano coerenti con quanto riportato in letteratura.

I risultati di questa tesi mettono le basi per lo sviluppo di uno strumento che possa essere utilizzato per implementare la navigazione autonoma su uno *stand-alone* CubeSat che viaggia nello spazio profondo.

Abstract

Deep-Space autonomous navigation for stand-alone CubeSats requires the tools to evaluate the state of the spacecraft. State of the art algorithms are able to determine the attitude of a spacecraft without the need for human intervention, even when starting from lost-in-space conditions. However, the position determination methods that are currently available, strongly rely on ground-segment operations thus not allowing the CubeSat to meet the requirements for the stand-alone condition.

This thesis proposes a method, based on the direction of planets with respect to a spacecraft, to evaluate the location of the CubeSat. Being the position of every planet in the solar system with respect to the Sun known at any given time, the location of a spacecraft can be computed if it is possible to retrieve the line of sight of a planet exploiting the on-board hardware.

The main topic of this work is the development of a line of sight extraction algorithm based on optical devices, such as star trackers, able to autonomously retrieve a planet direction starting from a compliant initial guess of the spacecraft position.

Moreover, a star tracker is modeled and the image generated in this way is processed exploiting a proposed centroiding algorithm. Subsequently, the attitude is reconstructed thanks to the implementation of a determination method based on a coherently built star identification process. The code is tested on an homogeneously distributed database allowing the obtainment of representative results that are coherent with the state of the art literature.

The outcome of this work lays the foundation for the development of a tool that can be exploited to implement autonomous navigation on a stand-alone CubeSat traveling in deep-space.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Since the late 1990s, the interest in space related services shown during the first space age has undergone a paradigm shift towards commercial services leading to a diversification and growth in satellites demand [1]. However, the required budget for the production of such technologies, in terms of both cost and time, was often not consistent with the market request [2].

In order to adapt to this situation, universities have researched on the development of smaller modular satellites (CubeSats) for which the production process could be standardized and the components always available off-the-shelf (COTS), thus reducing both the production time and cost [3].

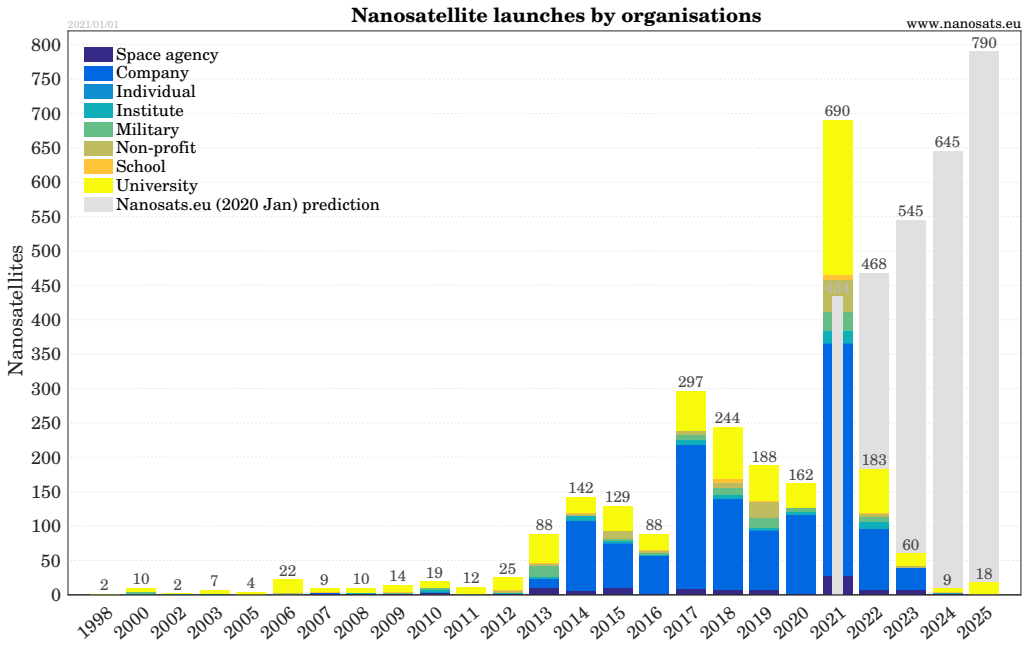


Figure 1.1 History of nanosatellites launches by organizations.[I]

The work carried on by universities during the first decade of the XXI century has successfully tackled the problem, getting the attention of different kinds of organisations among which military ones and private companies as reported in Fig. 1.1. Indeed, the data in Fig. 1.2 show that, although the universities continue their studies in such a growing field, the developed technologies have become more and more appealing for companies.

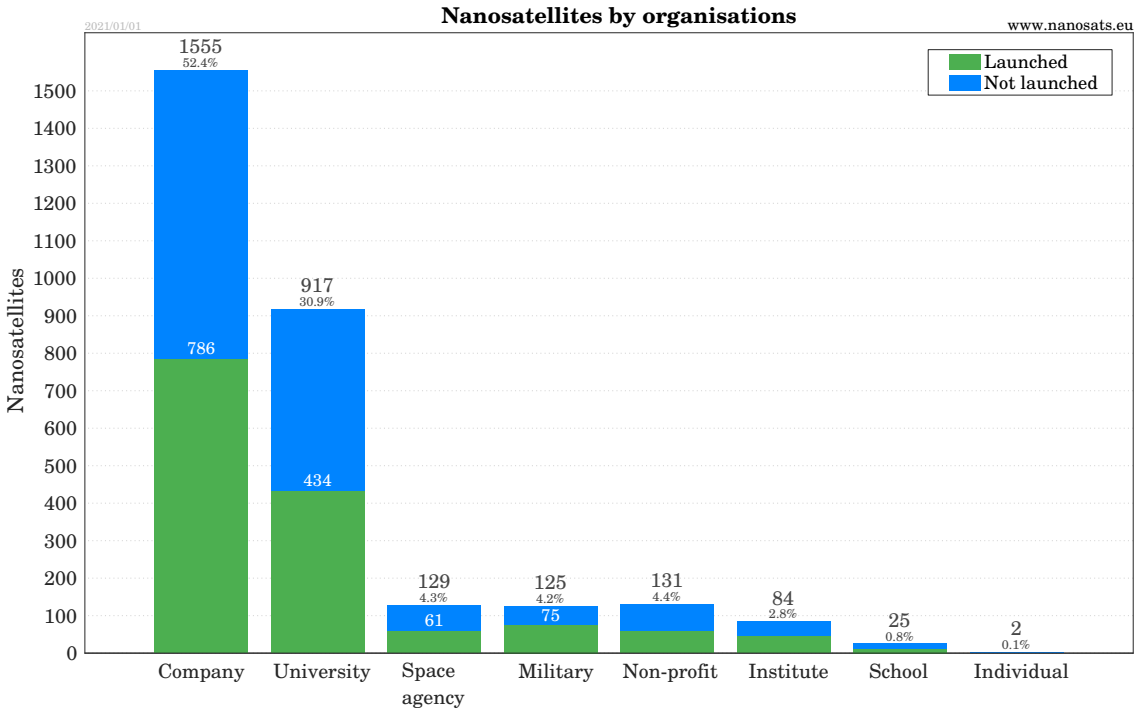


Figure 1.2 Total nanosatellites by organizations.[II]

During the last five years the CubeSat production has skyrocketed as showed in Fig. 1.3. In most cases [III] this class of spacecrafts is currently intended for use in low earth orbit (LEO), but studies are carried on to use them in deep space missions [IV] [V] [VI]. Moreover, the technological development associated with the expansion to such fields of application could also lead to a further reduction in the production budget in terms of both cost and time.

[I] *https://www.nanosats.eu/#figures*
[II] *https://www.nanosats.eu/#figures*
[III] *https://www.nanosats.eu/database*
[IV] *https://www.space.com/29306-cubesats-deep-space-exploration.html*
[V] *https://www.nytimes.com/2019/03/18/science/cubesats-marco-mars.html*
[VI] *https://www.esa.int/Safety_Security/Hera/CubeSats_joining_Hera_mission_to_asteroid_system*

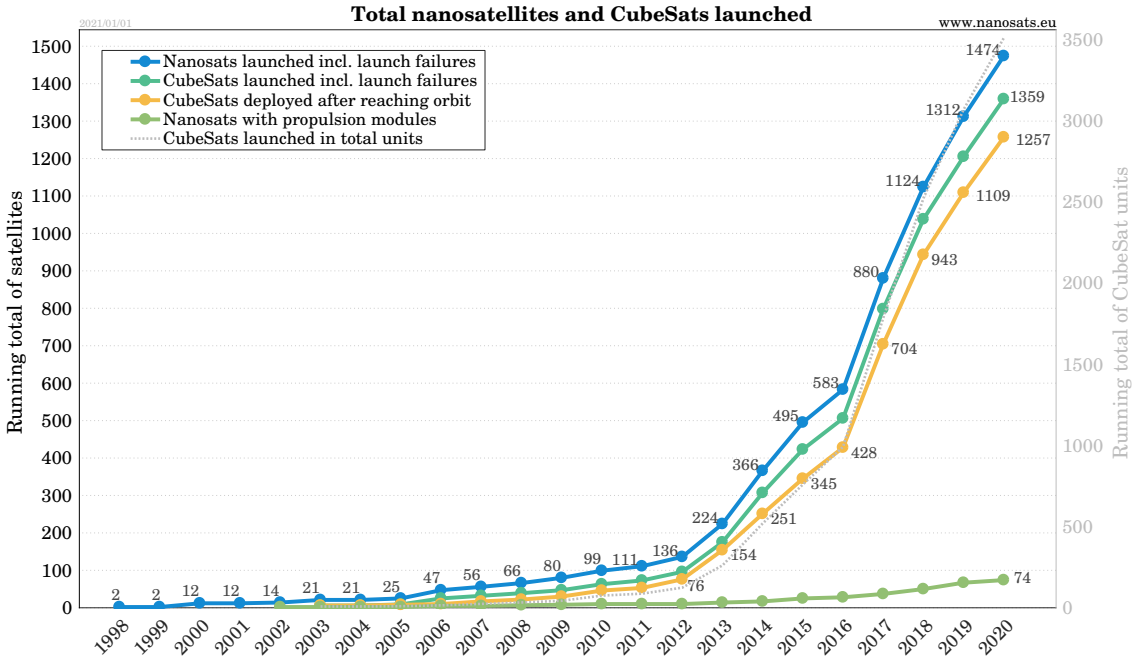


Figure 1.3 Total nanosatellites and cubesat launched.[VII]

It is worth to consider that, as with every new technology, the miniaturization trend will eventually reach a point in which the benefits provided will settle. Due to this premise, it is useful to notice that the overall budget of a space mission is mainly due to the production contribution, dependent on the miniaturization process, and the mission management which can be carried on by ground-based facilities or on-orbit (for manned missions). Therefore, while the CubeSat solution is focused on the production front, the mission management allocated budget remains almost untouched from the miniaturization process. For this reason, such technology is more and more often coupled with automatic on-board systems that are able to accomplish tasks on their own without the need of human interactions.

This family of solutions is already partially applied to actual missions and, in the next future, the aim is to achieve completely autonomous miniaturized satellites that are capable to cut the budget of a mission not only from a production point of view but also from a management and launch prospective.

[VII] *https://www.nanosats.eu/#figures*

1.2 Problem statement

The paradigm upon which a fully autonomous CubeSat is based is the complete and reliable automation of the guidance, navigation and control process (GNC). Each step towards this goal is achieved thanks to the improvements in the following fields:

- *Guidance*: determination of the optimal trajectory and the variations in the state required to follow such path;
- *Navigation*: determination of the state vector (i.e. linear and angular position and velocity) at a given time and consequently of the orbit;
- *Control*: handling of actuators in order to apply guidance instructions.

Moreover, when it comes to a class of satellites as the one described in this context, it is necessary to keep in mind that using a reduced number of sensors and actuators is essential to improve the miniaturization process. Indeed, the main difficulties in reaching a complete and reliable system are strictly related to the number of available sensors and actuators and to the conditions in which they can be used without failures.

Considering the navigation goals, it is useful to investigate the problem of linear and angular states separately. This is due to the fact that the computation of the angular state has been well studied during the last years and different solutions are available to tackle such problem. On the other hand, the autonomous linear position determination is a field of study which is still under investigation; indeed, in most cases the position determination is carried on through ground-based equipment and radiometric tracking techniques such as range and Doppler methods [4].

1.3 Thesis objective

Autonomous position determination can be achieved through the use of an optical device, as a star tracker or a camera, by exploiting visible planets [5]. Considering the case reported in Fig. 1.4 it is possible to notice that:

$$\begin{cases} r = R_1 - \|\rho_1\| \hat{\rho}_1 = R_2 - \|\rho_2\| \hat{\rho}_2 \\ \text{where } \rho = \|\rho\| \hat{\rho} \end{cases} \tag{1.1}$$

Also, multiplying through a scalar product Eq. (1.1) with $\hat{\rho}_1$ and $\hat{\rho}_2$:

$$\begin{cases} \hat{\rho}_1^\intercal R_1 - \|\rho_1\| = \hat{\rho}_1^\intercal R_2 - (\hat{\rho}_1^\intercal \hat{\rho}_2^\intercal) \|\rho_2\| \\ \hat{\rho}_2^\intercal R_2 - \|\rho_2\| = \hat{\rho}_2^\intercal R_1 - (\hat{\rho}_2^\intercal \hat{\rho}_1^\intercal) \|\rho_1\| \end{cases} \tag{1.2}$$

This system can be written in matrix form as:

$$\begin{bmatrix} -1 & \hat{\rho}_1^\intercal \hat{\rho}_2^\intercal \\ -\hat{\rho}_2^\intercal \hat{\rho}_1^\intercal & 1 \end{bmatrix} \begin{bmatrix} \|\rho_1\| \\ \|\rho_2\| \end{bmatrix} = \begin{bmatrix} \hat{\rho}_1^\intercal (R_2 - R_1) \\ \hat{\rho}_2^\intercal (R_2 - R_1) \end{bmatrix} \tag{1.3}$$

This linear system can be solved in order to retrieve $\|\rho_1\|$ and $\|\rho_2\|$ if the other variables are known. In this way, it is possible to determine the position of a spacecraft autonomously.

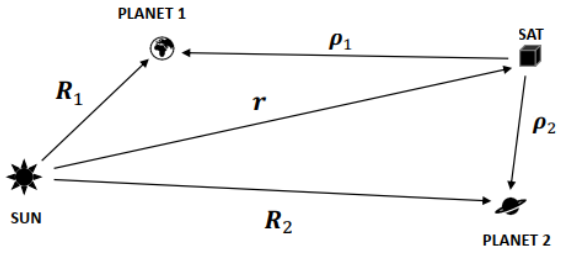


Figure 1.4 Planets position vectors. [6]

Considering that the planets position with respect to the Sun is available from the ephemerides, it is useful to notice how the problem has shifted to the computation of the planets directions relatively to the spacecraft, $\hat{\rho}_1$ and $\hat{\rho}_2$. Although several techniques are available in literature [7] [8] [9] to compute a planet direction when it is well visible on the picture taken, this condition is not easily achievable in deep space since such major celestial bodies are unresolved and not easily distinguishable.

The main goal of this thesis is to develop an algorithm capable of retrieving the direction of a planet with respect to the spacecraft, when the major body is at unresolved scale, through the use of a star tracker. Furthermore, the angular position (i.e. attitude) can also be determined using the same sensor.

1.4 Thesis outline

The topics addressed in this chapter aim at clarifying the context in which the thesis is placed and its contribution to the field. In order to achieve the previously defined goals, a test framework is defined. Indeed, in Chapter 2 a procedure to model a star tracker is identified, this is required due to the lack of actual images upon which a test can be carried on. Moreover, it allows to identify the hardware conditions on which the presented software is tested and, therefore, works.

The actual on-board software is initially defined in Chapter 3: an algorithm to identify the position of the celestial objects on an image, the so called centroid, is proposed. The aforementioned algorithm is used to retrieve the position of the stars on an image which are used as an input for a features extraction and matching software which goal is to identify the framed stars. The attitude matrix is computed starting from this data.

The knowledge achieved up to this point is exploited in Chapter 4 to develop a strategy for the line of sight (LoS) extraction process of a previously defined major body. Once the theoretical assumptions of the mentioned chapters are fixed, they are tested and the results reported in Chapter 5.

In conclusion, such results are analysed and commented in Chapter 6 together with some suggestions to improve this work in the future. The logical link between the chapters is reported in Fig. 1.5.

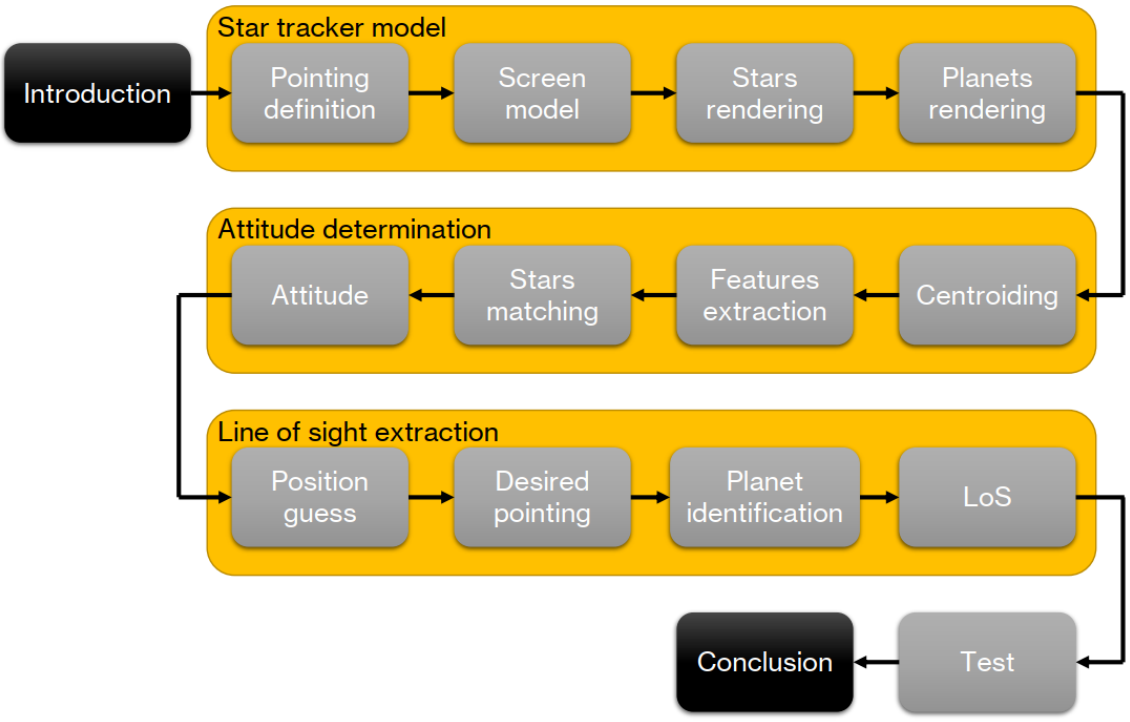


Figure 1.5 Thesis logical flowchart.

Chapter 2

Star tracker model

2.1 Motivation and purpose

In order to achieve the goals stated in Section 1.3, it is necessary to build a framework in which such objectives can be monitored and each step can be well noticed and documented. The main requirement, dealing with algorithms that are based on image processing techniques, is being able to produce a reliable image under any condition requested from the algorithms demand.

Moreover, taking into account the background introduced in Section 1.1, it is possible to define a preliminary list of functional requirements (Table 2.1) that the generated images must satisfy to be defined reliable.

Table 2.1 Star tracker model functional requirements.

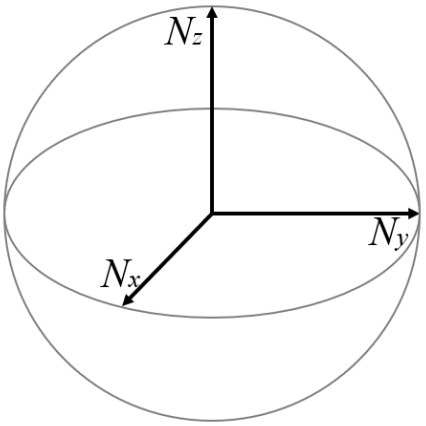
ID	Requirement
R-F-001	Provide an output for any camera pointing direction
R-F-002	Provide an output for any camera twist angle
R-F-003	Take into account the camera technical specifications
R-F-004	Represent planets when they are unresolved
R-F-005	Export in a format coherent with the processing algorithms

Once such image is produced, it can serve as an input to the subsequent steps as explained in Section 1.4. Moreover, some additional information may be extracted during the image generation process with test purposes (e.g. true celestial objects position, noiseless image, coordinates in different reference frames, etc.).

2.2 Reference frames

The coordinates of a celestial object, such as stars and planets, are often expressed according to an inertial reference frame (i.e. celestial sphere, **N**) and collected in catalogs [10]. However, a star tracker is able to deal only with a small portion of this sphere according to its technical specifications and its pointing direction. Therefore, before starting with the production of a representative image, it is necessary to identify the involved reference frames and the global rotation matrix ($\mathbf{R}_{\mathrm{BN}}$), which allows to express the coordinates of the celestial objects according to the camera reference frame (**B**).

The following considerations are made under the assumption that the axes of the inertial frame are centered on the spacecraft and defined according to the celestial sphere reference frame [11] as depicted in Fig. 2.1.



$$\begin{cases} \mathbf{N}_{\mathrm{x}} : \text{vernal equinox;} \\ \mathbf{N}_{\mathrm{z}} : \text{celestial north pole;} \\ \mathbf{N}_{\mathrm{y}} : \mathbf{N}_z \times \mathbf{N}_{\mathrm{x}}. \end{cases}$$

Figure 2.1 Celestial sphere.

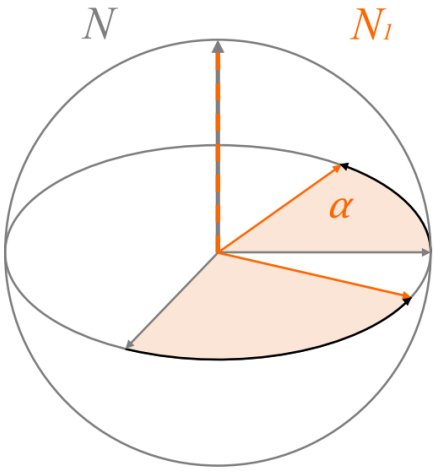
In order to achieve the desired global rotation matrix, a series of counterclockwise rotations is applied to the inertial reference frame taking into account the camera pointing angles, defined as: right ascension (α), declination (δ) and twist angle (ϕ). Also, the domains in which these angles are respectively limited to: [0°, 360°], [-90°, 90°] and [0°, 360°].

Moreover, the camera pointing axis is considered to be coincident with its reference frame third axis. The mathematical formulation of the problem is expressed in Eq. (2.1) and the visual steps to achieve it are illustrated in Fig. 2.2, Fig. 2.3 and Fig. 2.4.

$$\mathbf{R}_{BN} = \mathbf{R}_3(\alpha)\mathbf{R}_2(\frac{\pi}{2} - \delta)\mathbf{R}_3(\phi) \tag{2.1}$$

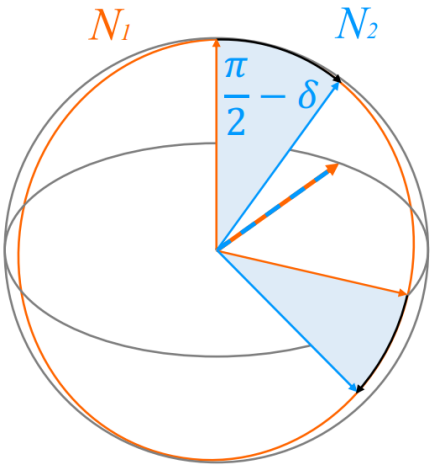
Finally, as previously discussed, it is possible to compute the coordinates of a point in the camera reference frame:

$$\mathbf{P}_B = \mathbf{R}_{BN}^{\intercal}\mathbf{P}_N\,. \tag{2.2}$$



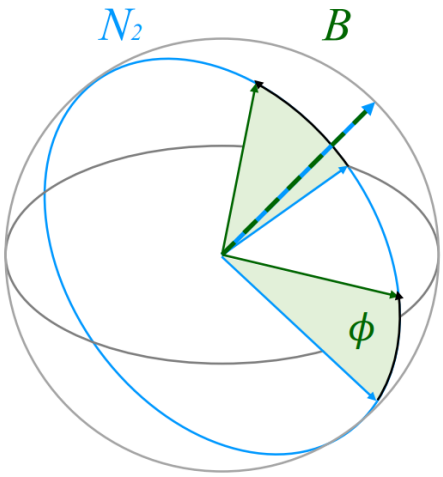
$$\mathbf{R}_3(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 2.2 Rotation from **N** to $\mathbf{N_1}$.



$$\mathbf{R}_2(\tfrac{\pi}{2}-\delta) = \begin{bmatrix} \cos(\frac{\pi}{2}-\delta) & 0 & \sin(\frac{\pi}{2}-\delta) \\ 0 & 1 & 0 \\ -\sin(\frac{\pi}{2}-\delta) & 0 & \cos(\frac{\pi}{2}-\delta) \end{bmatrix}$$

Figure 2.3 Rotation from $\mathbf{N_1}$ to $\mathbf{N_2}$.



$$\mathbf{R}_3(\phi) = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 2.4 Rotation from $\mathbf{N_2}$ to **B**.

2.3 Optical camera

Although the position of a celestial object in the sensor reference frame is retrieved as in Eq. (2.2), further details are required to define how such object is visualized on the star tracker. In order to obtain this information, it is necessary to compute the point coordinates on the screen. Considering that the device working principle [12] can be basically summarized as:

- collection of the incoming light through an optical sensor;
- transfer of the collected light to a reading sensor;
- elaboration of the analog information;
- extraction of the digital data.

The major concerns in the modeling of this phenomenon are related to the correct approximation of the different kind of errors and to the geometrical transformation through which the points are projected from the three-dimensional world to a flat surface.

2.3.1 Pinhole camera model

The pinhole camera (Fig. 2.5) is a simplified model [13] which comes in use for the purpose of this thesis. It can be used to simulate the projection of the points from the camera reference frame to the image plane. It consists in approximating a lens as a point that projects the incoming light onto the image plane. The main physical parameters of this model are:

- focal length (f): distance between the image plane and the pinhole;
- field of view (FoV): angle that identifies the three-dimensional world fraction to be projected on the image plane;
- screen size (L): half-length of a square shaped screen.

These parameters are correlated, depending on the problem geometry, according to Eq. (2.3) and their meaning is illustrated in Fig. 2.5.

$$FoV = 2\,\theta = 2 \arctan\left(\frac{L}{f}\right). \tag{2.3}$$

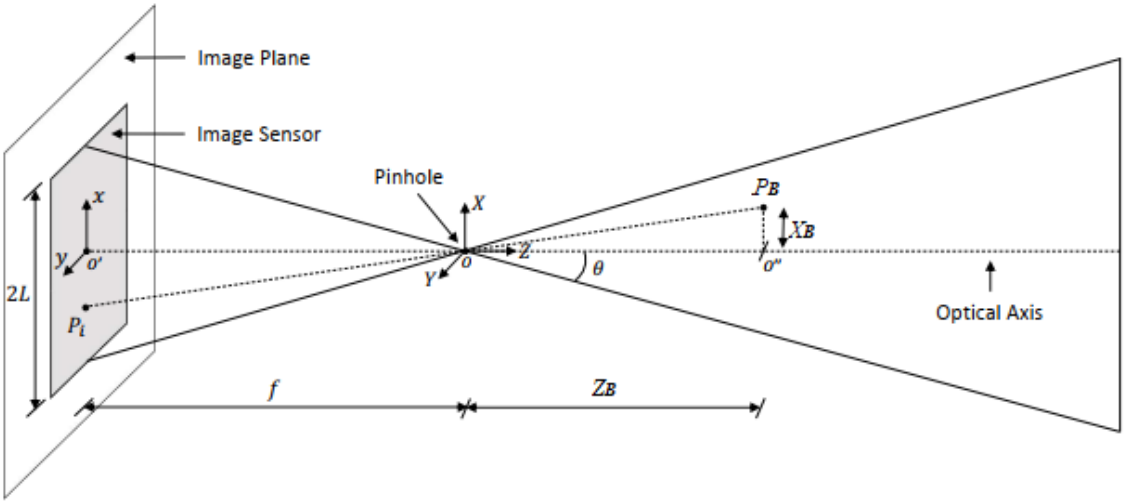


Figure 2.5 Pinhole camera model. [6]

Considering a point $P_B = [X_B, Y_B, Z_B]^\intercal$ and its projection onto the image plane according to this model $P_i = [x_i, y_i]^\intercal$, it is possible to retrieve the mathematical correlation through geometrical considerations. Indeed, exploiting the similarity between the triangles OP_BO" and OP_iO', it is possible to compute the relations that govern this transformation:

$$\begin{cases} x_i = -f\frac{X_B}{Z_B}; \\ y_i = -f\frac{Y_B}{Z_B}. \end{cases} \tag{2.4}$$

Even if the transformation is straightforward, it is useful to notice how, due to the intrinsic nature of the problem, the images generated through this projection are flipped. Therefore, in order to visualize the correct image, as it would be displayed on the screen of an actual star tracker, this phenomenon has to be taken into account when the image is processed to be displayed as on a screen.

2.3.2 Screen model

Up to this point, lengths have been defined according to the International System of Units standards (i.e. meters and its sub-multiples). However, when dealing with screens, using pixel units is more suitable since they are related to the nature of the device itself. According the model used for the purpose of this work, a screen is considered as a $N_{px} \times N_{px}$ set of dots (i.e. pixels). The screen dimension is related to the pixels size as:

$$2L = L_{px} N_{px}. \tag{2.5}$$

This is true under the assumption that each pixel is squared and has the same characteristic dimension (L_{px}).

Finally, the relation to obtain the coordinates in the screen reference frame is expressed in Eq. (2.6). The minus sign is due to the correction for the image being flipped from the previous transformation. Moreover s_x and s_y are the scaling factors for the conversion from metric system to pixel units, while $[o_x, o_y]^\intercal$ is the translation vector between the image and pixel reference frames expressed in pixels.

It is useful to note that the pixel reference frame origin coordinates ([0 0]) is often placed on the image top-left corner, with the positive y axis pointing down and the positive x axis right.

$$\begin{cases} x_p = -s_x x_i + o_x\,; \\ y_p = -s_y y_i + o_y\,, \end{cases} \quad \begin{cases} s_x = s_y = \frac{N_{px}}{2L}\,; \\ o_x = o_y = \frac{N_{px}}{2}\,. \end{cases} \tag{2.6}$$

2.3.3 Light-gathering capability

In order to understand the selection process of a proper optical device, it is useful to introduce the light-gathering capability parameter[14]. It takes into account the lens focal length and its aperture diameter (d) allowing to understand how much light an instrument can inherently collect. Such parameter is called the F-number and defined as:

$$F = \frac{f}{d}\,. \tag{2.7}$$

It is useful to highlight that the actual amount of collected light is also influenced by the amount of time during which the lens is exposed to the environment. Such time is known as exposure time (T). When it comes to the notation, considering a lens with an F-number of 2, such characteristic is expressed as "$f/2$".

In summary, considering two devices with different F-numbers that collect light for the same amount of time, the one with the smallest F-number collects more light. To clarify the meaning of this consideration, Fig. 2.6 illustrates the behaviour of the F-number with respect to the lens aperture for a fixed value of focal length.

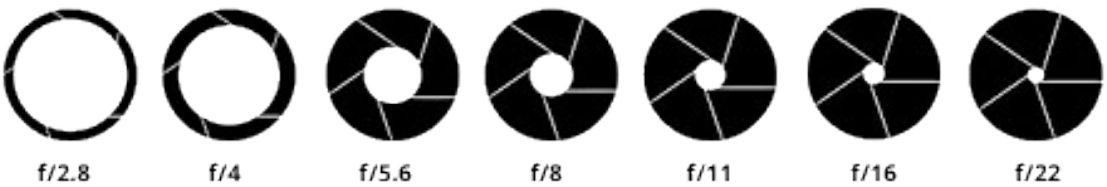


Figure 2.6 Comparison between different f-numbers.[I]

[I] *https://lucagherardi.com/*

2.4 Celestial objects

Up to now, celestial objects have been considered as points with known coordinates that are projected from the inertial frame to the camera reference frame by exploiting the previously described equations, in particular Eq. (2.2), Eq. (2.4) and Eq. (2.6). However, to achieve a realistic representation of such objects it is necessary to define:

- the kind of sources from which the inertial positions are retrieved;
- which celestial objects are represented on the screen;
- how the light emitted from such objects interacts with the screen.

2.4.1 Stars

Concerning stars, their data is often collected in catalogs. Such data can be considered to be constant for a reasonable amount of time and, therefore, independent from the epoch owing to their distance with respect to the Solar System.

For the purpose of this thesis the *Hipparcos* catalog [10] is used to retrieve such data. Among the different kind of information the aforementioned catalog can provide for each star, those required for this work are:

- identifier (ID);
- right ascension (α);
- declination (δ);
- magnitude in the Johnson UBV photometric system (V).

Indeed, it is possible to exploit the knowledge of the stars right ascension and declination (Fig. 2.7) to compute their inertial direction on the celestial sphere:

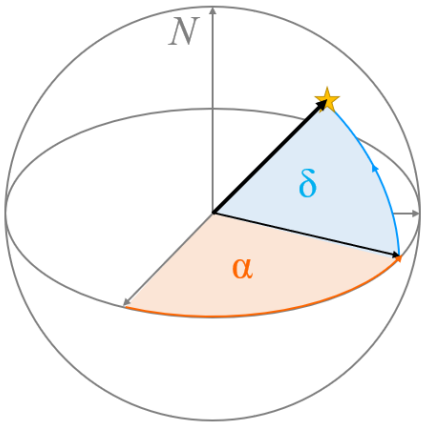


Figure 2.7 Star in the celestial sphere.

$$\begin{cases} X_N = \cos(\delta)\cos(\alpha)\,; \\ Y_N = \cos(\delta)\sin(\alpha)\,; \\ Z_N = \sin(\delta)\,. \end{cases} \tag{2.8}$$

Table 2.2 Planets magnitude parameters [6].

	V(1, 0)	m (β in degrees)
Mercury	-0.36	$3.8(\beta/100) - 2.73(\beta/100)^2 + 2.00(\beta/100)^3$
Venus	-4.29	$0.09(\beta/100) + 2.39(\beta/100)^2 - 0.65(\beta/100)^3$
Earth [II]	-3.86	0.016β
Mars	-1.52	0.016β
Jupiter	-9.25	0.005β
Saturn	-8.90	0.044β
Uranus	-7.19	0.028β

2.4.2 Major bodies

On the other hand, when it comes to bodies inside the Solar System, their position strongly depends on the current epoch. In the context of this project, planets *ephemeris* are used to retrieve their position with respect to the Sun in the **N** reference frame (R) and, knowing the spacecraft position with respect to the Sun (r), the relative position between a planet and the spacecraft (ρ) can be computed as in Eq. (2.9).

To clarify the geometrical reasoning an illustration is reported in Fig. 1.4.

$$\rho_N = R_N - r_N. \tag{2.9}$$

Moreover, it is also required to obtain the magnitude of each planet as already done with the stars. A model [15], based on the intrinsic definition of magnitude, is applied with this purpose:

$$V = V(1,0) + 5\log_{10}(\|\rho\|\,\|R\|) + m, \tag{2.10}$$

where V(1,0) is the absolute magnitude and m is the phase law. Both are tabulated in Table 2.2. In addition, the phase law depends on the phase angle (β) defined as the angle between ρ and R.

2.4.3 Objects selection

As previously mentioned, it is necessary to identify which celestial objects can be represented on the screen. For such purpose, it is necessary to state that only the objects inside the camera FoV and with a proper magnitude can be selected for the image generation

[II] Earth phase law is preliminary supposed to be the same as Mars due to lack of data.

process. The criteria according to which the geometrical selection is performed are:

$$\begin{cases} -\sin\left(\frac{FoV}{2}\right) < X_B < \sin\left(\frac{FoV}{2}\right); \\ -\sin\left(\frac{FoV}{2}\right) < Y_B < \sin\left(\frac{FoV}{2}\right); \\ Z_B > 0\,. \end{cases} \tag{2.11}$$

Furthermore, as stated in Section 1.3, the planets are selected only if they are unresolved objects. Such selection is achieved by defining the object to pixel ratio (OPR) as in Eq. (2.12) and considering admissible only the planets with $OPR \leq 1$. The meaning of each variable is reported in Fig. 2.8.

$$OPR = 2\,\alpha\,\frac{N_{px}}{FoV} = 2\arctan\left(\frac{R_p}{\|\rho\|}\right)\frac{N_{px}}{FoV} \tag{2.12}$$

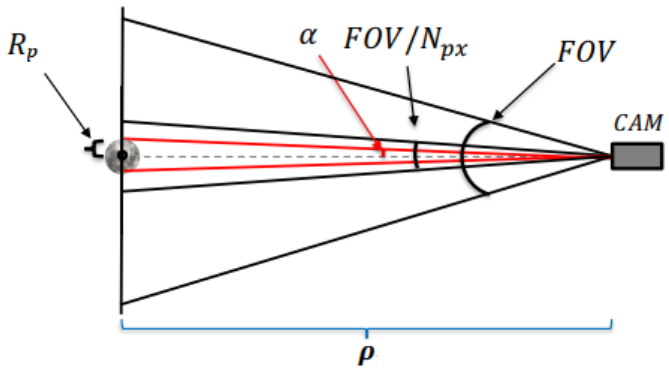


Figure 2.8 Object to Pixel Ratio. [6]

2.4.4 Photoelectrons distribution

Once the position of a celestial object on the screen is identified, it is necessary to evaluate how the light spreads on the screen starting from the object center to its surroundings.

The effect under study is modeled according to three main events, taking into account that each of this events intrinsically involve a loss of signal.

- The photons coming from a celestial object hit the lens with a certain flux;
- a certain amount of these photons interacts with the lens;
- such interaction produces photoelectrons that are collected.

The theoretical approach that supports this model [16] is reported below.

Considering the Planck-Einstein relation, a photon energy (E_γ) can be computed from the speed of light (c), the Planck constant (h) and the wavelength of interest (λ):

$$E_\gamma = \frac{c\,h}{\lambda}\,, \tag{2.13}$$

from this, provided the photon flux density (F_λ) and the bandwidth (BW), the photon flux count (F_γ) can be evaluated:

$$F_\gamma = \frac{F_\lambda\,BW}{E_\gamma}\,. \tag{2.14}$$

Although a more accurate modeling of the numerator of Eq. (2.14) consists in computing the integral of F_λ over the BW range, since F_λ is not constant. However, considering the mentioned approximation, knowing the lens transmission factor ($\mathrm{T_{lens}}$) and the lens aperture (d) it is possible to retrieve the photon count on the sensor ($F_{\gamma/sens}$):

$$F_{\gamma/sens} = F_\gamma\,T_{lens}\,\pi\left(\frac{d}{2}\right)^2, \tag{2.15}$$

and considering the quantum efficiency ($\mathrm{Q_e}$), the electron count on the sensor ($F_{e/sens}$):

$$F_{e/sens} = Q_e\,F_{\gamma/sens}\,. \tag{2.16}$$

Finally, the total number of electrons on the sensor ($N_{e/sens}$) depends on the exposure time (T) as:

$$N_{e/sens} = F_{e/sens}\,T\,. \tag{2.17}$$

In order to avoid the computation of these values for each celestial object, starting from the definition of magnitude (V), it is possible to select a reference object and evaluate the others through similarity:

$$N_{e/sens}(V) = N_{e/sens}(V_{ref}) \times 10^{(V_{ref}-V)/2.5}\,. \tag{2.18}$$

The previous equations aim is to compute the number of photoelectrons on the screen due to the contribution of a specific celestial object. Therefore, the last step consists in clarifying how these photoelectrons are spread over the pixels surrounding the object. For this purpose a normal distribution is considered:

$$N_{e/px}(x,y) = \frac{N_{e/sens}(V)}{\sigma\sqrt{2\pi}}\exp\left(-\left(\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}\right)\right), \tag{2.19}$$

where σ represents the object defocus level and $[\mathrm{x_0},\,\mathrm{y_0}]^\intercal$ are the object center coordinates both expressed in pixel units. Moreover, the maximum number of electrons on a pixel

(Q_{max}) is limited by the sensor specifics and the pixel intensity (I_{px}), which is an integer number between 0 and 255. In first approximation, the pixel intensity can be computed as:

$$I_{px} = 255\, round^{\text{III}}\left(\frac{N_{e/px}}{Q_{max}}\right). \tag{2.20}$$

2.4.5 Noise

The photoelectrons distribution computed in such way, corresponds to an ideal approximation that does not take into account some aspects of the real process. In reality the image is subjected to different sources of noise related to the equipment setup and to different technical parameters.

For this purpose a simplified noise model is built and applied on the ideal image (I_0) before the conversion from photoelectron units to pixel intensity as in Eq. (2.22) and in Eq. (2.23). Each noise contribution [17] has the following meaning according to its source:

- ε_Q: Quantization noise due to A/D conversion;
- ε_R: Readout noise due to A/D conversion;
- ε_{FP}: Fixed Pattern noise due to the inherent tendency of some pixel to be brighter or darker than expected;
- ε_{DS}: Dark Signal noise due to photoelectrons generated even when no photons interact with the device;
- ε_{DSNU}: fixed pattern noise due to Dark Signal Non Uniformity;
- ε_{PRNU}: fixed pattern nosie due to Photo Response Non Uniformity;
- $\varepsilon_{\%}$: margin that takes into account eventual errors during the estimation of the noise.

Therefore, a partial noise is computed as:

$$\varepsilon = \left[\varepsilon_Q + \varepsilon_R + \varepsilon_{FP} + (\varepsilon_{DS} + \varepsilon_{DSNU})\, T\right](1 + \varepsilon_{\%}), \tag{2.21}$$

Subsequently, it is applied to the image as:

$$I_1 = I_0 + \left|\varepsilon\, randn^{\text{IV}}[size^{\text{V}}(I_0)]\right|. \tag{2.22}$$

Finally, this output is used to apply the PRNU contribution as:

$$I_2 = I_1 + \left|\varepsilon_{PRNU}\, mean^{\text{VI}}(I_1)\, randn[size(I_1)]\right|. \tag{2.23}$$

[III] rounds each element of the input to its nearest integer, rounding up the singularities (e.g. $0.5 \rightarrow 1$).
[IV] returns a matrix, with the same dimensions of the input, of normally distributed random numbers.
[V] returns the lengths of the corresponding dimensions of the input.
[VI] returns the mean over all the elements of the input.

Chapter 3

Attitude reconstruction

3.1 Motivation and purpose

In order to achieve a proper attitude determination, the stars coordinates must be known in the inertial reference frame and in the camera reference frame. Starting from an image generated through the process described in Chapter 2, or from any other source, the required coordinates can be computed.

When dealing with the celestial objects on the image, the first step is to retrieve their position expressed in pixel coordinates and defined as centroid (Section 3.2). There is a variety of methods that has been developed through the years and is available in literature [18], with different limitations and tuning requirements according to the specific case. Once such algorithms are applied and the centroids extracted, this information can be used to retrieve the position in both the required frames.

Concerning the camera reference frame information, it can be obtained by knowing the sensor technical specifications and using the equations that represent its underlying functioning processes studied in the previous chapter.

The process is less trivial when it comes to the inertial reference frame information (Section 3.3), this is due to the fact that the image itself is not able to provide the desired data. The most common methods [19] consist in selecting some geometrical or, less frequently, physical parameters related to some stars on the image and extracting such features. The features are then compared to a previously built database and, thanks to this process, matched to the corresponding stars identification code, for which the inertial position can be computed according to a selected matching catalog [10].

Once the coordinates are computed for both frames, an attitude determination method [20] is selected to retrieve the actual attitude as in Section 3.4. The selection of this method depends on different factors, among which the computational capability of the spacecraft and the installed hardware. Finally, a summary of the main functional requirements discussed is reported in Table 3.1.

Table 3.1 Attitude reconstruction functional requirements.

ID	Requirement
R-F-006	Provide celestial objects centroids for any given image
R-F-007	Provide celestial objects position in the camera reference frame
R-F-008	Provide celestial objects position in the inertial reference frame
R-F-009	Identify feature matching for known geometries
R-F-010	Provide the current attitude

3.2 Centroid extraction algorithm

In the process of attitude reconstruction, the first step consists in being able to retrieve the position of some stars from an image. For the purpose of this thesis, such image is modeled as discussed in Chapter 2. This process is closely linked to the computation of the stars position in the camera reference frame.

3.2.1 Noise cancellation

Before acting on the image itself, pre-processing is required. In particular, it is necessary to remove the noise from the image in order to avoid errors due to this source. The canon is to set a threshold I_{thrs}, expressed in pixel intensity, and set the intensity of every pixel below such threshold to zero:

$$\begin{cases} I_{px}(x,y) = I_{px}(x,y) & \text{if} \quad I_{px}(x,y) > I_{thrs}\,; \\ I_{px}(x,y) = 0 & \text{if} \quad I_{px}(x,y) \leq I_{thrs}\,. \end{cases} \tag{3.1}$$

Nevertheless, it is required to define the threshold according to a precise method. A method is static when the threshold is defined once as a constant (K) and is independent on any parameter. On the other hand, it is known as dynamic when the threshold value depends on the image properties.

- static methods [9]: $I_{thrs} = K$;
- local dynamic methods [21]: $\begin{cases} I_{thrs} = \mu(I_{px}) + K\ \sigma(I_{px})\,; \\ \mu(I_{px}) = \frac{1}{N}\sum_{i=1}^{N_x}\sum_{j=1}^{N_y} I_{px}(x_i,y_j)\,; \\ \sigma(I_{px}) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N_x}\sum_{j=1}^{N_y} |I_{px}(x_i,y_j) - \mu(I_{px})|^2}\,; \end{cases}$
- iteratively dynamic methods [8]: *maximization of interclass variance*.

For this case, a local dynamic method is selected. This is due to the fact that it better fits each image with respect to a static method, since the threshold is computed according to the current pixels properties such as the mean (μ) and standard deviation (σ) of the pixel intensity over the total amount of pixels ($N = N_x N_y$). Moreover, its burden on the hardware is reduced with respect to an iterative method since the threshold value is computed only once per image.

3.2.2 Centroiding

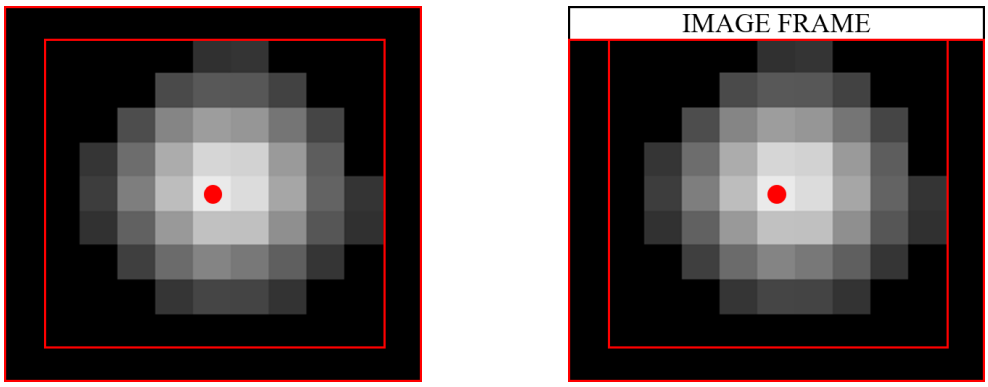
Once the image has been pre-processed, the actual centroiding algorithm is applied. It can be briefly summarized in the following steps:

- detect pixel level centroids coordinates;
- define centroiding window;
- compute sub-pixel level centroids coordinates.

These steps are sequential due to the algorithm nature.

In order to detect a pixel level centroid, the brightest pixel in the image is selected and taken as a candidate [22]. After the coordinates are collected, a squared centroiding window is selected such that the candidate celestial object is fully included inside its borders with a margin of one pixel on each side (Fig. 3.1).

The algorithm also takes into account the case for which the margin can't be achieved on at least one border because the celestial object ends on the image frame. In this case a non squared centroiding window is allowed.



(a) Standard case. (b) Image frame case.

Figure 3.1 Centroiding window.

Once a M-by-N region of influence is defined, a method to compute the sub-pixel level centroid has to be selected. The most commonly used methods [18] are:

- center of gravity;
- weighted center of gravity;
- iteratively weighted center of gravity.

In a general form, each family of methods relies on the computation of the image moments:

$$I_{HK} = \sum_{i=1}^{M} \sum_{j=1}^{N} x_i^H y_j^K I_{px}(x_i, y_j) W_{HK}(x_i, y_j)\,; \tag{3.2}$$

where x and y are the pixel level centroid coordinates and $W_{H,K}(x,y)$ is the weighting parameter, which assumes a different form according to the specific method. While the center of gravity method is the simplest among the listed ones, it is often outperformed by the other two methods. Nevertheless, the iteratively weighted center of gravity method requires a proper optimization process to work as intended.

Therefore, for the purpose of this thesis, the weighted center of gravity method is considered a proper trade-off. In particular, the intensity weighted center of gravity method is selected defining $W_{HK}(x,y) = I_{px}(x,y)$. Finally, under these assumptions, the sub-pixel level centroid coordinates can be retrieved:

$$\begin{cases} x_C = \frac{I_{10}}{I_{00}}\,; \\ y_C = \frac{I_{01}}{I_{00}}\,. \end{cases} \tag{3.3}$$

Once this procedure is completed, the previously defined centroiding window is deleted from the overall image and the extraction process is repeated for the next candidate until all the possible celestial objects have been evaluated. The flowchart in Fig. 3.2 represents the logical steps of the algorithm.

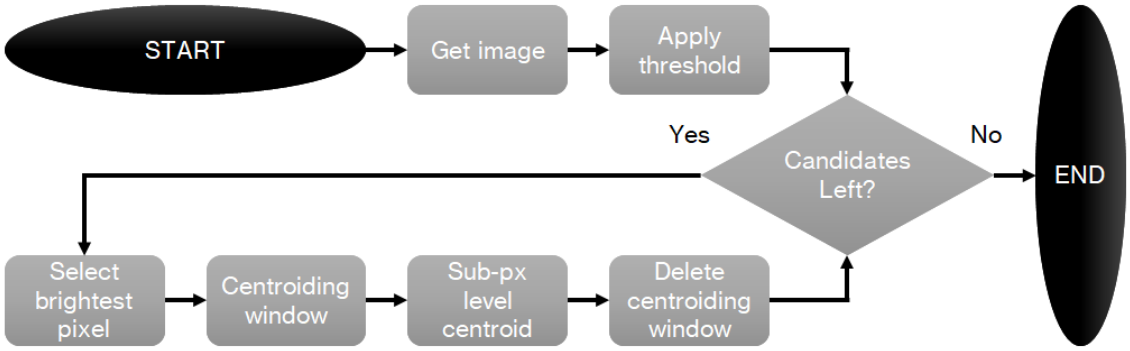


Figure 3.2 Centroiding algorithm flowchart.

3.3 Star identification

In order to retrieve the current attitude, it is necessary to identify at least three stars and get their position in both the inertial reference frame and in the camera one. It is trivial to extract the latter information from the stars centroid knowing the camera technical specifications.

3.3.1 Camera frame coordinates

The first step is to exploit Eq. (2.6) to convert the coordinates from the pixel reference frame to the image one and then combine Eq. (2.4) with the geometrical definition of the celestial sphere ($X_B^2 + Y_B^2 + Z_B^2 = 1$). This last step leads to:

$$\begin{cases} Z_B = \frac{f}{\sqrt{x_i^2 + y_i^2 + f^2}}; \\ X_B = -Z_B \frac{x_i}{f}; \\ Y_B = -Z_B \frac{y_i}{f}. \end{cases} \tag{3.4}$$

While the computation of the coordinates of a generic star in the camera reference frame is quite straightforward, retrieving the same data in the inertial reference frame requires a different approach.

3.3.2 Inertial frame coordinates

Taking into account that at least three stars are required to properly compute the spacecraft attitude. Considering exactly three stars, the most common method consist in defining some geometrical features for each triplet on the image and exploit them to identify at least one triplet. In order to accomplish this task, a database of features is defined and a matching criteria is applied to link the features on the image to their counterpart in the database.

3.3.3 Geometrical features

For the purpose of this thesis a quite simple set of parameters, known as Liebe's parameters [19], is selected. Indeed, exploiting the fact that the features are extracted from three stars at time, it is possible to consider the spherical triangle defined by such celestial objects as in Fig. 3.3.

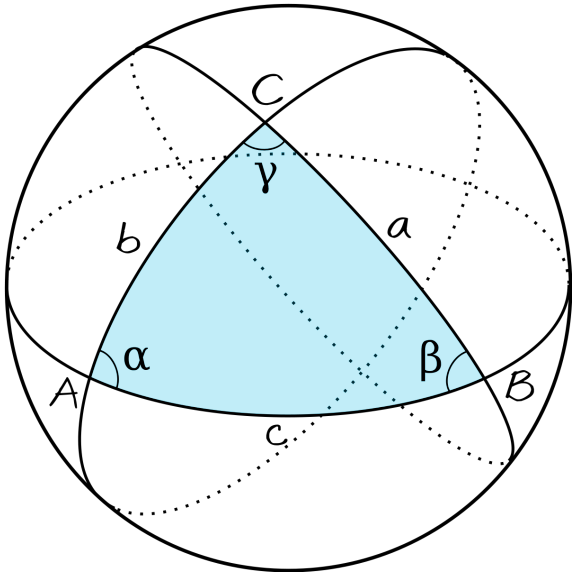


Figure 3.3 Spherical triangle. [I]

The points A, B and C represent the three stars and the spherical triangle is characterized by six parameters. However, three of those are enough to properly identify the stars. Therefore, considering A as a reference star, the geometrical features can be defined as:

- angular distance from the other stars (b, c);
- angle between those angular distances (α).

The reference star is selected according to its brightness, being the brightest in the triplet. In this context it is necessary to remind that a star brightness is directly linked to its magnitude which is available when dealing with a catalog.

However, when dealing with stars that have not been identified yet, this information is not available. To overcome this problem an equivalent brightness parameter is defined according to the pixel intensity distribution over a M-by-N centroiding window:

$$B = \sum_{i=1}^{M} \sum_{j=1}^{N} I_{px}(x_i, y_j)\,. \tag{3.5}$$

This parameter properly represents a celestial object brightness according to the model in Section 2.4.4, since from the photoelectrons distribution function is clear that:

- brightest stars activate more pixels;
- pixels at the same distance from the center are more intense for brightest stars.

[1] *https://www.cleanpng.com/*

For each triplet, the features can be extracted reminding that the stars positions on the celestial sphere (P_A, P_B and P_C) computed from Eq. (3.4) are unitary vectors:

$$\begin{cases} \cos(a) = P_B \cdot P_C \rightarrow a; \\ \cos(b) = P_A \cdot P_B \rightarrow b; \\ \cos(c) = P_A \cdot P_C \rightarrow c; \\ \cos(a) = \cos(b)\cos(c) + \cos(\alpha)\sin(b)\sin(c) \rightarrow \alpha\,. \end{cases} \tag{3.6}$$

Considering that, as previously stated, three parameters (b, c and α) are enough, the angular distance a is computed only because it is required to solve the equation to compute the angle α. Moreover, the equivalent brightness defined in Eq. (3.5) can be used to reject false celestial objects due to agglomerates of background leftovers by using a threshold value ($\overline{B}$):

$$\begin{cases} \text{if} \quad B < \overline{B}; \\ \text{reject candidate}\,. \end{cases} \tag{3.7}$$

.

3.3.4 Database

When it comes to building the database, it is necessary to build it without influencing the results of the algorithms, in particular the features must be available for any image coherently with the camera technical specifications.

The building procedure [23] consists in:

- selecting a set of evenly distributed pointing directions;
- considering the sub-set of stars within the field of view for each pointing direction;
- extracting features from such sub-set;
- saving such features in an unambiguous format.

Pointing distribution

There are several criteria to define a even distribution of points on a sphere (Fig. 3.4), among these:

- packing method: maximizes the minimum distance between points;
- covering method: minimizes the maximum distance between points;
- minimum energy method: exploits the repelling electrons model to minimize the Coulomb potential;
- maximal volume method: maximizes the volume of the convex hull.

The covering method is the one that better suits this work, indeed it allows to produce a distribution of points with the best worst case scenario.

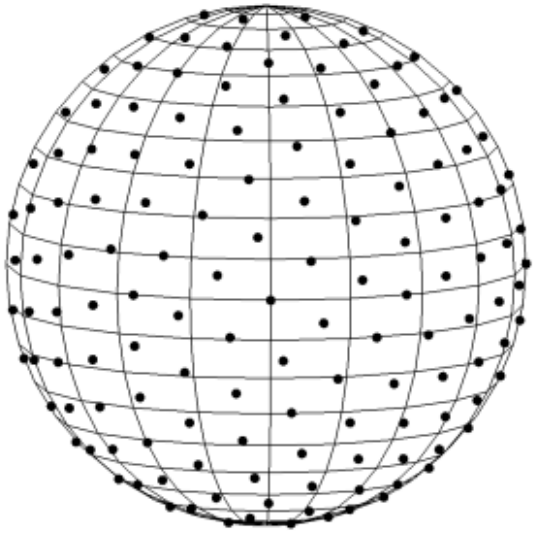


Figure 3.4 Even distribution of points on a sphere. [23]

Although a method is selected, the number of points that satisfy such definition is variable and up to the user. In particular, some optimal distributions [24] can be retrieved for different number of points (N_{pts}). For the covering method, a generic set of points can be identified as follows:

$$N_{pts}(i, j) = 2 + 10(i^2 + ij + j^2); \tag{3.8}$$

where the indexes i and j represent the spatial arrangement of such points, and the closer such indexes are the more optimized the distribution is.

Sub-set definition

Considering all the combinations of stars triplets within the field of view would lead to an excessive amount of data. A possible choice is to define a limit magnitude and exclude from the database the stars with magnitude higher than the selected one.

This limit could be expressed according to different criteria, for example:

- maximum magnitude the camera can appreciate;
- magnitude that allows to have at least 3 visible stars for every database image.

Between these examples, the second one allows to build a smaller database selecting only the three brightest stars for each image.

Nevertheless, the possible presence of planets and the intrinsic model errors may lead to a wrong identification of such stars. Therefore, the final choice is to select the n brightest stars for each pointing direction leading to a number of triplets (N) that can be computed as the number of k-combinations (k=3 for triplets) of n elements without repetitions:

$$N = \frac{n!}{k!(n-k)!}. \tag{3.9}$$

It is also necessary to remind that the reference star of each triplet is selected as the brightest star. This leads to the fact that due to the modeling approximation introduced in Eq. (3.5) the actual brightness may slightly vary from the equivalent brightness parameter. To avoid this problem the brightness of the h brightest stars in each triplet must be interchanged leading to:

$$N = h\left[\frac{n!}{k!(n-k)!}\right]. \tag{3.10}$$

Finally, the entry for each triplet is defined as in Eq. (3.11) leading to an overall database matrix with dimensions N-by-6 for each image.

$$DB = \begin{bmatrix} ID_A & ID_B & ID_C & \alpha & b & c \end{bmatrix}. \tag{3.11}$$

However, the dimensions of the database that takes into account all the pointing directions can not be computed a priori since some triplets combinations will be repeated for images of close portions of the sky. All the repeated triplets are deleted in order to reduce the overall database size.

3.3.5 Matching algorithm

Once the database is ready and the features have been computed for the image under study, it is necessary to extract the IDs of the stars corresponding to such features.

The proposed algorithm (Fig. 3.5) works by extracting the features from the m brightest objects on the image, with brightness defined as in Eq. (3.5). Then the features are computed for the triplets formed by such objects and collected in a M-by-6 set with:

$$M = \left[\frac{m!}{k!(m-k)!}\right]. \tag{3.12}$$

The features of each triplet belonging to this set are compared to those in the database according to a cost function (J):

$$J = |\Delta\alpha| + |\Delta b| + |\Delta c|; \tag{3.13}$$

where Δ represents the difference between the same feature on the set and on the database.

Finally, the triplet that minimizes the defined cost function is selected as the matching one, the IDs of the stars are extracted from Eq. (3.11) and with this information the angular coordinates on the celestial sphere are collected from the Hipparcos catalog. The position vector in the inertial frame can be computed from Eq. (2.8) once the angular coordinates are known.

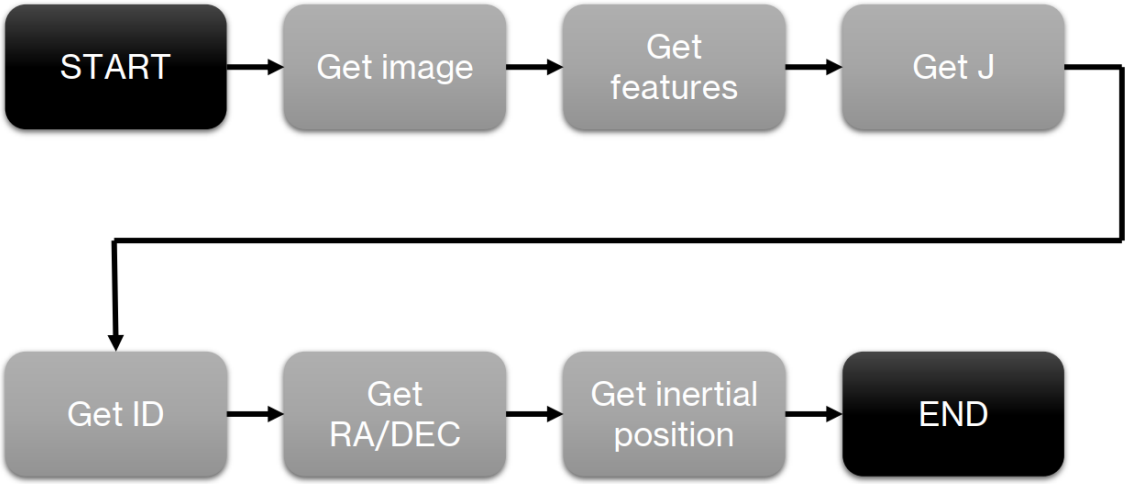


Figure 3.5 Star identification flowchart.

Moreover, a rejection threshold ($\bar{J}$) is experimentally defined in order to prevent any false identification.

$$\begin{cases} \text{if} \quad J > \bar{J}; \\ \text{reject match}. \end{cases} \tag{3.14}$$

3.4 Attitude determination

As stated in Section 3.3.1 it is possible to compute the position of three stars in the camera reference frame and, following the procedure in Section 3.3.2 the same information can be retrieved in the inertial reference frame. These data can be collected in matrices as:

$$A_B = \begin{bmatrix} X1_B & X2_B & X3_B \\ Y1_B & Y2_B & Y3_B \\ Z1_B & Z2_B & Z3_B \end{bmatrix} \; ; \; A_N = \begin{bmatrix} X1_N & X2_N & X3_N \\ Y1_N & Y2_N & Y3_N \\ Z1_N & Z2_N & Z3_N \end{bmatrix} . \tag{3.15}$$

Once at least three linearly independent directions are known in both the camera (v_{Bi}) and inertial (v_{Ni}) reference frame, there are a variety of methods that can be applied in order to retrieve the actual attitude matrix.

For the purpose of this thesis, the singular value decomposition method [20] is applied, knowing its limitations due to the fact that it is a static method. The reason of this choice

is related to the consideration that the investigation of the attitude determination process is not the main goal of this work but a step to solve a more complex problem.

According to this method the computation of the attitude matrix ($\mathbf{R}_{BN}$) relies on the minimization of a cost function defined as:

$$J = \frac{1}{2}\sum_{i=1}^{N} \alpha_i ||v_{Bi} - \mathbf{R}_{BN} v_{Ni}||^2 ; \tag{3.16}$$

where α is a weighting parameter that depends on the sensor with which the i-th measurement is taken.

With some algebraic steps, it is possible to retrieve the counterpart function to maximize, which depends on the trace of the product between $\mathbf{R}_{BN}$ and $\mathbf{B}$ as in Eq. (3.17).

$$\begin{cases} \widetilde{J} = \sum_{i=1}^{N} \alpha_i (v_{Bi}^{\intercal} \mathbf{R}_{BN} v_{Ni}) = tr(\mathbf{R}_{BN}\mathbf{B}^{\intercal}) ; \\ \mathbf{B} = \sum_{i=1}^{N} \alpha_i (v_{Bi} v_{Ni}^{\intercal}) ; \end{cases} \tag{3.17}$$

in these equations N is the number of linearly independent directions, which in this case is considered to be three.

Considering the singular value decomposition $\mathbf{B} = \mathbf{U}\mathbf{S}\mathbf{V}^{\intercal}$, it can be proven [20] that the attitude can be retrieved as:

$$\mathbf{R}_{BN} = \mathbf{U}\mathbf{M}\mathbf{V}^{\intercal}; \tag{3.18}$$

where $\mathbf{M}$ is defined starting from $\mathbf{U}$ and $\mathbf{V}$ as:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det(\mathbf{U})det(\mathbf{V}) \end{bmatrix} . \tag{3.19}$$

Chapter 4

Line of sight extraction

4.1 Motivation and purpose

The last aim proposed in this work is, as stated in Section 1.3, to retrieve the line of sight of a planet (i.e. its direction with respect to the spacecraft). Such planet is selected according to a precise criterion, and the information on its line of sight can be later exploited to retrieve the spacecraft position.

The main process relies on the availability of an initial guess of the spacecraft position provided with an error kept under a certain value. Moreover, the ephemerides of the planets of interest and the time at which the algorithm is run have to be always accessible. The algorithm computes a possible position for the selected planet according to the provided data and asks for a slew manoeuvre to point towards such direction. Once the pointing is completed, a picture is obtained through the model described in Chapter 2 and processed in order to get the desired line of sight.

The main problem is that the error in the initial position involves an error in the pointing direction computation, therefore the algorithm also has to find the planet among the celestial objects present on the image. Other minor problems are present and are related to the visualization of the planet provided by a camera with specific settings.

The final requirements for the line of sight extraction algorithm are presented in Table 4.1.

Table 4.1 Line of sight extraction functional requirements.

ID	Requirement
R-F-011	Provide desired pointing for a given planet selection criterion
R-F-012	Identify the planet in presence of pointing error
R-F-013	Provide the planet line of sight

4.2 Initial state

The problem of attitude reconstruction addressed in Chapter 3 deals with lost in space initial conditions, implying a completely unknown initial state. On the other hand, when it comes to the line of sight extraction problem, the first step for the purpose of this work is to deal with an initial guess of the spacecraft position in the inertial frame (r_{0N}) and the knowledge of the time at which the algorithm is run, expressed according to the modified Julian date (*MJD2000*).

In order for the problem to be coherent, the image generated from the model in Chapter 2 takes into account the real position of the spacecraft (r_N), also expressed in the inertial reference frame. However, the computations are carried on with a known position guess which has an embedded error expressed in the body frame (Δr_B) of intensity ε_r. The error is modeled on the image plane, since the main problem is related to the position of the planet on such plane.

Considering the parameters illustrated in Fig. 4.1 and the knowledge of the exact rotation matrix ($\boldsymbol{R}_{BN}$), the initial guess on the spacecraft position can be computed as:

$$\begin{cases} r_B = \mathbf{R}_{BN}^{\intercal}\, r_N\,; \\ \Delta r_B = \varepsilon_r[\cos(\theta)\ \sin(\theta)\ 0]^{\intercal}\,; \\ r_{0B} = r_B + \Delta r_B\,; \\ r_{0N} = \mathbf{R}_{BN}\, r_{0B}\,. \end{cases} \tag{4.1}$$

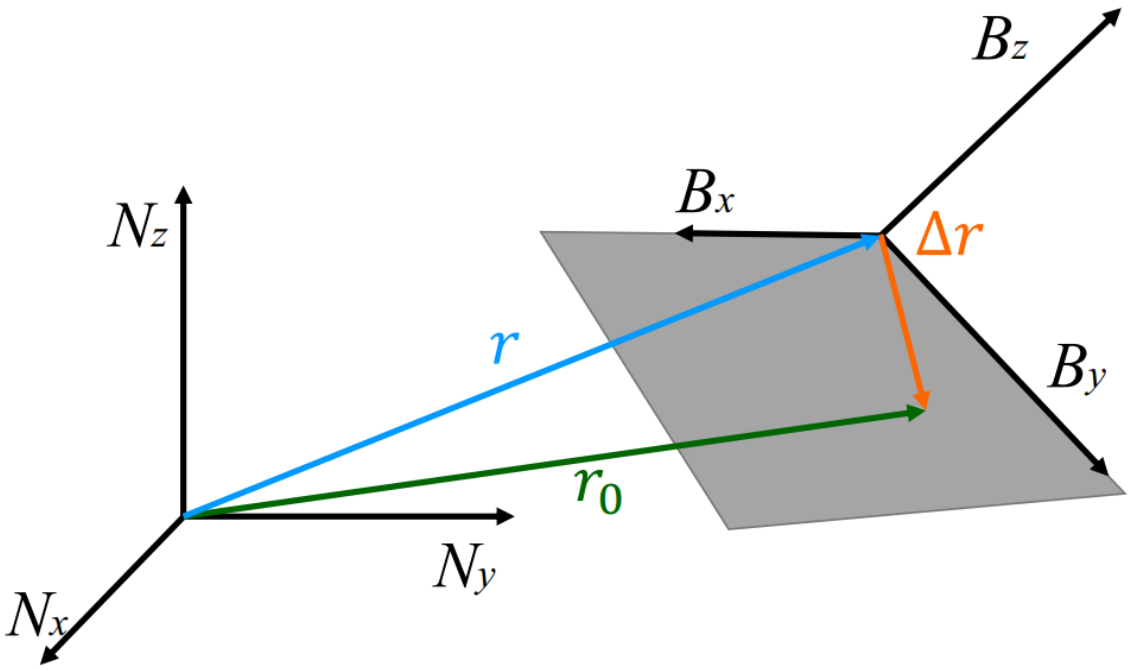


Figure 4.1 Error on the initial position.

4.3 Extraction algorithm

The core of this process is the extraction of the line of sight. The proposed extraction algorithm involves different steps which, from the computation of the desired pointing direction lead to the final result. Such steps also involve the correction of visualization errors and the planet identification.

4.3.1 Desired poiting direction

Once the initial position guess is defined, the following step is choosing the direction in which the camera has to be pointed to find a possible planet. This requirement, involves the selection of a criterion to define which planet has to be pointed. For the purpose of this thesis, the brightest planet (i.e. with the lowest magnitude) is selected as a candidate planet.

Considering the previously computed initial position guess, and the on-board availability of the planets ephemerides, the position (ρ) and magnitude (V) can be computed for every k-th major body as discussed in Section 2.4.2. Finally, the desired pointing direction ($\overline{\rho}$) can be selected as:

$$\begin{cases} \overline{V} = \min_k(V_k)\,; \\ \overline{\rho} = \rho(\overline{V})\,. \end{cases} \tag{4.2}$$

4.3.2 Planet identification

Once the desired pointing direction is chosen, another image is generated according to the model in Chapter 2. Since the camera is set to point a selected planet, such major object is expected to be at the center of the image, however this is not the case. This is due to the fact that the desired pointing direction has been computed taking into account a position different from the real one (i.e. r_{0N}), which means that the camera is not exactly pointing the desired planet as depicted in Fig. 4.2.

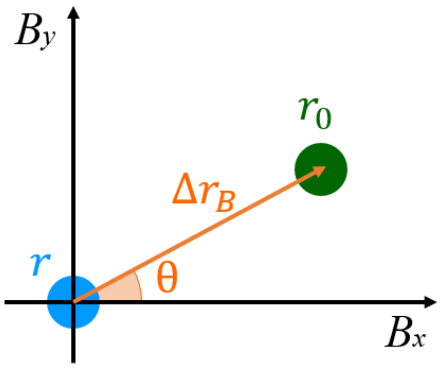


Figure 4.2 Camera poiting error.

In order to overcome this problem, the centroiding algorithm developed in Section 3.2 is exploited to compute the centroid coordinates vector (C) of the celestial objects on the screen, together with their equivalent brightness (B) from Eq. (3.5).

Once this task is accomplished, the bi-dimensional distance from each celestial object to the center of the image is computed. Finally, the object selected as a candidate planet is chosen as the one that minimizes a function (J) defined in Eq. (4.3).

$$J_k = \left(1 - \frac{B}{max_k(B_k)}\right) + \frac{D}{max_k(D_k)}. \tag{4.3}$$

This allows to define the candidate planet as the brightest object closest to the pointing direction, coherently with the fact that the camera is theoretically pointing (i.e. closest to the center) the most visible planet (i.e. the brightest one).

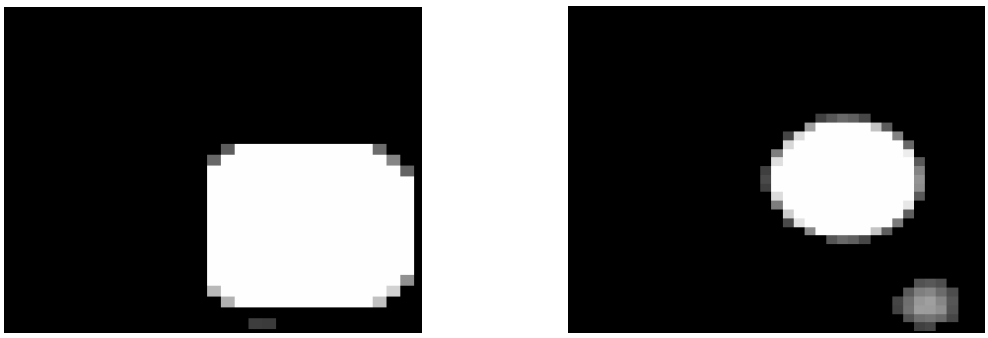
4.3.3 Visualization error

Even in the case in which the candidate planet actually is the desired planet, some errors may arise in the visualization process leading to a wrong line of sight computation. The main error that can be encountered is the presence of active pixels inside the centroiding window that do not belong to the planet. This can happen due to residual background noise or because other light sources are present inside the centroiding window.

Residual background noise and unconnected light sources

When it comes to residual background noise, the planet active pixels are generally detached from the noise-generated ones as represented in the planet close up in Fig. 4.3a. It is possible to identify the boundaries of such regions and select the brightest one, leading to the exclusion of the residual background noise contribution.

The same procedure can be applied if a light source produces a region which is not connected to the planet one as depicted in the planet close-up Fig. 4.3b.



(a) Residual background noise. (b) Unconnected light sources.

Figure 4.3 Unconnected visualization errors.

Connected light sources

Another problem that can be present during the process of the line of sight extraction is the presence of a source of light close enough to the planet that the the regions defined at the pixel level are connected as in the planet close-up in Fig. 4.4.

Although some algorithms that are capable of identifying the two sources are available in literature, for the purpose of this work the cases in which the planet merges with another source of light are rejected without undermining the validity of the algorithm.

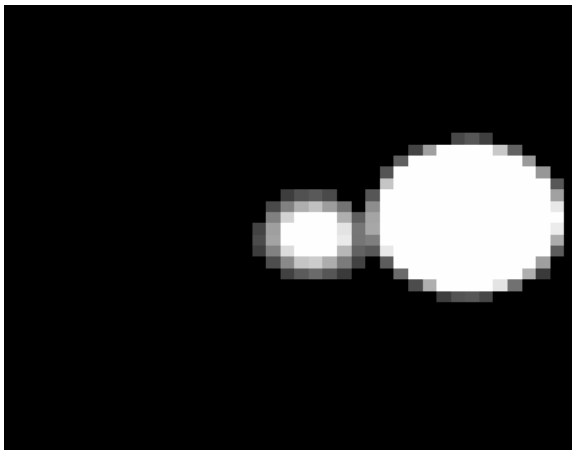


Figure 4.4 Connected light sources.

4.3.4 Planet direction computation

Finally, once the planet is identified and the errors are corrected, its centroid coordinates can be exploited to retrieve its direction (i.e. line of sight, LoS) in the camera reference frame through Eq. (3.4), since the hardware technical specifications are known.

Moreover, thanks to the availability of the attitude matrix computed through the steps explained in Chapter 3, the measurement of the line of sight can be converted to the inertial reference frame as:

$$LoS_N = \mathbf{R}_{BN}\, LoS_B\,. \tag{4.4}$$

Although it is possible to obtain the measurement in both reference frames, it is useful to notice that the error with which LoS_N is computed is greater than the one of LoS_B. This is due to the fact that the rotation matrix exploited for the conversion has an intrinsic error due to its computational nature, as shown in Fig. 5.5.

Chapter 5

Results

5.1 Star tracker model

In order to test the star tracker model some constellations are considered, comparing images from amateur photographers to the simulated ones with adjustments on the sensor properties to match the original images.

5.1.1 Framework

Before the analysis of such images, a reference celestial object has to be chosen to apply the equations by Marin and Bang Section 2.4.4. The selected object is α-Lyrae as suggested in [16] and its properties are reported in Table 5.1.

Moreover, the optical camera setup that characterize each image is listed in Table 5.2. In particular, the meaning of each parameter is:

- FoV: field of view ;
- f: focal length ;
- T: exposure time ;
- N_{px}: image size ;
- F: F-number ;
- Q_e: quantum efficiency ;
- T_{lens}: transmission factor ;
- Q_{max}: limitation on electrons for each pixel ;
- σ: defocus level.

Table 5.1 α-Lyrae optical parameters.

	V	F_λ	λ
α-Lyrae	0.03	3.44×10^{-8} $\frac{W}{m^2 \mu m}$	555.6 nm

Table 5.2 Optical camera setup.

	Orion	Canis Major	Crux
FoV [deg]	17.5	23.5	7.5
f [mm]	25	25	200
T [ms]	300	150	200
N_{px} [px]	1024x1024	1024x1024	1024x1024
F [-]	0.9	0.9	2.5
$Q_e \times T_{lens}$	0.49	0.49	0.49
Q_{max} [e]	14'000	14'000	14'000
σ [px]	2	2	2

Finally, the noise parameters applied are those in Table 5.3 with the variables meaning intended as:

- ε_Q: Quantization;
- ε_R: Readout;
- ε_{FP}: Fixed Pattern;
- ε_{DS}: Dark Signal;
- ε_{DSNU}: Dark Signal Non Uniformity;
- ε_{PRNU}: Photo Response Non Uniformity;
- $\varepsilon_{\%}$: margin.

Table 5.3 Noise parameters.

ε_Q	ε_{FP}	ε_{DS}	ε_{DSNU}	ε_R	ε_{PRNU}	$\varepsilon_{\%}$
7 e	100 e	200 $\frac{e}{s}$	100 $\frac{e}{s}$	100 e	0.02	0.2

5.1.2 Validation

For this case, the validation process consists in identifying each constellation features in terms of their absolute and relative position. This is due to the fact that the real images are taken on a more complex spectrum than the one simulated. From Fig. 5.1, Fig. 5.2 and Fig. 5.3 it is possible to notice how the brightest stars are visible, making the comparison of the main constellation features (highlighted inside the circles) a viable task for each example.

Figure 5.1 Orion constellation: real image [I](on the left) and simulation (on the right).

Figure 5.2 Canis Major constellation: real image [II](on the left) and simulation (on the right).

[I] *https://www.skyatnightmagazine.com/*

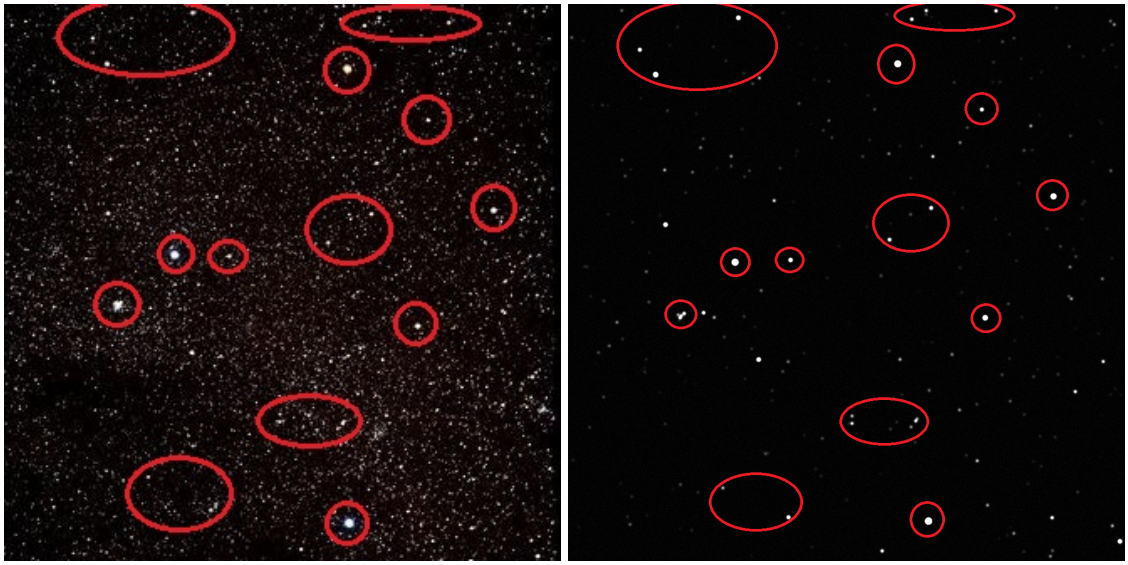


Figure 5.3 Crux constellation: real image [III](on the left) and simulation (on the right).

[II] *https://www.everypixel.com/*
[III] *https://www.sciencephoto.com*

5.2 Centroid extraction algorithm

The extraction algorithm is tested on a specific set of images that is selected in order to avoid the results to be dependant on such set. Moreover, the camera setup is selected and kept constant for all the simulations to achieve a better comparison between the data.

5.2.1 Framework

Although, in the previous section, the optical camera setup is adapted to match each test image, it is necessary to select a fixed setup for a proper analysis of the following results.

The noise and reference celestial object are still those in Table 5.1 and Table 5.3. In order to select a proper framework, data from different cameras are collected in Table 5.4.

Table 5.4 Comparison between different optical cameras setup.

	FoV [deg]	Image size [px]	f [mm]	F [-]	$Q_e \times T_{lens}$
NavCam [IV]	16×10	2048×1280	40	3.2	-
Sinclair ST-16RT2 [V]	15×20	2592 x 1944	16	1.6	-
Blue Canyon NST [VI]	10×12	-	-	-	-
HAS 2 [VII] [VIII]	20×20	1024×1024	-	-	0.45
FaintStar [IX] [X]	20×20	1024×1024	-	-	0.49
CMV 4000 [XI]	14×14	2048×2048	44	1.1	0.60

Therefore, the final optimal camera setup is selected as in Table 5.5. Moreover, the exposure time, the maximum number of electrons per pixel and the defocus level are the same as in Table 5.2.

Table 5.5 Optical camera setup.

	FoV [deg]	Image size [px]	f [mm]	F [-]	$Q_e \times T_{lens}$
Selected setup	20×20	1024×1024	40	3.2	0.49

[IV] First iteration of M-ARGO NavCam [6].
[V] *http://www.sinclairinterplanetary.com/*
[VI] *https://www.bluecanyontech.com/*
[VII] *https://www.cypress.com/*
[VIII] *https://www.cypress.com/*
[IX] *https://www.terma.com/*
[X] *https://link.springer.com/*
[XI] *https://ams.com/*

When it comes to the set of pointing directions from which the results are extracted, the selection falls on a set of 1082(6,6) points defined according to Eq. (3.8). Finally, selecting K = 10 to compute the threshold of Eq. (3.1), the framework is completely characterized.

5.2.2 Validation

The validation results are taken considering the centroid error relative to the 6 brightest objects for each image and shown in Fig. 5.4 according to their mean (μ) and standard deviation (σ) defined in Table 5.6. The data can also be interpreted by expressing them in arcseconds with the following conversion: $arcsec = 3600 \frac{FoV}{\text{image size}} px$.

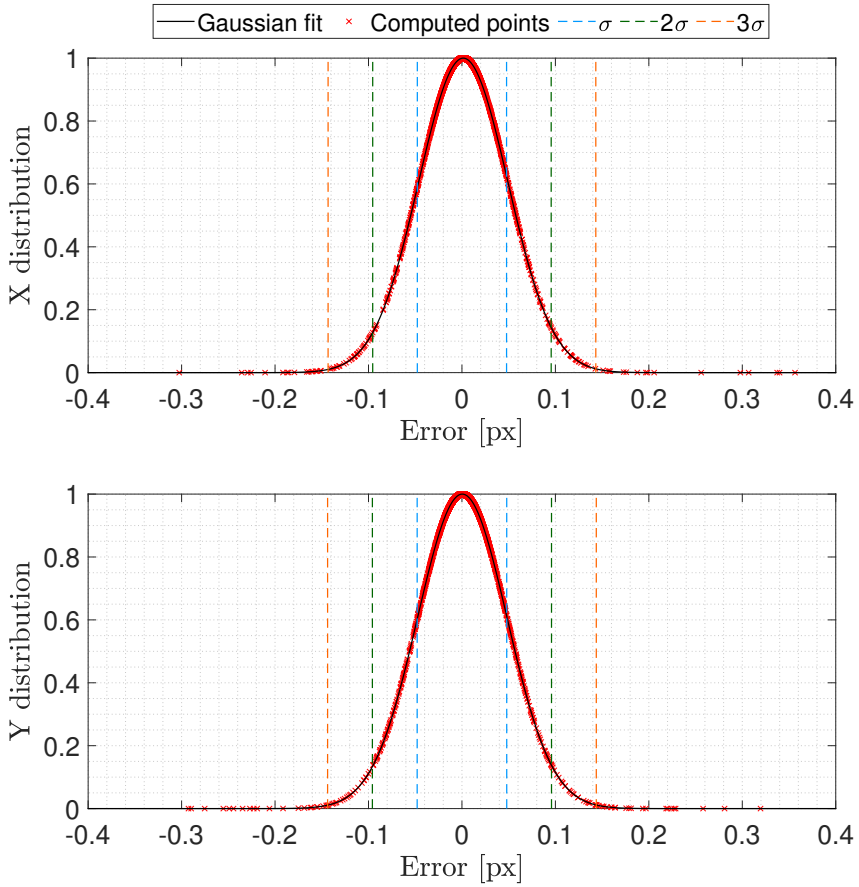


Figure 5.4 Centroiding error distribution.

Table 5.6 Centroiding error distribution features.

	μ [px]	σ [px]	μ [arcsec]	σ [arcsec]	$\leq \sigma$ [%]	$\leq 2\sigma$ [%]	$\leq 3\sigma$ [%]
X	0.0014	0.0477	0.0984	3.354	89.33	96.61	98.77
Y	0.0004	0.0479	0.0281	3.368	89.92	96.98	98.49

5.3 Star identification and attitude determination

In order to test the star identification and attitude determination algorithms, the same setup and validation process of the centroid extraction algorithm is used.

The algorithms are both tested taking into account the pointing error (ε), defined as the angle between the real pointing direction and the computed one, expressed in arcseconds. Moreover, the validity of the first algorithm is tested taking into account its failing modes occurrences.

5.3.1 Framework

A comparison among different values of the parameters involved in Eq. (3.11) and Eq. (3.12) is carried on as a first part of the testing procedure, and then the data are analysed for the best choice among those values.

Some parameters have been tuned according to a trial and error process, as the threshold for agglomerates of background leftovers ($\overline{B}$) and the rejection threshold for false identifications ($\overline{J}$) from Eq. (3.7) and Eq. (3.14). The selected values are those in Eq. (5.1).

$$\begin{cases} \overline{B} = 50 & [-]\,; \\ \overline{J} = 2 \times 10^{-3} & [rad]\,. \end{cases} \tag{5.1}$$

The matching and database parameters to be tested are identified with a code for simplicity. Indeed, each case is named as **nhm**, with clear meaning of the parameters from Eq. (3.11) and Eq. (3.12).

5.3.2 Validation

The previously mentioned error is fitted to an half-gaussian distribution with standard deviation equal to σ_ε and mean equal to zero. The distribution behaviour is due to the error intrinsic nature:

$$\varepsilon = 3600 \, \arccos(P_N \cdot P'_N)\,; \tag{5.2}$$

where P_N is the real pointing direction while P'_N is the computed one, with both being expressed in the inertial reference frame. The values of the standard deviation are reported in Table 5.7 for each case, together with the relative data distribution.

Table 5.7 Attitue error for different parameters.

nhm code	σ_ε [arcsec]	$\leq \sigma_\varepsilon$ [%]	$\leq 2\sigma_\varepsilon$ [%]	$\leq 3\sigma_\varepsilon$ [%]
936	6.1075	82.39	94.22	98.04
836	6.0558	82.09	93.94	98.04
736	6.2105	83.02	94.03	97.86
636	5.9630	81.99	94.59	97.95
629	6.1015	81.90	94.50	98.23
628	6.0841	81.90	94.59	98.23
627	5.9868	81.62	94.40	98.23
626	5.9444	82.09	94.68	97.95
616	5.5887	81.12	94.58	97.85

When it comes to star identification, the other parameter to take into account is the occurrences of failing modes and their nature. There are two ways such algorithm is unable to retrieve the desired output:

- there are less than 3 identified stars;
- the minimum cost function J is greater than its exclusion threshold $\overline{J}$.

The first rejection mode is related to the fact that the identification features are computed relaying on the presence of three stars. Moreover, in order to compute the attitude matrix, at least three independent directions are required (i.e. 3 stars).

The second rejection mode may be due to the presence of planets that leads to false features or to a poor noise cancellation. It is useful to notice that the proposed algorithm is capable of autonomously identifying the failing modes and exclude them from computation, avoiding the production of wrong outputs. The results related to such process are presented in Table 5.8.

Considering both Table 5.7 and Table 5.8, the objective is to get valuable data with a reduced database dimension computed according to Eq. (3.11) and a relief on the on-board computational burden from Eq. (3.12). This leads to the selection of the nhm code 626, which results are depicted in Fig. 5.5.

Table 5.8 Star identification rejected cases.

nhm code	Rejected cases	Rejection for J > $\bar{J}$	Rejection for $N_{stars} < 3$
936	9 (0.83 %)	3 (0.28 %)	6 (0.55 %)
836	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
736	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
636	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
629	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
628	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
627	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
626	10 (0.82 %)	4 (0.37 %)	6 (0.55 %)
616	12 (1.10 %)	6 (0.55 %)	6 (0.55 %)

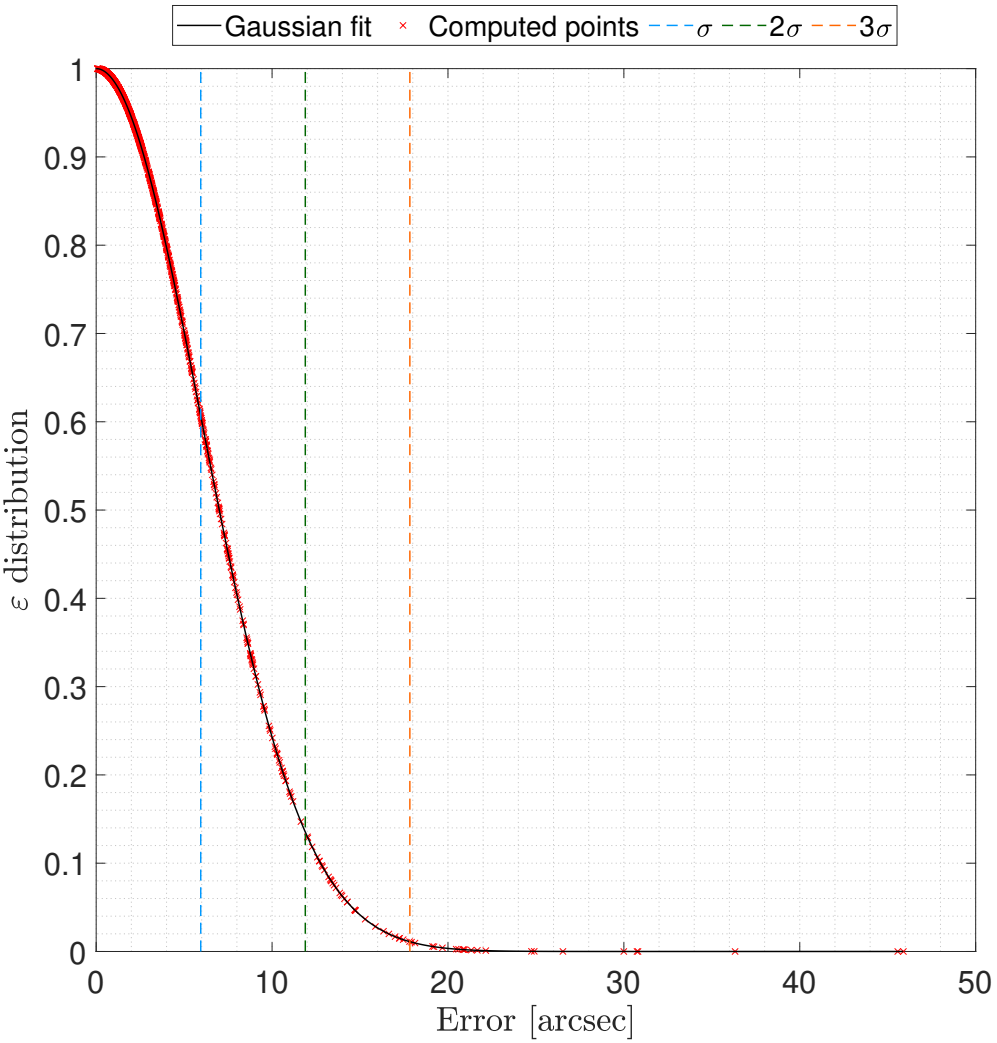


Figure 5.5 Attitude error distribution.

5.4 Line of sight extraction

The line of sight extraction algorithm is tested taking into account the maximum error on the initial position guess that still allows to identify the planet and its direction avoiding wrong identifications with different celestial objects. Moreover, the amount of cases in which it fails due to the presence of connected light sources as defined in Section 4.3.3 are also considered.

5.4.1 Framework

The camera setup is the randomly generated one used in the previous cases . This is true also for the chosen set of pointing directions, while the star identification method is the one with nhm code 626 for the reasons explained in Section 5.3.2. The spacecraft position is randomly computed in a range within 0.5 AU and 10.5 AU for each image. These values are a good portion of the solar system planets position, being representative, in terms of orders of magnitude, of the distance from the Sun of Mercury and Saturn.

The test is carried on according to different error vectors (Δ_r) on the initial position guess. The position error intensity (ε_r) is defined to prove up to which point the algorithm is still valid. On the other hand, the error vector direction, represented by the angle θ is randomly selected in a range between 0 and 2π radians for each image. From these values Eq. (4.1) can be exploited to compute the error vector.

5.4.2 Validation

The error in the line of sight computation is an angle between two vectors, the real planet line of sight direction (*LoS*) and the computed one (*LoS'*), indeed it can be computed as:

$$\varepsilon = 3600 \arccos(LoS \cdot LoS') \tag{5.3}$$

From the simulations, the algorithm is valid for $\varepsilon_r \leq 10^6$ km. Among the tested values of ε_r and reported in Table 5.9, the relevant cases are those on the first three rows and are also reported in Fig. 5.6. This is due to the fact that for $\varepsilon_r > 10^6$ km the planet towards which the camera is theoretically pointing can be outside the image.

Since the goal is to make the algorithm available in any case, the values of ε_r for which this happens at least once are discarded. Finally, the error distributions for the most significant cases are presented in Fig. 5.7, Fig. 5.8 and Fig. 5.9.

Table 5.9 Line of sight error for different parameters.

ε_r [km]	σ_ε [arcsec]	$\leq \sigma_\varepsilon$ [%]	$\leq 2\sigma_\varepsilon$ [%]	$\leq 3\sigma_\varepsilon$ [%]
10^4	6.6924	84.54	88.89	97.96
10^5	5.7415	85.56	91.57	96.67
10^6	5.6085	86.39	91.57	97.22
10^7	-	-	-	-
10^8	-	-	-	-
10^9	-	-	-	-

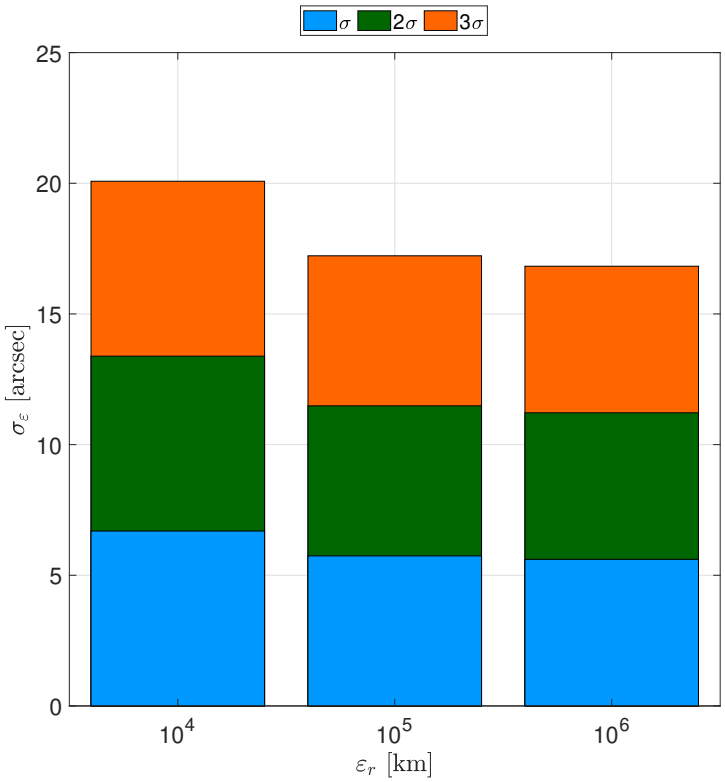


Figure 5.6 Line of sight standard deviation according to position error.

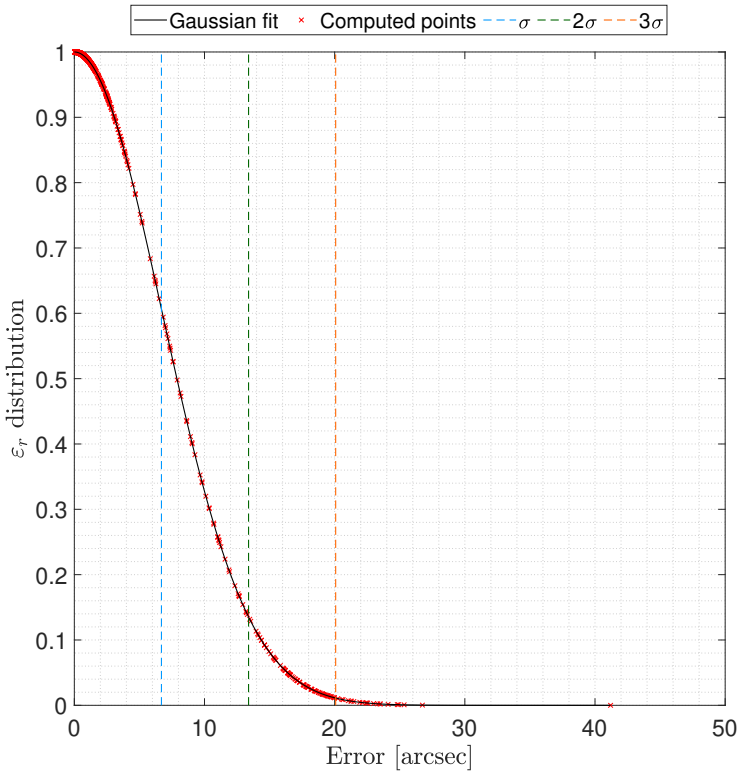


Figure 5.7 Line of sight error distribution for $\varepsilon_r = 10^4 km$.

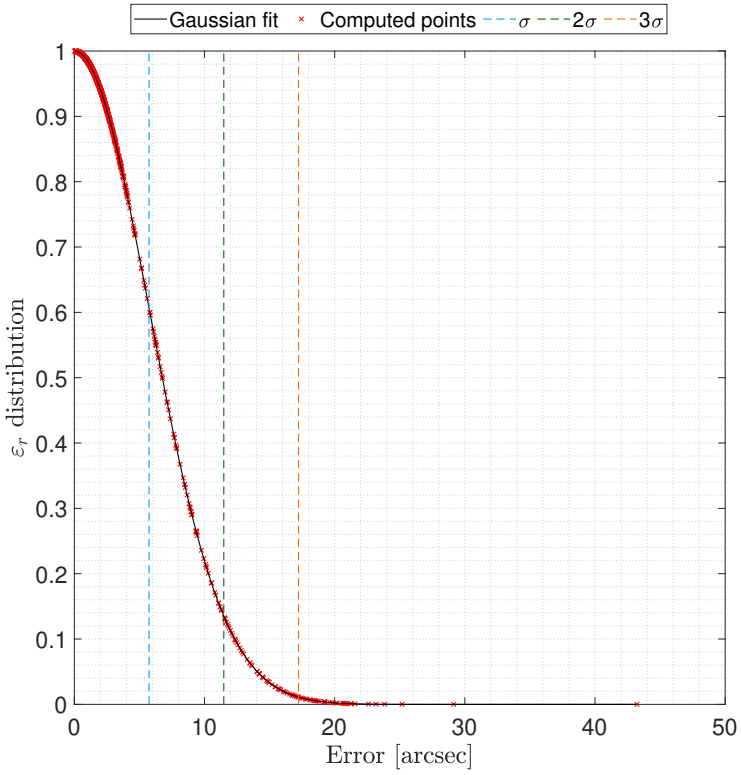


Figure 5.8 Line of sight error distribution for $\varepsilon_r = 10^5 km$.

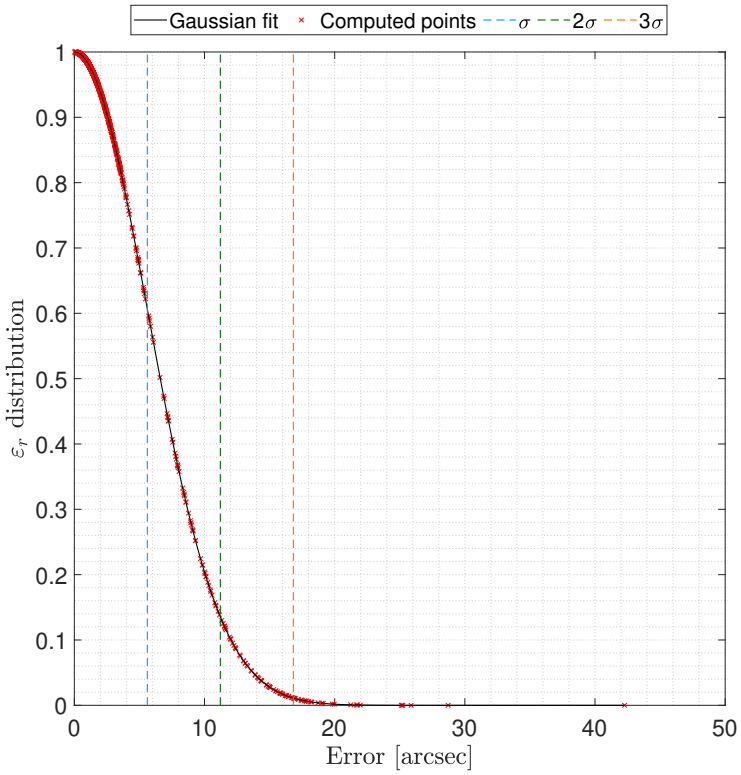


Figure 5.9 Line of sight error distribution for $\varepsilon_r = 10^6 km$.

5.5 Test case

In order to better explain the procedures involved in the overall planet line of sight extraction algorithm, it is useful to provide a step by step commented example. Reminding the premises in Section 5.4.1 and the assumptions done in the previous sections, it is possible to consider a starting state with initial position (r_N) and attitude matrix ($\mathbf{R}_{BN}$) defined by the pointing direction ($[\alpha\ \delta\ \phi]^intercal$), and generate the picture relative to this conditions as shown in Fig. 5.10 where the stars used for the initial attitude determination are those inside circles.

For this case, the initial position is $r_N = [2.5756\ -5.4487\ 0]^\intercal \times 10^8$ km and the pointing direction, defined as $[5.1191\ 0.7870\ 0]$ radians, is computed with an error of 1.7206 arcseconds.

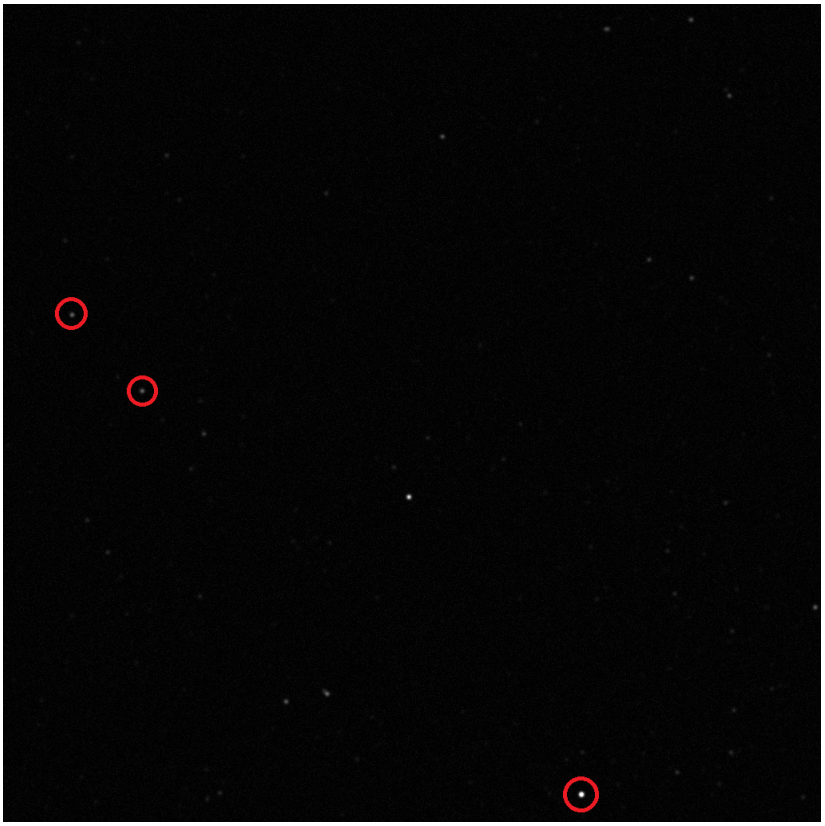


Figure 5.10 Test case: attitude determination.

Considering the measured position to have an error of $\Delta r_N = [-7.1190\ -6.3478\ -3.0043]^\intercal \times 10^5$ km (Fig. 5.11), the optimal planet is computed according to Section 4.3.2 and the results select Jupiter with a magnitude of -2.3355.

The pointing direction is updated according to the predicted direction of the planet and a new image is generated as in Fig. 5.12. As in the previous picture the circled celestial objects are those used for attitude determination. On the other hand, the celestial object inside the square is the candidate planet.

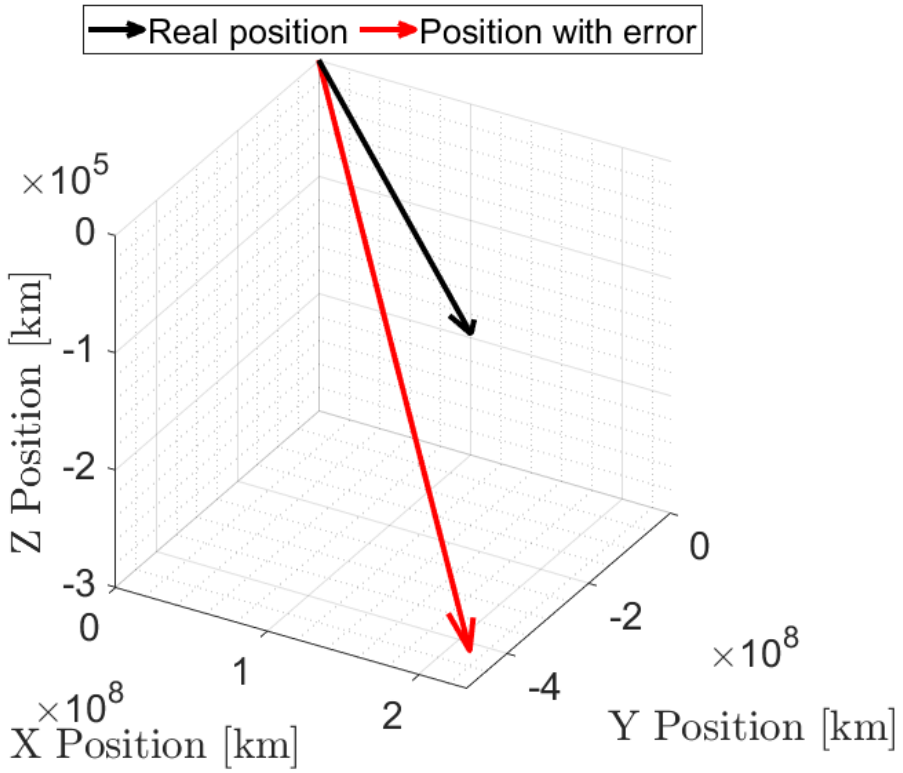


Figure 5.11 Test case: position estimation error.

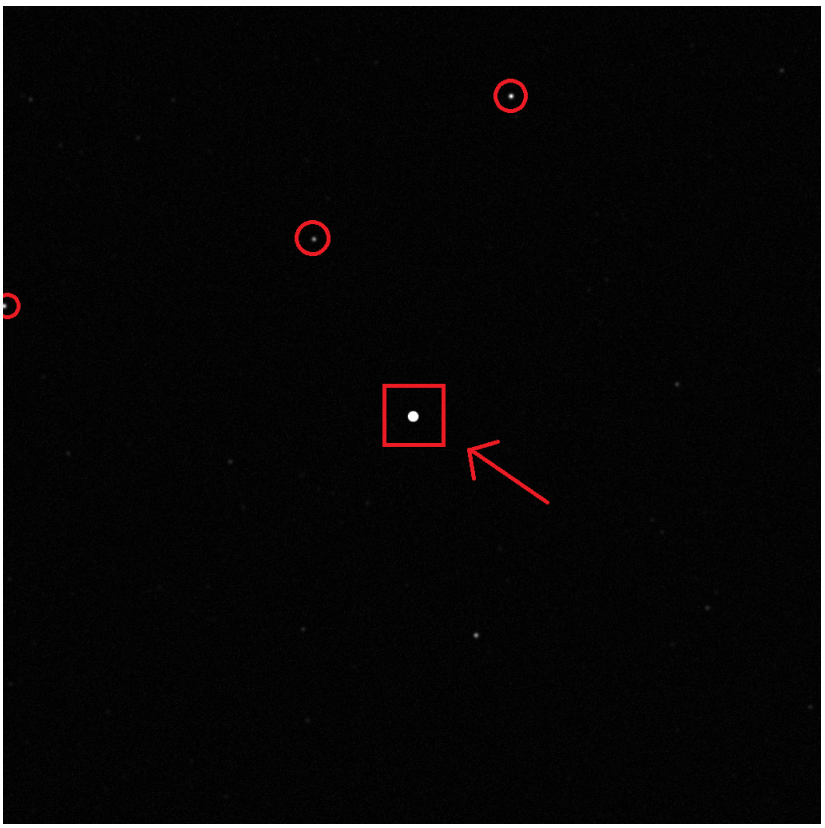


Figure 5.12 Test case: optimal planet pointing.

Indeed, considering a close up of the planet inside the square (Fig. 5.13), it is possible to notice the error introduced by the guess on the initial spacecraft position. This is the cause of the expected planet centroid not being located in the highlighted pixel (i.e. the center of the picture at coordinates $[512\ 512]^\intercal$ px).

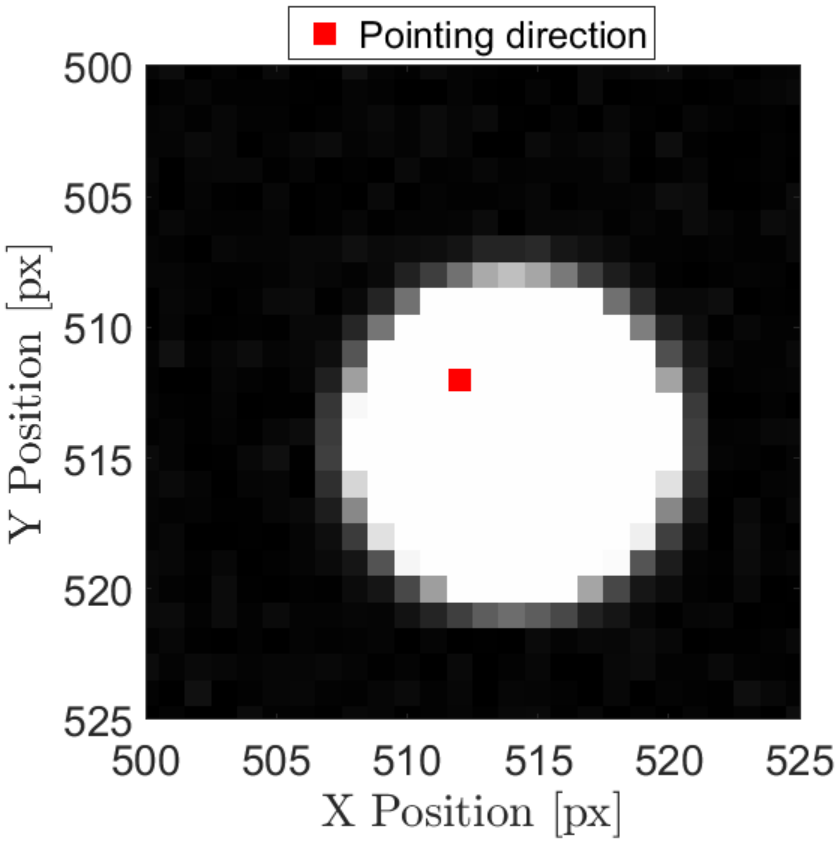


Figure 5.13 Test case: optimal planet pointing.

Finally, in presence of the described errors, the line of sight is evaluated according to the process presented in Chapter 4. The angle between the measured line of sight (LoS) and the real one (LoS_{real}) is 1.2655 arcseconds, being their difference in vectorial terms: $LoS - LoS_{real} = [1.5497\ 5.9366\ -4.5550 \times 10^{-3}]^\intercal \times 10^{-6}$ km.

Chapter 6

Conclusion

6.1 Considerations

The results in Chapter 5 can be analysed according to the premises in Section 1.3. Firstly, the star tracker model presented in Chapter 2 is able to represent specific patterns involving the absolute and relative position of the considered celestial objects (i.e. stars and planets) with respect to the spacecraft as shown in Section 5.1.2. Thus meaning that the results on which the presented algorithms are tested can be defined reliable.

The characteristic mean (μ) and standard deviation (σ) of the centroiding error reported in Section 5.2.2 identifies an algorithm capable of accomplishing its task within an error of 10.16 arcseconds, in terms of 3σ. Moreover, the proper definition and execution of the presented stars features matching and identification process is proven accordingly to the results on the attitude error in Section 5.3.2, with particular emphasis on the set-up with nhm code 626 accomplishing its task in 99.18% of cases and being able to autonomously identify the rejected cases for correction. The error for the whole attitude determination process remains within 17.83 arcseconds, again in terms of 3σ.

Finally, the planet line of sight extraction algorithm is able to find the requested direction for an error (ε_r) on the spacecraft position up to 10^6 km defined as in Section 5.4.2, leading to results within a 3σ range of 20 arcseconds.

Therefore, considering the results achieved by the attitude determination and line of sight extraction procedures, both in terms of standard deviation and maximum error, it is possible to state that the presented solutions work within the literature margins [25].

6.2 Future work

Considering the nature of the proposed solution and the deterministic link among all its steps, there are some improvements that can be applied to the process to increase its overall performance.

Centroiding

The centroiding computation algorithm can be enhanced with more complex methods, both in terms of thresholding techniques and sub-pixel level position, thus allowing an advancement in the computation of the celestial objects coordinates on the screen. Moreover, cases in which the representation on the screen of an object is connected to another light source can be processed according to the state of the art techniques.

Star features matching and identification

The star identification process can rely on the selection of more and/or different features, thus leading to better overall results. Also, the database can possibly be smaller and the overall required computational effort lower;

Attitude determination

The attitude determination solution can be optimized through state of the art procedures already available in literature, which can reduce the overall error in both angular and linear position determination;

Line of sight extraction

The planet selection process can be optimized according to the mission requirements. Moreover the process could serve as a basis for the development of a lost in space line of sight extraction algorithm (i.e. without the need of an initial guess on the spacecraft position).

All the proposed improvements would lead to an algorithm capable of finding the line of sight of an optimally selected planet with errors within the acceptable margins for the determination of the spacecraft position as well as its attitude. Thus making the guidance, navigation and control of a completely autonomous spacecraft an appealing alternative that does not require any change of the on board hardware with respect to the current standards.

Bibliography

[1] Todd Harrison, Zack Cooper, Kaitlyn Johnson, and Thomas Roberts. Escalation & deterrence in the second space age. 11 2017. doi: 10.13140/RG.2.2.15240.11525.

[2] Jiun-Jih Miau Richard Holdaway. *Reducing the Cost of Spacecraft Ground Systems and Operations*. Space Technology Proceedings 3. Springer Netherlands, 1 edition, 2000. ISBN 978-90-481-5400-5,978-94-015-9395-3.

[3] Kirk Woellert, Pascale Ehrenfreund, Antonio J. Ricco, and Henry Hertzfeld. Cubesats: Cost-effective science and technology platforms for emerging and developing nations. *Advances in Space Research*, 47(4):663–684, 2011. ISSN 0273-1177. doi: 10.1016/j.asr.2010.10.009.

[4] James S. Border Catherine L. Thornton. *Radiometric Tracking Techniques for Deep-Space Navigation*. Deep-Space Communications and Navigation Series. Wiley-Interscience, 2003. ISBN 9780471445340,0471445347.

[5] Reza Raymond Karimi and Daniele Mortari. Interplanetary autonomous navigation using visible planets. *Journal of Guidance, Control, and Dynamics*, 38(6):1151–1156, 2015. doi: 10.2514/1.G000575.

[6] V. Franzese. *Autonomous Navigation for Interplanetary CubeSats at different scales*. PhD thesis, Politecnico di Milano, 2021.

[7] Siliang Du, Mi Wang, Xiao Chen, Shenghui Fang, and Hongbo Su. A high-accuracy extraction algorithm of planet centroid image in deep-space autonomous optical navigation. *Journal of Navigation*, 69(4):828–844, 2016. doi: 10.1017/S0373463315000910.

[8] Xiuqiang Jiang, Shuang Li, Long Gu, Jun Sun, and Dongdong Xiao. Optical image generation and high-precision line-of-sight extraction for mars approach navigation. *Journal of Navigation*, 72(1):229–252, 2019. doi: 10.1017/S0373463318000450.

[9] J. Jiang, H. Wang, and G. Zhang. High-accuracy synchronous extraction algorithm of star and celestial body features for optical navigation sensor. *IEEE Sensors Journal*, 18(2):713–723, 2018. doi: 10.1109/JSEN.2017.2777493.

[10] M. A. C. Perryman, L. Lindegren, J. Kovalevsky, E. Hog, U. Bastian, P. L. Bernacca, M. Creze, F. Donati, M. Grenon, M. Grewing, F. van Leeuwen, H. van der Marel, F. Mignard, C. A. Murray, R. S. Le Poole, H. Schrijver, C. Turon, F. Arenou, M. Froeschle, and C. S. Petersen. The Hipparcos Catalogue. *Astronomy and Astrophysics*, 500:501–504, jul 1997.

[11] Howard D. Curtis. Chapter 4 - orbits in three dimensions. In Howard D. Curtis, editor, *Orbital Mechanics for Engineering Students (Third Edition)*, pages 187 – 237. Butterworth-Heinemann, third edition edition, 2014. doi: 10.1016/B978-0-08-097747-8.00004-9.

[12] C. C. Liebe. Accuracy performance of star trackers - a tutorial. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2):587–599, April 2002. doi: 10.1109/TAES.2002.1008988.

[13] Yi Ma, Stefano Soatto, Jana Kosecka, and S. Shankar Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003. ISBN 0387008934.

[14] Warren J. Smith. *Modern optical engineering: the design of optical systems*. McGraw Hill, 4th ed edition, 2008. ISBN 0071476873,978-0071476874.

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

[16] Mikaël Marin and Hyochoong Bang. Design and simulation of a high-speed star tracker for direct optical feedback control in adcs. *Sensors*, 20(8):2388, Apr 2020. doi: 10.3390/s20082388.

[17] Gerald C. Holst. *CCD Arrays, Cameras and Displays*. SPIE-International Society for Optical Engine, 2 sub edition, 1998. ISBN 0819428531,9780819428530,0964000040,9780964000049.

[18] A. Vyas, M. B. Roopashree, B. R. Prasad, and A. Vyas. Performance of centroiding algorithms at low light level conditions in adaptive optics. In *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, pages 366–369, 2009. doi: 10.1109/ARTCom.2009.30.

[19] Benjamin B. Spratling and Daniele Mortari. A survey on star identification algorithms. *Algorithms*, 2(1):93–107, 2009. ISSN 1999-4893. doi: 10.3390/a2010093. URL `https://www.mdpi.com/1999-4893/2/1/93`.

[20] Landis Markley and John Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. Springer-Verlag New York, 01 2014. ISBN ISBN: 978-1-4939-0802-8. doi: 10.1007/978-1-4939-0802-8.

[21] L. Kazemi, J. Enright, and T. Dzamba. Improving star tracker centroiding performance in dynamic imaging conditions. In *2015 IEEE Aerospace Conference*, pages 1–8, 2015. doi: 10.1109/AERO.2015.7119226.

[22] Malak Samaan and Stephan Theil. Development of a low cost star tracker for the shefex mission. *Aerospace Science and Technology*, 23(1):469–478, 2012. ISSN 1270-9638. doi: 10.1016/j.ast.2011.09.013.

[23] Tjorven Delabie, Thomas Durt, and Jeroen Vandersteen. Highly robust lost-in-space algorithm based on the shortest distance transform. *Journal of Guidance, Control, and Dynamics*, 36(2):476–484, 2013. doi: 10.2514/1.56860.

[24] R. H. Hardin, N. J. A. Sloane, and W. D. Smith. Tables of spherical codes with icosahedral symmetry. *http://NeilSloane.com/icosahedral.codes/*, 2012.

[25] James R. Wertz Wiley J. Larson. *Space Mission Analysis and Design*. Space Technology Library. Microcosm, 3rd edition, 2005. ISBN 1881883108.