POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Performance Evaluation of a Vision Based Navigation Algorithm for CubeSat Docking Missions

Supervisors

Professor Elisa CAPELLO

Professor Hyeongjun PARK

Candidate

Giacomo ICHINO

December 2021

Abstract

The availability of structures that can autonomously assemble in orbit could have a key role in future space explorations and research. Thanks to the last 20 years of miniaturization in space technologies, CubeSats have emerged as the perfect tool for in-orbit demonstration missions. However, a successful docking mission between two CubeSats still represents an unmet challenge because of the high accuracy required in the last meters of this operation. In fact, during the Final Approach (FA) phase, the relative position and attitude of the two spacecraft are coupled and all the 6 degrees of freedom of the chasing CubeSat must be estimated and controlled simultaneously. To perform this delicate relative motion, a Vision Based Navigation (VBN) solution represents the best available option.

In this thesis, a simulator of CubeSat docking missions is designed. A 3U "chaser", equipped with a monocular camera, approaches a "target", equipped with a cross-shaped pattern of Light Emitting Diodes (LEDs). Three different LEDs patterns, different in size and location on the docking face, are studied. The target travels on a circular Low-Earth-Orbit and is considered perfectly aligned to its local orbital frame. The mission starts from a first Station Keeping point (SK0): it represents the position in which the handover between the Guidance Navigation and Control system that previously operated the chaser and the VBN system occurs. The VBN algorithm is mimicked by generating fake LEDs' pictures based on the simulated true relative state. The LEDs' position in pixels is then corrupted to account for the optical device noise and LEDs detection error. These noisy positions are fed to an Extended Kalman Filter which, using a linearized version of the coupled dynamics involved, performs simultaneous estimation and filtering of the state variables. The estimated state is used by a Linear Quadratic Regulator to generate the position and attitude control inputs for the chaser. These inputs are corrupted with realistic sources of disturbance and then returned to the chaser, thus closing the loop.

The first goal of this thesis is to test the complete FA mission for the three LEDs patterns. Simulations show that the overshoot and final accuracy requirements can be always met, allowing the CubeSat to achieve docking with high precision regardless of the pattern used. The second goal of this thesis is to evaluate how the handover accuracy in SK0 affects the initialization of the VBN system. A Monte Carlo analysis is used to generate 100 sets of random initial conditions per pattern, and the initialization time required for each set is computed. The simulation shows how the performances of the three patterns are comparable. The final goal is to determine how the LEDs' detection error in pixels affects the initialization of the VBN system. During such phase, the two CubeSats are distant, so the

LEDs are seen closer to each other on the image plane and the detection error has the biggest impact on state estimation. A second Monte Carlo analysis is used to generate 100 random values for the standard deviation of the detection error. The simulation shows that two of the patterns perform in a comparable way. Consequently, the choice between the two can be purely motivated by hardware integration requirements. The third pattern instead, which is peculiar because of a smaller distance between its LEDs, requires more time for the stabilization. The spacing between the LEDs strongly affects the initialization performance.

Keywords: Space, CubeSat, Rendezvous, Docking, Vision-Based Navigation.

Acknowledgements

Giunto è il momento in cui voglio ringraziare
tutte le persone che negli anni han dato mano,
chi con un aiuto od un consiglio per studiare,
chi con un abbraccio che mi ha mantenuto sano.

Troppi son gli amici, tuttavia, che ho attorno,
prenderli uno a uno questa tesi poi m'allunga!
Quindi ho deciso che una buffa strofa sforno
per ogni bel gruppo che compone la mia giungla.

Grazie ai miei amici "Supersemiconduttori",
ciò che mi ha permesso di finir la triennale.
Lunghe le studiate e le bevute di liquori,
per poi ritrovarsi da Grellina ogni Natale.

Poi ci sono i Jimboss, primi amici torinesi,
solo il loro Idraulico è realmente di Torino.
Scattano le foto a verdure vecchie mesi
per poi conquistare tutti insieme la C1.

Quelli dell'Antella, sono il gruppo che mi ha accolto,
pare sia il furgone che mi tiene a lor vicino,
ma è il loro affetto a render luminoso il volto
proprio come quando parte il Mambo Ciabattino.

Gli amici di ACT, son gli amici delle bici,
a volte si pedala a volte invece si va al bar.
A volte s'organizzano garette assai felici,
ma in ogni occasione con sti matti è bello star.

Una strofa intera voglio invece dedicare,
al mio amico Banda, che è ormai per me un gemello:
senza il suo appoggio a questo mondo non so stare,
ogni sua parola è meglio del dono più bello.

Giunto sono quasi alla fine della lista,
e dei miei fretelli io volevo un po' rimar:
anche se da anni siamo sparsi, fuori vista,
siete i miei modelli ed il mio esempio da seguir.

Grazie mille volte invece ai miei genitori,
cittadini giusti come pochi a questo mondo.
Celebri poiché sono assai bravi revisori:
proprio a questa tesi han dato aspetto più rotondo.

Finally my advisors, you have unlocked space to me
opening careers in the field I always dream.
Even with new life, or being far far oversea,
thank you very much because your guidance was supreme.

Giacomo Ichino,
Decembre 5th 2021

Table of Contents

List of Tables

List of Figures

Acronyms

ATV Autnomous Transfer Vehicle

CDGPS Carrier Phase Differential GPS

CCD Charge Coupled Device

CoM Centre of Mass

CRP Close Range Point

DCM Direction Cosine Matrix

DoF Degrees of Freedom

ECI Earth-Centred Inertial

EKF Extanded Kalman Filter

ESA European Space Agency

FA Final Approach

FF Formation Flying

FoV Field of View

FRP Far Range Point

GNC Guidance Navigation and Control

GPS Global Positioning System

IOD In-Orbit Demonstration

ISL Inter Satellite Link

ISS International Space Station

KF Kalman Filter

LED Light Emitting Diode

LEO Low-Earth-Orbit

LoS Line of Sight

LQG Linear Quadratic Gaussian

LQR Linear Quadratic Regulator

LVLH Local-Vertical Local-Horizontal

MEMS Micro Electro Mechanical Systems

P2P Port To Port

PWPF Pulse-Width-Pulse-Frequency

RANSAC Random Sample Consensus

RCS Reaction Control System

RF Reference Frame

RMS Root Mean Square

ROI Region of Interest

RVD Rendezvous & Docking

RW Reaction Wheel

SIFT Scale-Invariant Feature Transform

SK Station Keeping

VBN Vision Based Navigation

Chapter 1

Introduction

1.1 Motivation and Background

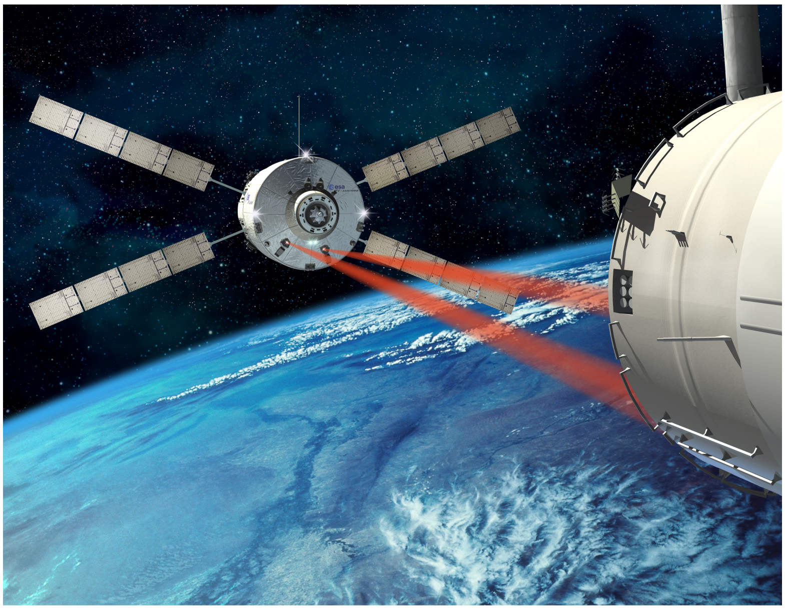
In the future, the availability of structures that can autonomously assemble directly in space could have a key role in space exploration and research. Space stations could be mounted starting from light and small building blocks, of the dimension of a shoe box, reducing the costs of launching bigger structures. New long telescopes, with lenses made of multiple stages, could be assembled in orbit without need of human intervention.

The building blocks of these innovations could be CubeSats, a cheap class of satellites which are already widely used by universities, state-owned or private companies and even amateurs. The key tool that engineers still need to improve and test is an efficient and cost effective autonomous docking technology. The goal of this thesis is to contribute to this fascinating quest.

1.1.1 Relative Navigation

Orbital Rendezvous & Docking (RVD) missions consist in two spacecraft approaching and then attaching one to the other by means of precise relative navigation, while orbiting around a celestial body. The two vehicles are commonly referred to as Target and Chaser. These types of maneuvers are of paramount importance in many space missions. In 1969 the Apollo 11 crew manually performed RVD between the lunar module and the command/service module, a key step of the first manned mission to the Moon [1]. In 1967 the Russian space program achieved the first autonomous RVD between the Cosmos 186 & 188 spacecraft [2]. Every time astronauts, supplies and scientific equipment reach the International Space Station (ISS) an RVD mission is successfully performed (Figure 1.1).

In the last decades, the progress in miniaturised electronics and Micro Electro Mechanical Systems (MEMS) led to the design of new classes of satellites, smaller



© ESA/D.Ducros - 2007

Figure 1.1: European Space Agency (ESA) Autonomous Transfer Vehicle (ATV) autonomously docking to the ISS [3].

than the classical ones, to reduce launching costs. CubeSats belong to the class of nano-satellites, and were first introduced to allow students around the world to have a real hands-on approach to satellite design [4]. A standardization process of form factors, interfaces, and deployment systems reduced the costs of in-orbit demonstration missions, and CubeSats established around the world as the most common class of small satellites. Their standard unit (1U) is $10 \times 10 \times 10$ cm, with a maximum mass of 1.33 kg, and satellites ranging from 1U to 12U have been used in demonstration flights [5].

However, the difference in size between CubeSats and spacecraft that perform RVD missions still implies technological challenges that engineers must solve. During the last phase of such missions, commonly referred to as Final Approach (FA), the chaser must translate and correct its orientation so that its docking port can connect to the target's one. Thus, all 6 Degrees of Freedom (DoF) which describe relative position and orientation must be controlled, requiring the so called "relative pose estimation". The accuracy required in this control task is determined by the dimension of the spacecraft: the smaller the satellite, the greater the impact of a nominal misalignment in achieving a successful docking.

The miniaturisation to CubeSat level of hardware components such as Reaction Wheel (RW) and Reaction Control System (RCS) has become available only in recent years. The achievable translation and pointing accuracy is promising and some demonstration flights showed that proximity operations and Formation Flying (FF) are now possible with CubeSats. However, none of the in-orbit demonstration missions performed until now has been accurate enough to fulfil autonomous docking requirements [6].

There are four possible ways to determine the relative position and/or orientation of the chaser with respect to the target :

1. **Carrier Phase Differential GPS (CDGPS)**: the two satellites are equipped with a communication system and share their Global Positioning System (GPS) phase measurement, which is then used to determine their relative position. No information on the relative orientation can be obtained with this technique. The latter was used in 2010 during the PRISMA mission which achieved autonomous FF between a small and a micro satellite [7].

2. **Radio Frequency**: the two satellites are equipped with radio emitters and receivers, and the signal exchanged is used to determine relative position and orientation jointly or separately. This system is still in use on board the Russian Soyuz and Progress vehicles when docking with the ISS [8].

3. **Radar**: the time of flight of a radar signal is measured to determine the range between two spacecraft. It is important to notice that this solution purely measures the range and not the relative position, which depend on the definition of a Reference Frame (RF). Consequently, such solution is usually discarded.

4. **Vision Based Navigation (VBN)**: optical devices, such as mono or stereo cameras, are used to observe fiducial markers and to determine relative position and orientation of spacecraft jointly or separately. This solution was in use on board the PRISMA mission together with the aforementioned CDGPS navigation system. This was activated in the final phases, when the two spacecraft where in close range [7].

The FA phase of the RVD mission is certainly the most delicate, in which the highest possible accuracy is needed in order to fulfil strict docking requirements in terms of relative position and orientation. Furthermore, in such phase both position and attitude variables must be controlled simultaneously: to maintain alignment after a target rotation, the chaser must both rotate and translate.

Focusing only on the FA phase of the mission, the VBN solution is surely the most appealing: modern cameras are extremely small, which is a fundamental requirement for CubeSats sensors, and the current literature shows how this solution yields the best state estimation accuracy [6].

The goal of this thesis will be to analyze the performance of a VBN algorithm for the FA phase of the mission.

1.1.2 Fiducial Markers

The term "fiducial marker" refers to edges, lines, or points which are as easy as possible to be detected with computer vision algorithms [9]. They can be divided in two classes:

- Passive fiducial markers can be easily applied to structures by means of stickers or special space paints. Their detection requires particular illumination conditions, implying the necessity of an active light source on the spacecraft which has to detect them. Passive markers do not involve any power consumption or electronics, making them suitable for long term solutions (Figure 1.2).
- Active fiducial markers implemented by means of active sources of light. This solution increases hardware complexity and implies power consumption, but is more robust to lighting conditions. In fact, environment light noise due to the Sun or to reflections can be in part filtered out regulating the wavelenght of the source of light and the exposure time of the optical device used to detect the markers.

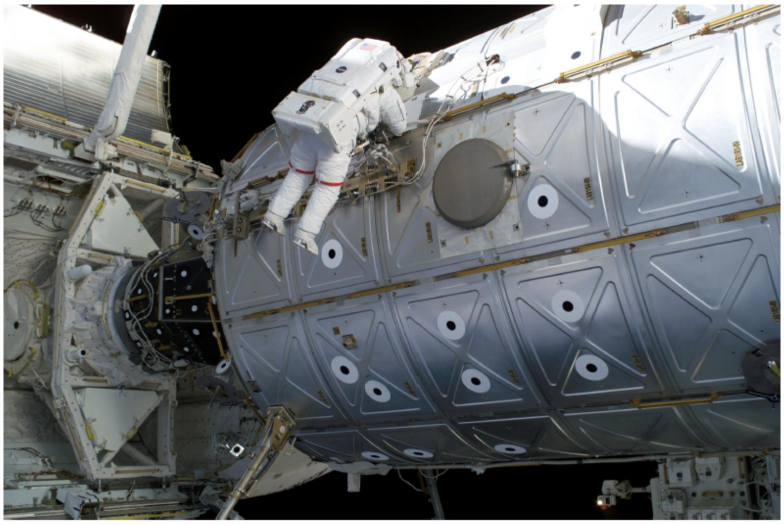


Figure 1.2: Passive fiducial markers on the surface of one of the ISS modules. Credit: NASA

Various VBN solutions have been already used in space for proximity operations or RVD missions. The optimal passive fiducial marker is the concentric contrasting circle, in use on the ISS and specifically designed for lighting environments of space [10]: two circles of contrasting color are centered on the same point (Figure 1.3).

Thanks to the concentricity, the circles’ centre position is not affected by rotations and translations, allowing its detection also under misalignment conditions.

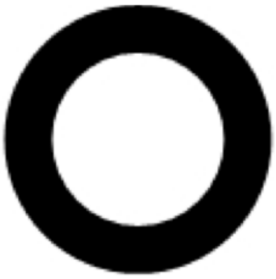


Figure 1.3: Example of a concentric contrasting circle [10]

Furthermore, also the area ratio between inner circle and outer disk remains constant, allowing detection algorithms to distinguish different circles belonging to the same pattern. In fact, at least four coplanar points are needed to fully estimate the 6 DoF, so a detection algorithm must be able to recognize and label different markers [11]. The use of this type of markers has been tested for nano-satellites relative navigation on SPHERES docking experiments on-board the ISS (Figure 1.4). SPHERES are nano-satellites developed by the Massachusetts Institute of Technologies (MIT) for didactic purpose. The tests presented in [9], even though the lighting conditions inside the ISS are not representative of an in-orbit demonstration, showed the accuracy that can be achieved for nano-satellites relative navigation. A drawback associated to the use of this type of markers is the high computational load required by the features detection algorithm.

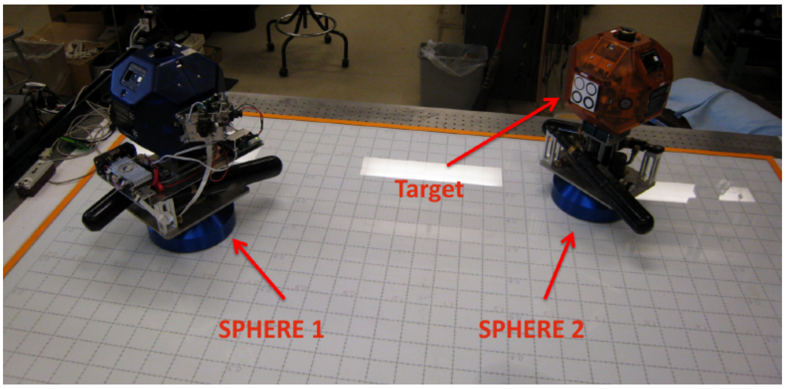


Figure 1.4: Relative pose estimation using concentric contrasting circles on SPHERES nano-satellites [9]

Modern Light Emitting Diodes (LED) are small and have a low power consumption. As a consequence the use of LEDs as active fiducial markers becomes an appealing solution for CubeSats applications. Sansone et al. [12] proposed a two patterns solution: a larger one, to be used from a relative distance of 1 m to 0.4 m one, and a smaller one, to be used from 0.4 m to docking. Such solution is designed to fit on a 2U CubeSat panel and allows a cooperation between the two satellites, which are both equipped with camera and patterns, and can improve the performance by sharing their respective pose estimation (Figure 1.5).

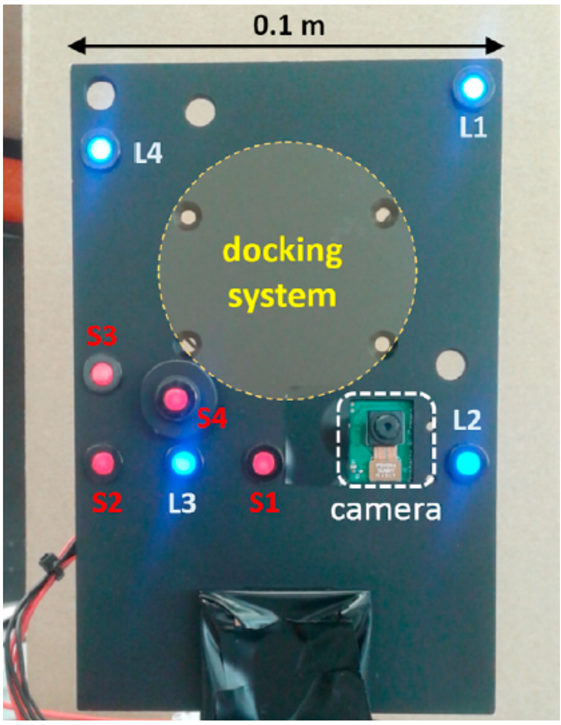


Figure 1.5: LEDs and camera setup on a 2U CubeSat panel [12]

The pose estimation is performed by means of the perspective 3-point algorithm [13]. The latter is efficient but computationally expensive, and the computed solution must still be filtered because of the intrinsic noise associated to vision sensors signal. Furthermore, the handover between the two patterns occurs when the two CubeSats are too close to each other, implying risks of collisions in case of failure during the transition.

A complete and exhaustive research has been performed by Pirat [6] and suggest the use of an internal 4-LEDs cross-shaped pattern, with a fifth out of plane LED at the center of the cross (Figure 1.6). This pattern is used from 5 m range to docking. Even though it is tested for 6U CubeSats, it is also designed to fit on 1U panels, with both the docking mechanism and metrology system contained within a 0.5U volume. This solution yields a set of nonlinear equations which relate the LEDs observation to the relative pose of the camera, implying the existence of an analytical solution to the problem. Such equations can be directly used in a navigation filter so to simultaneously perform relative pose estimation and filtering.

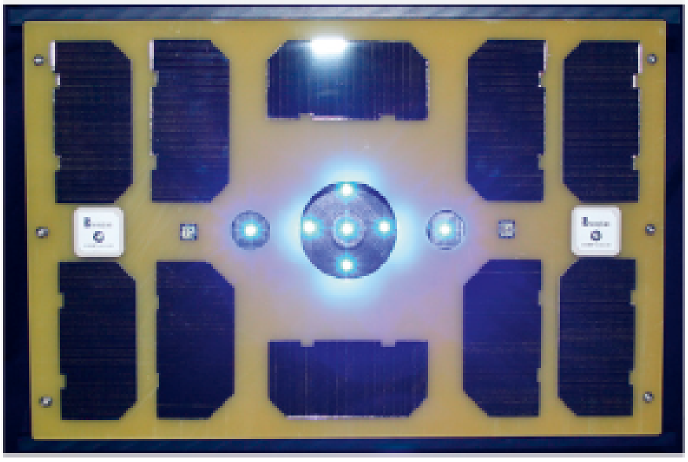


Figure 1.6: Pirat's LEDs setup on a 6U CubeSat panel [6]

In this thesis it was decided to follow the solution proposed by Pirat [6], which is efficient for pose estimation and filtering, and allows handover at a safe range for collision avoidance. Three different patterns are tested: the first one is centered on the target's docking face, assuming the integrated docking mechanism designed by Pirat is used; the second and third are placed on a corner of the docking face, allowing hardware integration also with a generic docking mechanism, and differ with respect to some pattern parameters. Moving the pattern to a corner of the target requires also moving the camera to a corner of the chaser, in order to optimize the field of view capabilities. The third pattern configuration also assumes a camera placed in a cavity a few centimeters from the docking plane, providing impacts protection without compromising performance.

1.1.3 Image Processing and State Estimation

The image processing algorithm in a VBN solution is of paramount importance because it intrinsically carries a computational burden. First of all, a features detection step is required: once a series of pictures is taken, fiducial markers have to be recognized, labeled correctly, and tracked from picture to picture.

A commonly used algorithm in computer vision applications is the Scale-Invariant Feature Transform (SIFT) detector: blurred copies of the images are generated in multiple orientation and scales to detect and then track features invariant to these transformations [14]. Other more advanced solutions, like the Random Sample

Consensus (RANSAC) one or algorithms which use the Hough Transform, can robustly detect and track specific shapes [11][15].

However, the computational load of this complex algorithms is high, and is only convenient when specific shapes must be detected, such as concentric contrasting circles. LEDs in a picture can be interpreted as a group of pixels, that can be detected by the Blob analysis algorithm of the MATLAB ® Computer Vision System toolbox. This solution is appealing because it keeps the computational load at the lowest possible level [6].

Once the fiducial markers are detected, the measurement of their position on the image plane must be used to estimate the relative pose of the camera. A common approach, chosen by [12], is to solve the perspective 3-point problem: 3 observed points provide up to four estimations of relative position and orientation and a fourth point is then used to disambiguate the solution [13]. The obtained estimation is then fed to a navigation filter to account for the noise associated with vision sensors. The process of state estimation and filtering is thus performed in two steps.

However, if the used pattern implies the existence of an analytical solution, the features' position on the image plane can be directly fed to a classical Extended Kalman Filter (EKF) [6]. The latter, which also contains a model of the dynamics involved, can yield a simultaneous estimation and filtering of the relative pose. Furthermore, such filtering technique is reliable because it has been used for decades and its tuning process is simple.

1.1.4 Control

Classically, spacecraft control systems are designed with two different controllers for position and attitude. This solution is effective, for example, when the states to be controlled are absolute position and attitude of a satellite in an orbital trajectory. Being position and attitude decoupled, two dynamic models are derived to design independent control laws, also maintaining simpler mathematical models.

The most common solution for attitude control is the implementation of a Quaternion Feedback Regulator, which avoids singularities associated to angle-based description of the attitude of a spacecraft [16]. For the position control, a Linear Quadratic Regulator (LQR) is the most simple yet effective solution. It is easy to tune and it performs great when the nonlinearities of the model are small. Instead, advanced solutions such as $\mathcal{H}_\infty$ or μ-synthesis controllers are more robust to disturbances and nonlinearities, and can be used both for position and attitude control [17][18].

As previously stated, during the FA, relative position and attitude of the two satellites are coupled. In order to achieve the required control accuracy, the chaser's 6 DoF must be controlled simultaneously, implying the necessity of a single controller. The scope of this research has not been optimal or robust control achievements. So, to maintain the controller tuning simple, it was decided to implement an LQR control law.

1.2 Thesis Overview

In this thesis, a simulator of CubeSat docking missions is designed. A 3U chaser CubeSat, equipped with a camera, approaches a target CubeSat, equipped with a cross-shaped pattern of LEDs. Three different patterns, different in size and location on the docking face, are studied. The target travels on a circular Low Earth Orbit (LEO) and is considered perfectly aligned to its local orbital frame. The mission starts from a first Station Keeping point (SK0), placed 5 meters from the target, and ends when the relative distance between the two CubeSats is reduced to 0.05 meters. SK0 represents the point in which the handover between the Guidance Navigation and Control system that previously operated the chaser and the VBN system occurs. The VBN algorithm is mimicked by generating LEDs pictures based on the simulated relative state. The LEDs position in pixels is then corrupted to account for the optical device noise and LEDs detection error. These noisy positions are fed to an EKF which, using a linearized version of the coupled dynamics, performs simultaneous estimation and filtering of the state variables. The estimated state is used by an LQRr to generate the position and attitude control inputs for the chaser. These inputs are corrupted with realistic sources of disturbances and then returned to the chaser, thus closing the loop.

The objective of this research is to evaluate the performance of the proposed VBN algorithm during Final Approach phase of the mission. Such performance is studied on the basis of three different aspects.

1. First of all, given requirements on the maximum overshoot and on the accuracy required at docking, the full FA mission is simulated for all three patterns. For each pattern, the time required to complete the mission and the Root Mean Square (RMS) estimation errors are evaluated to compare the performance of the considered solutions.

2. Then, the effects of the handover precision on the VBN system initialization process is considered. The Guidance Navigation and Control (GNC) system that operates the chaser to the point in which the optical sensor starts working cannot be infinitely precise: its accuracy determines the feasibility of the handover. A Monte Carlo analysis approach is used to understand the effects of uncorrelated and randomly generated initial conditions on the initialization time.

3. Finally, the effects of the error associated with the LEDs detection algorithm is studied to understand the role it plays during the Final Approach. Intuitively, this error will mostly affect the state estimation when the LEDs are seen very close to each other on the image plane. This happens when the two satellites are far apart, so during the initialization of the VBN system. To understand the role of this error, modeled as a Gaussian white noise, the Monte Carlo analysis approach is again used to generate random values for the standard deviation associated to it. The stabilization time is analyzed to understand which pattern is more robust to an increase in the detection error.

The thesis is organised in six chapters. Chapter 2 provides descriptions of the reference frame defined for the study, of the mathematical model of the coupled position and attitude dynamics involved, and of the disturbances the simulator accounts for. In Chapter 3 a thorough description of the VBN algorithm is presented, together with a detailed description of the three patterns tested. Chapter 4 describes the estimation and control techniques adopted for the study. The handover problem is also discussed. Chapter 5 reports a description of the parameters used in the simulator and the simulation results obtained in terms of handover performance, pattern performance and pixel detection error. The final chapter provides the conclusions and a description of the possible directions that can be taken for related future works.

Chapter 2

Definitions and Mathematical Model

To achieve the docking conditions which satisfy the accuracy requirements of the final approach phase, the chaser CubeSat has to be controlled with respect to its docking port. For this reason, it is convenient to define the so called Port-To-Port (P2P) coupled dynamics, that is a mathematical model of the dynamics of the chaser docking port with respect to the target's one.

This chapter is devoted to the formulation of the mathematical background required to understand the coupled relative position and attitude dynamics. In the first sections the mathematical notation used, the Reference Frames (RF) required to define the docking phase and an overview of the CubeSat structure are presented. Then, the relative dynamics is derived and the state space model of the system is described followed by an explanation of the actuators' models implemented. Finally, the disturbances that have been considered in the research are defined.

2.1 Notation

2.1.1 Vector

In this research, vectors will be expressed using the following notation:

- the name of the vector will be in **bold**;
- the superscript indicates what the vector refers to;
- the subscript indicates the RF in which the vector is expressed.

Example:

$$\boldsymbol{r}_i^t = \quad \text{position vector of object } t \text{ in frame } i$$

2.1.2 Direction Cosine Matrix

A Direction Cosine Matrix (DCM) is a linear algebra tool belonging to an orthogonal space of dimension three. Such matrices, which belong to the class of rotation matrices, are needed to map a vector from a frame to another. In other words, if the position vector of object t is expressed in frame i, but we need it expressed in frame j, we can use the DCM defined as A_{ji} and obtain the required vector by performing the following matrix operation

$$\boldsymbol{r}_j^t = A_{ji}\boldsymbol{r}_i^t \tag{2.1}$$

Any DCM matrix belongs to a space of dimension three, and can thus be obtained by means of a multiplication of three different matrices. As a consequence, three fundamental matrices describing each rotation around a specific axis of a reference frame can be defined

$$R_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \quad \Rightarrow \quad \text{rotation around } x \tag{2.2a}$$

$$R_2(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad \Rightarrow \quad \text{rotation around } y \tag{2.2b}$$

$$R_3(\psi) = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \Rightarrow \quad \text{rotation around } z \tag{2.2c}$$

In other words, if we consider a first RF i and a second RF j obtained by means of a positive rotation of an angle ψ around the z axis of the first frame, a vector in i is mapped in j by means of the operation

$$\boldsymbol{r}_j^t = R_3(\psi)\boldsymbol{r}_i^t$$

Given the DCM A_{ji}, the matrix which performs the opposite operation, namely mapping a vector from frame j to frame i, is given by its inverse. Rotation matrices are orthonormal, i.e., they are orthogonal and have determinant equal to 1. The most important property of orthonormal matrices is that their inverse is equal to their transpose. As a consequence, the following holds

$$A_{ij} = \left(A_{ji}\right)^{-1} = \left(A_{ji}\right)^{T} \quad \Rightarrow \quad A_{ij} = A_{ji}^T \tag{2.3}$$

2.1.3 Angular Velocity

Angular velocities will be expressed with letter $\boldsymbol{\omega}$ and, to be rigorously defined, require the definition of three reference frames:

1. the reference frame which is rotating;

2. the reference frame with respect to which the rotation is described;

3. the reference frame in which such vector is expressed.

Example:

$\boldsymbol{\omega}_a^{ab} =$ angular velocity of RF a with respect to RF b, expressed in RF a

Angular velocities can be also referred to as rotation rates.

2.2 Reference Frames

The reference frames required to describe the dynamics involved during the final approach phase of the docking mission are five and are briefly described in this section.

2.2.1 Earth-Centred Inertial Frame

To describe the dynamics of a system, an inertial frame is required. To be correct, since the Earth rotates around the Sun, a reference frame centred on our planet is not exactly an inertial frame. However, for what regards LEO dynamics, it is a perfectly valid approximation.

The Earth-Centred Inertial (ECI) frame $\mathcal{F}_I$ will be used as a starting point in the definition of the dynamics (Figure 2.1). It is centred in the centre of Earth, which is assumed perfectly spherical. The $\hat{X}_I$ axis lays on the equatorial plane and points toward the vernal equinox, the $\hat{Z}_I$ axis points towards the north pole, and the $\hat{Y}_I$ axis completes the right-handed triad [8].

2.2.2 Local-Vertical Local-Horizontal Frame

The orbital frame is often referred to as Local-Vertical Local-Horizontal (LVLH) frame, $\mathcal{F}_O$. It is the fundamental frame needed to describe the relative motion of two satellites during an RVD mission (Figure 2.2). Such RF is centred on the target CubeSat Centre of Mass (CoM). The $\hat{z}_o$ axis, commonly referred to as $\bar{R}$, goes in a radial direction from the CoM of the satellite to the centre of Earth; the $\hat{y}_o$ axis, commonly referred to as $\bar{H}$, points in the opposite direction with respect to the angular momentum vector of the orbit; the $\hat{x}_o$ axis, commonly referred to as $\bar{V}$, completes the right-handed triad and is in the direction of the orbital velocity vector, even though it might not be always aligned with it [8].

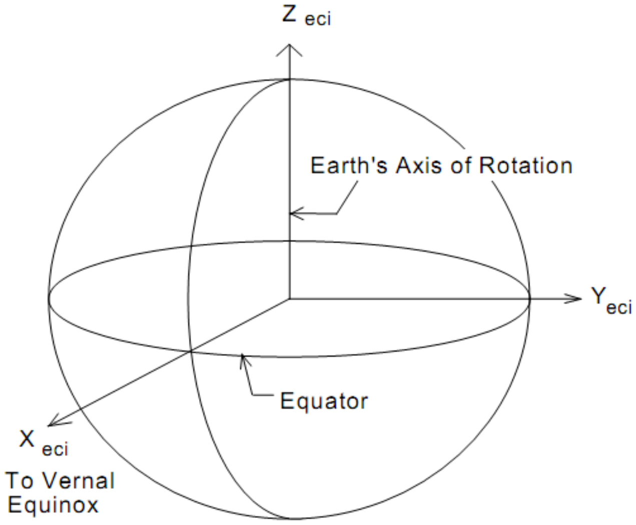


Figure 2.1: Earth-Centred Inertial reference frame [19]

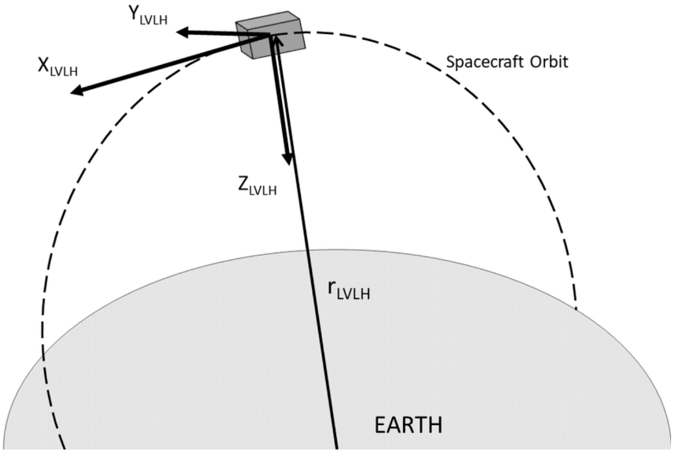


Figure 2.2: Local-Vertical Local-Horizontal reference frame [20]

2.2.3 Body Frame

The body frame $\mathcal{F}_b$ of a satellite is needed to describe its attitude dynamics with respect to the orbital frame. It is in fact important to notice that $\mathcal{F}_b$ might be rotating with respect to $\mathcal{F}_\mathcal{O}$ if for example the satellite is spinning with respect to its $\bar{V}$ axis.

$\mathcal{F}_b$ is fixed to the satellite and centred on its CoM. Even though the satellite's CoM position is not fixed because of the fuel consumption throughout the mission, torque and force misallignments due to such displacement can be considered as disturbances, allowing to consider the CoM as fixed. In order to simplify some of the math that will be presented in the following sections of this chapter, the body frame is considered aligned and oriented as the docking frame (Figure 2.3). Subscripts t and c will be used to distinguish between target's and chaser's specific frames.

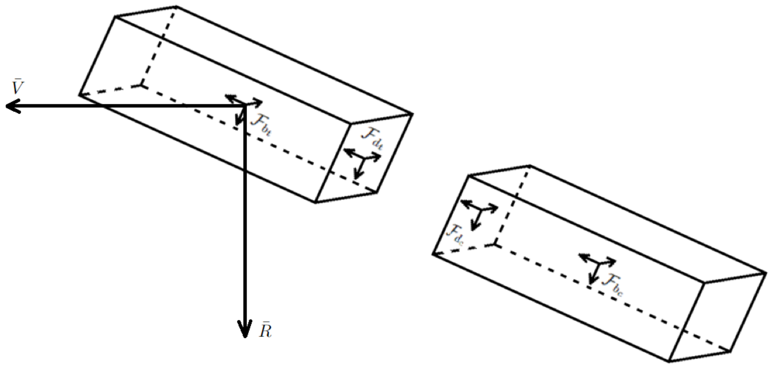


Figure 2.3: Body and docking frame of target and chaser, represented in the LVLH reference frame

2.2.4 Docking Frame

To properly define the Port To Port (P2P) coupled dynamics, the docking reference frames $\mathcal{F}_d$ of both target and chaser are required. $\mathcal{F}_d$ is centred in the centre of the docking mechanism; the chaser's $\hat{x}_{d_c}$ axis is normal to the docking face and points outside the satellite, so in the direction of approach to the target. For what regards $\hat{x}_{d_t}$, it is as well normal to the docking face but points inside the target, so that when docking is completed $\hat{x}_{d_c}$ and $\hat{x}_{d_t}$ are superimposed (Figure 2.3). The $\hat{z}_d$ axes point in the direction of the CubeSat's side which is arbitrarily established to be the bottom side of the satellite, such that, at docking, $\hat{z}_{d_c}$ and $\hat{z}_{d_t}$ are superimposed. The $\hat{y}_d$ axes complete the right-handed triads.

2.2.5 Navigation Frame

The navigation frame $\mathcal{F}_n$, one for each of the satellites, is the one required by the Vision-Based Navigation system, i.e., the sensor suit which allows relative navigation. For the chaser, $\mathcal{F}_{n_c}$ is centred on the focal plane of the optical device the satellite is equipped with, and it is oriented as the docking frame. For the target, $\mathcal{F}_{n_t}$ is centred in a point which allows a convenient definition of the fiducial markers' positions on the docking face. As for the chaser, it is as well oriented as the docking frame (See section 3.2).

It is convenient to define the CubeSat reference frames oriented in the same way ($\mathcal{F}_b$, $\mathcal{F}_d$ and $\mathcal{F}_n$) so that the rotation matrices needed to map vectors from one frame to the other become identity matrices, thus simplifying the math

$$A_{db} = A_{nd} = \mathbb{1}_3 \tag{2.4}$$

The navigation frame implies a constant vector describing its position with respect to the docking frame: $\boldsymbol{r}_{d_t}^{n_t d_t}$ and $\boldsymbol{r}_{d_c}^{n_c d_c}$.

2.3 CubeSat Structure

In literature, many different classifications by weight of satellites can be found. Here, the definitions used by [21] are considered (Figure 2.4).

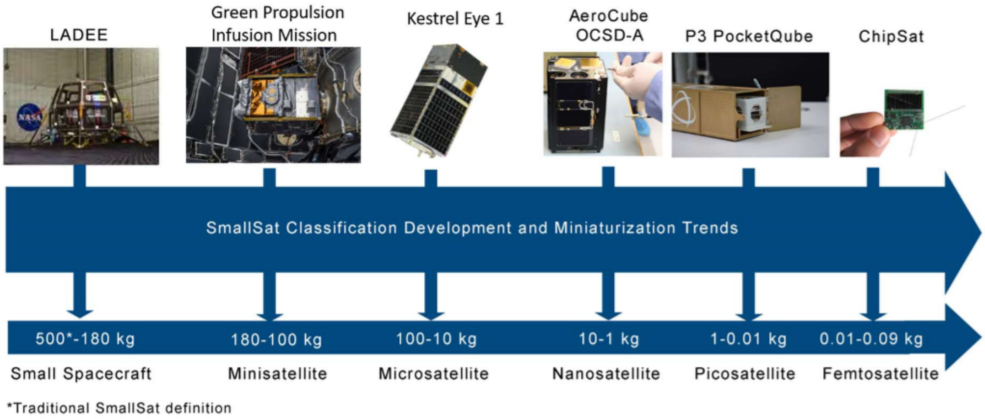


Figure 2.4: Satellites classification by weight [21]

Light and small satellites, defined as those spacecraft weighing less then 180 kg [21], became of extreme interest in the last few decades. In fact, the lighter is the satellite, the cheaper is the launch, and the miniaturization process of space technologies of recent years allows the design of light yet technologically capable satellites.

Among the SmallSat family, the place of honor is held by CubeSats. These satellites where first introduced for academic purposes in the USA [4], but the definition of their specific form factor led to the establishment of standards which helped reducing the design cost and increased the interest for CubeSats worldwide. The standard form factor for CubeSats is a 10 cm cube, referred to as 1U, with a mass up to 1.33 kg [5]. Multiple units are usually combined, and nowadays

CubeSat ranging from 1U to 12U are used for In-Orbit Demonstration (IOD) missions (Figure 2.5).

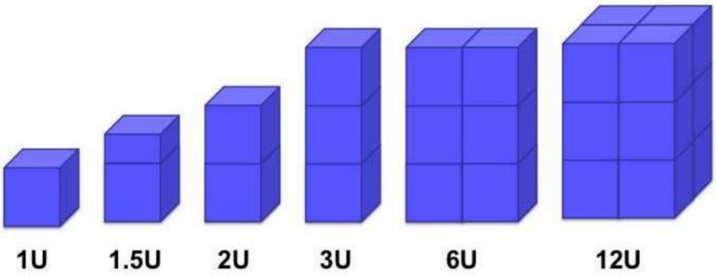


Figure 2.5: CubeSat family [21]

For this thesis, two 4 kg 3U+ CubeSats are considered (Figure 2.6).

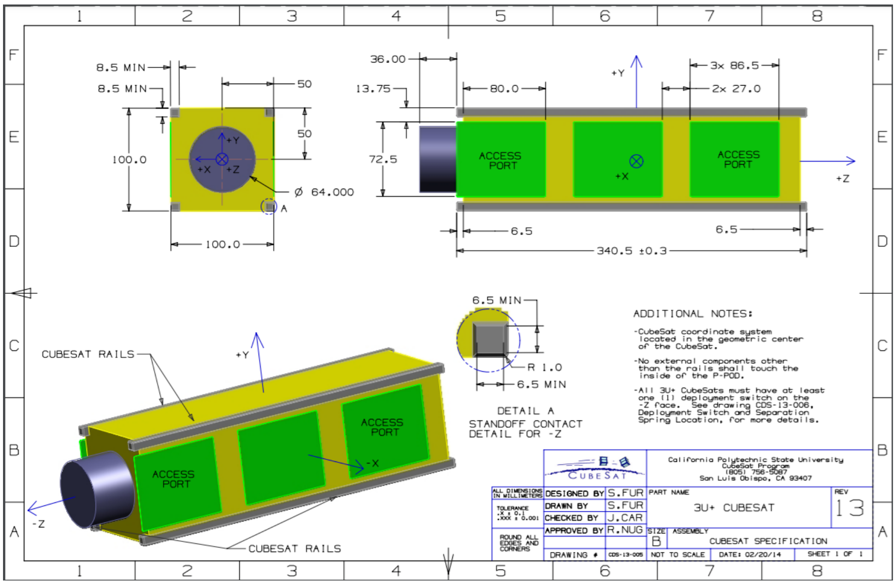


Figure 2.6: 3U CubeSat specifications [5]. The axes convention used in this scheme is different with respect to the one used in the thesis

The docking mechanism is assumed to be placed in correspondence of the so called "tuna-can" that can be seen in Figure 2.6, thus defining the position of reference frame $\mathcal{F}_d$. For simplicity, the CoM of the satellite is assumed to be placed in correspondence of the geometrical centre of the structure, thus defining the position of reference frame $\mathcal{F}_b$.

We can now define two vectors, which describe the position of the docking frame with respect to the body frame for the two CubeSats

$$\boldsymbol{r}_{b_c}^{d_c} = \begin{bmatrix} 0.15 \\ 0 \\ 0 \end{bmatrix} \text{m} \tag{2.5a}$$

$$\boldsymbol{r}_{b_t}^{d_t} = \begin{bmatrix} -0.15 \\ 0 \\ 0 \end{bmatrix} \text{m} \tag{2.5b}$$

2.4 Dynamics

The derivation of the dynamics which describes the relative motion between the docking ports of the two CubeSats has been inspired by the PhD thesis of Dr. Pirat. Such dynamics will be referred to as Port-To-Port (P2P) coupled dynamics, and it gives the best possible description of the coupling involved during the Final Approach phase. The intrinsic complexity of the coupling phenomena can be grasped thinking about how a target rotation implies both a chaser rotation and a chaser translation (Figure 2.7). The next two sections describe the main steps required for the derivation of this complicate model, but some of the math is skipped for sake of brevity. The full and detailed derivation of the model can be found in Pirat's thesis, available online [6].

2.4.1 Relative Attitude Dynamics

To derive the relative attitude dynamics it is first required to define the absolute attitude of a spacecraft, namely its orientation with respect to its local orbital frame $\mathcal{F}_{\mathcal{O}}$. To do so, it is convenient to start from the well-known Euler's Equations, obtained from the time derivative of the angular momentum in the body frame of the spacecraft

$$\dot{\boldsymbol{\omega}}_b^{bI} = I_b^{-1}\Big[\boldsymbol{T}_b - \boldsymbol{\omega}_b^{bI} \times I_b \boldsymbol{\omega}_b^{bI}\Big] \tag{2.6}$$

where

- $\boldsymbol{\omega}_b^{bI}$ is the angular velocity of the body frame with respect to the inertia frame, expressed in the body frame;
- I_b is the inertia tensor of the spacecraft, expressed in the body frame (in which it is constant);

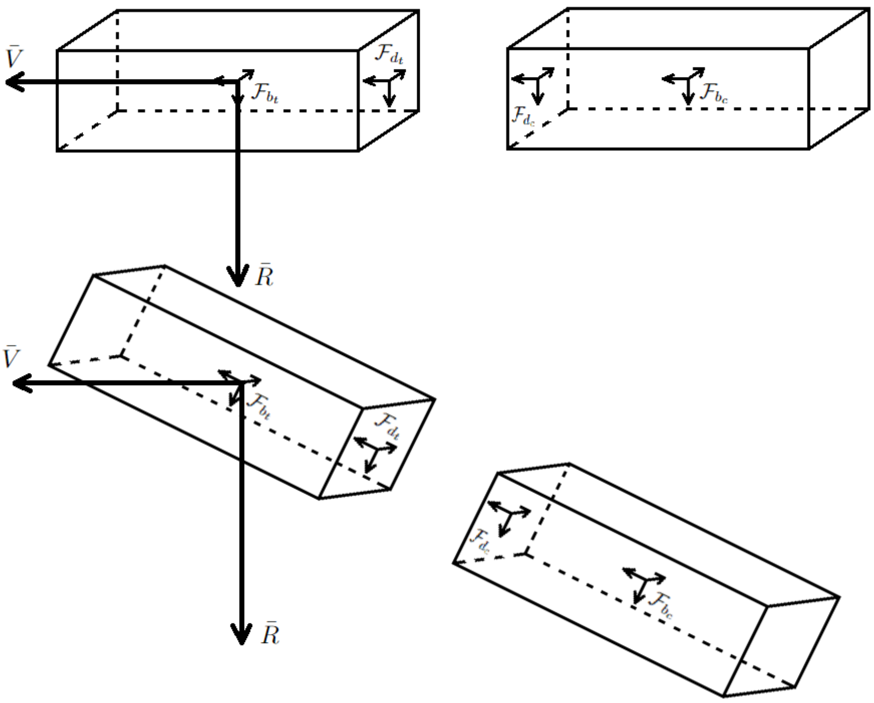


Figure 2.7: In order to maintain the alignment, to a target rotation around the Centre of Mass must correspond both a chaser rotation and a chaser translation

- $\boldsymbol{T}_b$ is the vector of input torques, coming from both actuators and disturbances, expressed in the body frame of the spacecraft;
- operator × represents the cross product, and it can be also rewritten by means of a skew symmetric matrix:

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}\times]\boldsymbol{b}, \qquad [\boldsymbol{a}\times] = \underbrace{\begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}}_{=\text{ skew symmetric matrix of } \boldsymbol{a}} \tag{2.7}$$

For the attitude control of the spacecraft, Reaction Wheels (RWs) will be used. Their angular momentum have to be added to the total angular momentum of the satellite, which becomes

$$\boldsymbol{H}_b = I_b \boldsymbol{\omega}_b^{bI} + \boldsymbol{H}_b^{RW} \tag{2.8}$$

where letter $\boldsymbol{H}$ is used to refer to angular momenta vectors. This leads to a new form of (2.6)

$$\dot{\boldsymbol{\omega}}_b^{bI} = I_b^{-1}\left[\boldsymbol{T}_b - \boldsymbol{T}_b^{RW} - \boldsymbol{\omega}_b^{bI} \times \left(I_b \boldsymbol{\omega}_b^{bI} + \boldsymbol{H}_b^{RW}\right)\right] \tag{2.9}$$

where the term $\boldsymbol{\omega}_b^{bI} \times \boldsymbol{H}_b^{RW}$ represents the gyroscopic coupling between reaction wheels and the satellite angular velocity. Since in this research the Vision-Based Navigation algorithm, described in Chapter 3, is evaluated in its most general form, so to have a navigation filter independent of system design choices, such coupling is accounted for in the RWs model rather than in the attitude dynamics.

The satellite angular velocities in (2.9) are expressed with respect to the inertial frame. However, since a satellite, during an orbital mission, is nominally aligned to its LVLH frame $\mathcal{F}_O$, it is convenient to parametrize such velocities with respect to orbital quantities by means of the DCM which maps $\mathcal{F}_O$ in $\mathcal{F}_b$

$$A_{bo} = A_{bI} A_{Io} \tag{2.10}$$

Differentiating the latter equation and performing some math leads to an important relation

$$\boldsymbol{\omega}_b^{bo} = \boldsymbol{\omega}_b^{bI} - A_{bo} \boldsymbol{\omega}_o^{oI} \tag{2.11}$$

where the DCM A_{bo} contains the attitude dynamics of the spacecraft and $\boldsymbol{\omega}_o^{oI}$ describes the angular velocity of the orbital frame with respect to the inertial one, expressed in the orbital frame itself. In this research we are considering anLEO trajectory, which can be safely approximated with a uniform circular motion

$$\boldsymbol{\omega}_o^{oI} = \begin{bmatrix} 0 \\ -\omega_O \\ 0 \end{bmatrix}, \qquad \omega_O = \sqrt{\frac{\mu_E}{r_O^3}} = \text{ orbital mean motion} \tag{2.12}$$

where μ_E is the standard Earth gravitational parameter and is obtained from the product between the universal gravitational constant G and the Earth's mass M_E. r_O, instead, is the orbit radius with respect to the Earth's centre. The gravitational parameter μ_E can be computed with high accuracy from laser distance measurements of artificial Earth satellites [22].

Substituting (2.11) in (2.6) leads to

$$\dot{\boldsymbol{\omega}}_b^{bo} = I_b^{-1} \left[\boldsymbol{T}_b - \left(\boldsymbol{\omega}_b^{bo} + A_{bo} \boldsymbol{\omega}_o^{oI} \right) \times \left(I_b (\boldsymbol{\omega}_b^{bo} + A_{bo} \boldsymbol{\omega}_o^{oI}) \right) \right] \tag{2.13}$$

To complete the model, the kinematic relation between attitude variables and angular velocities is required. The VBN algorithm is based on the Euler Angles. To have a single type of an attitude variable for all the parts implemented in this thesis, also the kinematics and the dynamics will be Euler Angles based. The 1-2-3 sequence, convenient for the VBN system, is chosen (see Section 3.1.2). In general, to refer to the attitude angles, the following vector is defined

$$\boldsymbol{\alpha} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{2.14}$$

such that the associated 1-2-3 DCM is obtained by means of the following combination of the fundamental rotation matrices defined in (2.2)

$$R_{123}(\boldsymbol{\alpha}) = R_3(\psi)R_2(\theta)R_1(\phi) \tag{2.15}$$

The kinematics associated to a specific Euler sequence is thoroughly described in [23], and the 1-2-3 sequence for the body absolute attitude problem leads to

$$\dot{\boldsymbol{\alpha}}^{bo} = B_{123}(\theta, \psi)\boldsymbol{\omega}_b^{bo} \tag{2.16}$$

with

$$B_{123}(\theta, \psi) = \frac{1}{\cos\theta}\begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \cos\theta\sin\psi & \cos\theta\cos\psi & 0 \\ -\sin\theta\cos\psi & \sin\theta\sin\psi & \cos\theta \end{bmatrix} \tag{2.17}$$

Combining kinematics and Euler's Equations (Equations (2.16) and (2.13)), the absolute attitude problem can now be rewritten as a nonlinear matricial function:

$$\begin{bmatrix} \dot{\boldsymbol{\alpha}}^{bo} \\ \dot{\boldsymbol{\omega}}_b^{bo} \end{bmatrix} = \boldsymbol{f}\Big(\boldsymbol{\alpha}^{bo}, \boldsymbol{\omega}_b^{bo}, \boldsymbol{T}_b\Big) \tag{2.18}$$

This model is required for the absolute attitude dynamics of the two CubeSats, which however is not accounted for in this research. In fact, the target is assumed to be perfectly aligned to its local orbital frame, situation described by the condition

$$\boldsymbol{\alpha}^{bo} = \boldsymbol{\omega}_b^{bo} = \boldsymbol{T}_b = \boldsymbol{0} \tag{2.19}$$

which is the result of an infinitely precise attitude control system. The chaser, instead, is controlled by means of the P2P coupled dynamics that will be derived now, which is an extension of the model described by (2.18). In fact, in the most general Final Approach case, it is pointless to control the chaser with respect to its local orbital frame by means of an absolute attitude model. If the target is not aligned to the same orbit, the docking conditions cannot be met. The situation considered in this thesis is simplified: the conditions described in (2.19) imply that the chaser is actually controlled to be perfectly aligned to the target local orbital frame. However, defining a general mathematical model will allow the extension of the research, in future works, to a situation in which also the target is being controlled.

The first step to define the P2P attitude dynamics is to modify the absolute attitude dynamics described by (2.13) so to express it with respect to the docking frame of the spacecraft. Recalling the vectors defined in (2.5a) and (2.5b), Steiner's theorem can be used to express the inertia tensor in the docking frame

$$I_d = A_{db}\bigg(I_b + m\Big[||\boldsymbol{r}_b^d||^2\mathbb{1}_3 - \boldsymbol{r}_b^d\boldsymbol{r}_b^{d^T}\Big]\bigg)A_{db}^T \tag{2.20}$$

where m is the mass of the spacecraft. The absolute attitude dynamics is now expressed with respect to a new reference frame

$$\dot{\boldsymbol{\omega}}_d^{do} = I_d^{-1}\Big[\boldsymbol{T}_d - \left(\boldsymbol{\omega}_d^{do} + A_{do}\boldsymbol{\omega}_o^{oI}\right) \times \left(I_d(\boldsymbol{\omega}_d^{do} + A_{do}\boldsymbol{\omega}_o^{oI})\right)\Big] \tag{2.21}$$

where $A_{do} = A_{db}A_{bo}$ describes the absolute attitude with respect to the docking port. Equation (2.21) describes a sort of absolute spacecraft dynamics, but with respect to the docking port of the satellite.

To describe the relative attitude between the two satellites, the associated rotation matrix must be defined: $A_{d_cd_t}$. The P2P attitude dynamics can, in principle, be expressed in different ways. However, our goal is to control the chaser with respect to the target, and not the other way around. It is convenient to consider the rotation of $\mathcal{F}_{d_c}$ with respect to $\mathcal{F}_{d_t}$ by means of the dynamics described by $\dot{\boldsymbol{\omega}}_{d_c}^{d_cd_t}$. The relative angular velocity can be expressed as

$$\boldsymbol{\omega}_{d_c}^{d_cd_t} = \boldsymbol{\omega}_{d_c}^{d_co} - A_{d_cd_t}\boldsymbol{\omega}_{d_t}^{d_to} \tag{2.22}$$

A time differentiation of the latter leads to a raw version of the coupled dynamics:

$$\dot{\boldsymbol{\omega}}_{d_c}^{d_cd_t} = \dot{\boldsymbol{\omega}}_{d_c}^{d_co} - A_{d_cd_t}\dot{\boldsymbol{\omega}}_{d_t}^{d_to} + [\boldsymbol{\omega}_{d_c}^{d_cd_t}\times](A_{d_cd_t}\boldsymbol{\omega}_{d_t}^{d_to}) \tag{2.23}$$

where $\dot{\boldsymbol{\omega}}_{d_t}^{d_to}$ and $\dot{\boldsymbol{\omega}}_{d_c}^{d_co}$ can be obtained from (2.21).

So far, three attitude dynamics, coupled between them, have been written: Equation (2.21) gives the absolute dynamics for the two satellites, while Equation (2.23) gives the relative dynamics. Recalling that our goal is to control the chaser only with respect to the target, and not with respect to an orbit, it is convenient to express its absolute attitude with respect to the target and the relative ones. Writing $A_{d_co} = A_{d_cd_t}A_{d_to}$ and using (2.22) to rewrite $\boldsymbol{\omega}_{d_c}^{d_co}$ leads to

$$\begin{aligned}\dot{\boldsymbol{\omega}}_{d_c}^{d_co} = I_{d_c}^{-1}\Bigg(\boldsymbol{T}_{d_c} - &\left(\boldsymbol{\omega}_{d_c}^{d_cd_t} + A_{d_cd_t}\boldsymbol{\omega}_{d_t}^{d_to} + A_{d_cd_t}A_{d_to}\boldsymbol{\omega}_o^{oI}\right)\times \\ &\left(I_{d_c}\left(\boldsymbol{\omega}_{d_c}^{d_cd_t} + A_{d_cd_t}\boldsymbol{\omega}_{d_t}^{d_to} + A_{d_cd_t}A_{d_to}\boldsymbol{\omega}_o^{oI}\right)\right)\Bigg)\end{aligned} \tag{2.24}$$

Equation (2.23) can now be written only in terms of absolute attitude of the target and relative attitude between the two CubeSats, using (2.24) and the target version of (2.21).

The kinematics maintains the same form as in (2.16):

$$\dot{\boldsymbol{\alpha}}^{d_cd_t} = B_{123}(\theta^{d_cd_t}, \psi^{d_cd_t})\boldsymbol{\omega}_{d_c}^{d_cd_t} \tag{2.25a}$$

$$\dot{\boldsymbol{\alpha}}^{d_to} = B_{123}(\theta^{d_to}, \psi^{d_to})\boldsymbol{\omega}_{d_t}^{d_to} \tag{2.25b}$$

The relative attitude model can now be written in a nonlinear state-space form as:

$$\dot{\boldsymbol{x}}_{att} = \begin{bmatrix} \dot{\boldsymbol{\alpha}}^{d_c d_t} \\ \dot{\boldsymbol{\omega}}_{d_c}^{d_c d_t} \\ \dot{\boldsymbol{\alpha}}^{d_t o} \\ \dot{\boldsymbol{\omega}}_{d_t}^{d_t o} \end{bmatrix} = \boldsymbol{f}_{att}\Big(\boldsymbol{x}_{att}, \boldsymbol{u}_{att}\Big), \quad \boldsymbol{x}_{att} = \begin{bmatrix} \boldsymbol{\alpha}^{d_c d_t} \\ \boldsymbol{\omega}_{d_c}^{d_c d_t} \\ \boldsymbol{\alpha}^{d_t o} \\ \boldsymbol{\omega}_{d_t}^{d_t o} \end{bmatrix}, \quad \boldsymbol{u}_{att} = \begin{bmatrix} \boldsymbol{T}_{d_c} \\ \boldsymbol{T}_{d_t} \end{bmatrix} \tag{2.26}$$

where the relative attitude state vector $\boldsymbol{x}_{att}$ and the attitude control input vector $\boldsymbol{u}_{att}$ have been defined.

To obtain a state space model of the system, needed for example to derive control and observation laws, (2.26) must be linearized. However, the P2P position dynamics will be derived first.

2.4.2 Port-to-Port Coupled Dynamics

The P2P coupled dynamics is a combination of the relative attitude dynamics just derived and a modified version of the well known Hill's Equations, which describe the position of a spacecraft with respect to another one, expressed in the local orbital frame of the latter [8].

The P2P position, namely the position of the chaser's docking frame with respect to the target's one, can be expressed in the inertial frame as (Figure 2.8):

$$\begin{aligned} \boldsymbol{s}_I^{d_c d_t} &= r_I^{b_c} + A_{Ib_c}\boldsymbol{r}_{b_c}^{d_c} - r_I^{b_t} - A_{Ib_t}\boldsymbol{r}_{b_t}^{d_t} \\ &= s_I^{b_c b_t} + A_{Ib_c}\boldsymbol{r}_{b_c}^{d_c} - A_{Ib_t}\boldsymbol{r}_{b_t}^{d_t} \end{aligned} \tag{2.27}$$

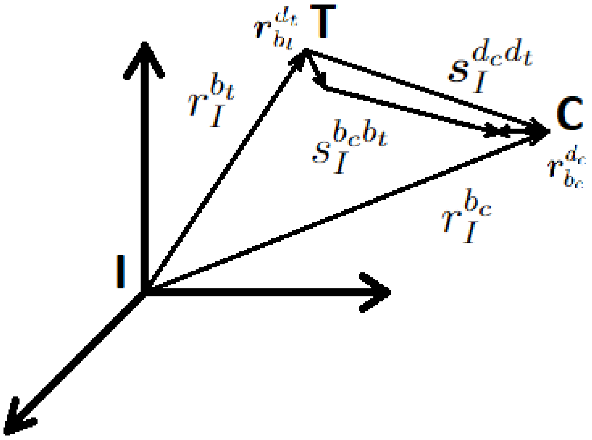


Figure 2.8: Position of the chaser's docking frame with respect to the target's one, expressed in the inertial frame

For the relative position, it is convenient to express quantities in the target docking frame, so (2.27) becomes

$$\boldsymbol{s}_{d_t}^{d_c d_t} = \underbrace{A_{d_t o} A_{oI} \boldsymbol{s}_I^{b_c b_t}}_{\boldsymbol{s}_{d_t}^{b_c b_t}} + A_{d_t d_c} \underbrace{A_{d_c b_c} \boldsymbol{r}_{b_c}^{d_c}}_{=\boldsymbol{r}_{d_c}^{d_c}} - \underbrace{A_{d_t b_t} \boldsymbol{r}_{b_t}^{d_t}}_{=\boldsymbol{r}_{d_t}^{d_t}} \tag{2.28}$$

where

- $A_{d_t o}$ is the target's absolute attitude in the docking frame, which can be obtained from the kinematics in (2.25b);
- $A_{d_t d_c}$ is the transpose of the relative attitude dynamics, which can be obtained from the kinematics in (2.25a);
- $r_{d_c}^{d_c}$ and $r_{d_t}^{d_t}$ are given by constant vectors (2.5a) and (2.5b) multiplied by constant matrices (2.4), and are thus constant;
- $A_o I$ is a parameter, depending only on the target's orbit.

Differentiating two times (2.28) and performing some substitutions, which are here skipped for brevity but can be found in [6], leads to the following equation:

$$\begin{aligned}
\ddot{\boldsymbol{s}}_{d_t}^{d_c d_t} = & - [\dot{\boldsymbol{\omega}}_{d_t}^{d_t o} \times] \overbrace{\underline{(\boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t})}}^{\boldsymbol{s}_{d_t}^{b_c b_t} \text{ from (2.28)}} - \\
& - [\boldsymbol{\omega}_{d_t}^{d_t o} \times][\boldsymbol{\omega}_{d_t}^{d_t o} \times] \underline{(\boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t})} - 2[\boldsymbol{\omega}_{d_t}^{d_t o} \times] \dot{\boldsymbol{s}}_{d_t}^{d_c d_t} - \\
& - [A_{d_t o} \boldsymbol{\omega}_o^{oI} \times][A_{d_t o} \boldsymbol{\omega}_o^{oI} \times] \underline{(\boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t})} - \\
& - 2[A_{d_t o} \boldsymbol{\omega}_o^{oI} \times] \dot{\boldsymbol{s}}_{d_t}^{d_c d_t} - 2[A_{d_t o} \boldsymbol{\omega}_o^{oI} \times][\boldsymbol{\omega}_{d_t}^{d_t o} \times] \underline{(\boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t})} + \\
& + 2[A_{d_t o} \boldsymbol{\omega}_o^{oI} + \boldsymbol{\omega}_{d_t}^{d_t o} \times][A_{d_c d_t}^T \boldsymbol{\omega}_{d_c}^{d_c d_t} \times] A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \\
& + [A_{d_c d_t}^T \dot{\boldsymbol{\omega}}_{d_c}^{d_c d_t} \times] A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + 2[A_{d_c d_t}^T \dot{\boldsymbol{\omega}}_{d_c}^{d_c d_t} \times][A_{d_c d_t}^T \dot{\boldsymbol{\omega}}_{d_c}^{d_c d_t} \times] A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \\
& + \mu_E \frac{A_{d_t o} \boldsymbol{r}_o^{b_t}}{||r_o^t||^3} - \mu_E \frac{A_{d_t o} \boldsymbol{r}_o^{b_t} + \boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t}}{||A_{d_t o} \boldsymbol{r}_o^{b_t} + \boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c} + \boldsymbol{r}_{d_t}^{d_t}||^3} + A_{d_t o} A_{oI} \frac{\boldsymbol{F}_I}{m_c}
\end{aligned} \tag{2.29}$$

where $\boldsymbol{r}_o^{b_t}$ is the target's CoM position with respect to the inertial frame but expressed in the orbital frame so that it is constant: $\boldsymbol{r}_o^{b_t} = \begin{bmatrix} 0 & 0 & -r_O \end{bmatrix}^T$, with r_O being the orbit radius.

The expression in (2.29) describes the two-body problem in a non-inertial frame, not referred to their CoMs but to their docking ports. It is possible to identify the following elements:

- the underlined terms represent the relative positions of the CoMs expressed in the docking frame of the target; the differentiation of this term in first place gives rise to a chain of other terms;

- all the terms of the form $2[\boldsymbol{\omega}\times]\dot{\boldsymbol{s}}$ represent Coriolis's accelerations;

- all the terms of the form $[\boldsymbol{\omega}\times][\boldsymbol{\omega}\times]\boldsymbol{s}$ represent centripetal accelerations;

- the last line of the equation is the inertial gravitational term present in a similar way also in the derivation of the Hill's Equations [8];

- the very last term, $A_{d_to}A_{oI}\frac{\boldsymbol{F}_I}{m_c}$, is the chaser's control force (divided by its mass) expressed in the target docking frame; however, it is better to have such force expressed in the target docking frame by means of the relative attitude involved:

$$A_{d_to}A_{oI}\frac{\boldsymbol{F}_I}{m_c} = A^T_{d_cd_t}\frac{\boldsymbol{F}_{d_c}}{m_c}; \tag{2.30}$$

- the chaser's mass, m_c, in a real mission is not constant, but will diminish due to the fuel consumption. The variable mass model, derived from Tsiolkovsky's rocket equation and used in the simulator designed in this thesis, is described in Appendix A;

- the P2P position dynamics just derived is parametrized such that only the target's absolute attitude dynamics in the docking frame (A_{d_to}) and the P2P relative attitude dynamics ($A_{d_cd_t}$) appear.

The kinematics involved in the P2P position is trivial

$$\frac{d}{dt}\boldsymbol{s}^{d_cd_t}_{d_t} = \dot{\boldsymbol{s}}^{d_cd_t}_{d_t} \tag{2.31}$$

The dynamics expressed by (2.29) can be written in a nonlinear matricial form as:

$$\dot{\boldsymbol{x}}_{pos} = \begin{bmatrix} \dot{\boldsymbol{s}}^{d_cd_t}_{d_t} \\ \ddot{\boldsymbol{s}}^{d_cd_t}_{d_t} \end{bmatrix} = \boldsymbol{f}_{pos}\Big(\boldsymbol{x}_{pos}, \boldsymbol{x}_{att}, \boldsymbol{u}_{pos}\Big), \quad \boldsymbol{x}_{pos} = \begin{bmatrix} \boldsymbol{s}^{d_cd_t}_{d_t} \\ \dot{\boldsymbol{s}}^{d_cd_t}_{d_t} \end{bmatrix}, \quad \boldsymbol{u}_{pos} = \boldsymbol{F}_{d_c} \tag{2.32}$$

The nonlinear models from (2.26) and (2.32) can now be put together to obtain the complete P2P dynamics of the system

$$\begin{aligned} \dot{\boldsymbol{x}}_{full} &= \begin{bmatrix} \dot{\boldsymbol{\alpha}}^{d_c d_t} \\ \dot{\boldsymbol{\omega}}^{d_c d_t}_{d_c} \\ \dot{\boldsymbol{\alpha}}^{d_t o} \\ \dot{\boldsymbol{\omega}}^{d_t o}_{d_t} \\ \dot{\boldsymbol{s}}^{d_c d_t}_{d_t} \\ \ddot{\boldsymbol{s}}^{d_c d_t}_{d_t} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_{att}\Big(\boldsymbol{x}_{att}, \boldsymbol{u}_{att}\Big) \\ \boldsymbol{f}_{pos}\Big(\boldsymbol{x}_{pos}, \boldsymbol{x}_{att}, \boldsymbol{u}_{pos}\Big) \end{bmatrix} = \boldsymbol{f}_{p2p}\Big(\boldsymbol{x}_{full}, \boldsymbol{u}_{full}\Big) \\ \boldsymbol{x}_{full} &= \begin{bmatrix} \boldsymbol{x}_{att} \\ \boldsymbol{x}_{pos} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}^{d_c d_t} \\ \boldsymbol{\omega}^{d_c d_t}_{d_c} \\ \boldsymbol{\alpha}^{d_t o} \\ \boldsymbol{\omega}^{d_t o}_{d_t} \\ \boldsymbol{s}^{d_c d_t}_{d_t} \\ \dot{\boldsymbol{s}}^{d_c d_t}_{d_t} \end{bmatrix}, \quad \boldsymbol{u}_{full} = \begin{bmatrix} \boldsymbol{u}_{att} \\ \boldsymbol{u}_{pos} \end{bmatrix} = \begin{bmatrix} \boldsymbol{T}_{d_c} \\ \boldsymbol{T}_{d_t} \\ \boldsymbol{F}_{d_c} \end{bmatrix} \end{aligned} \tag{2.33}$$

To obtain the P2P state space model, as mentioned at the end of Section 2.4.1, the nonlinear functions in (2.33) must be linearized around the operational point that best describes the situation desired throughout the final approach phase. Such condition is described as follows

- target nominally aligned to its local orbital frame and not rotation with respect to it: $\bar{\boldsymbol{\alpha}}^{d_t o} = \bar{\boldsymbol{\omega}}^{d_t o}_{d_t} = \boldsymbol{0}$;

- chaser docking frame nominally aligned to the target docking frame and not rotating with respect to it: $\bar{\boldsymbol{\alpha}}^{d_c d_t} = \bar{\boldsymbol{\omega}}^{d_c d_t}_{d_c} = \boldsymbol{0}$;

- relative position of the two satellites reduced to zero, with no relative velocity (this linearization will be more and more accurate as the chaser get closer to the target and slows down): $\boldsymbol{s}^{d_c d_t}_{d_t} = \dot{\boldsymbol{s}}^{d_c d_t}_{d_t} = \boldsymbol{0}$;

- control torques and forces are all zero: $\boldsymbol{T}_{d_c} = \boldsymbol{T}_{d_t} = \boldsymbol{F}_{d_c} = \boldsymbol{0}$.

Such linearization has been obtained by means of MATLAB®'s Symbolic Math Toolbox, and the obtained state space model that is of the form:

$$\begin{aligned} \dot{\boldsymbol{x}}_{full} &= A_{full}\boldsymbol{x}_{full} + B_{full}\boldsymbol{u}_{full} \\ \boldsymbol{y}_{full} &= C_{full}\boldsymbol{x}_{full} + D_{full}\boldsymbol{u}_{full} \end{aligned} \tag{2.34}$$

where $A_{full} \in \mathbb{R}^{18\times 18}$ and $B_{full} \in \mathbb{R}^{18\times 9}$ are matrices of the form

$$A_{full} = \begin{bmatrix} 0 & A_{12} & 0 & 0 & 0 & 0 \\ A_{21} & A_{22} & A_{23} & A_{24} & 0 & 0 \\ 0 & 0 & 0 & A_{34} & 0 & 0 \\ 0 & 0 & A_{43} & A_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{56} \\ A_{61} & A_{62} & A_{63} & A_{64} & A_{65} & A_{66} \end{bmatrix} \tag{2.35}$$
$$\text{with } A_{ij} \in \mathbb{R}^{3\times 3} \quad \forall i,j = 1,\dots,6;$$

$$B_{full} = \begin{bmatrix} 0 & 0 & 0 \\ B_{21} & B_{22} & 0 \\ 0 & 0 & 0 \\ 0 & B_{42} & 0 \\ 0 & 0 & 0 \\ B_{61} & B_{62} & B_{63} \end{bmatrix} \tag{2.36}$$
$$\text{with } B_{ij} \in \mathbb{R}^{3\times 3} \quad \forall i = 1,\dots,6; \quad j = 1,\dots,3;$$

An explanation of some of the elements of A_{full} and B_{full} under particular circumstances can be found in [6]. For what regards matrices C_{full} and D_{full} relative to the measurement equation in (2.34), their forms depend on the sensor suit used for the state measurement and they will be described fully in Chapter 3.

It is worth to notice that the state space model just derived could be used for a simultaneous control of the absolute attitude of the target and the relative attitude and position. T_{d_t} appears explicitly, thus allowing, in general, the derivation of a control law which determines the control input for both satellites at the same time. However, such solution would rely on a communication system such as an Inter Satellite Link (ISL). For operational reasons [6] it is desirable to maintain target's and chaser's GNC systems independent during the FA phase of the mission, so to avoid the technological challenge associated to the requirement of a constant communication between the two CubeSats.

As a consequence, the model in (2.34) must be reduced removing the target's absolute kinematics and dynamics. This will intrinsically imply a loss of information regarding the coupled motion of the two docking ports. Fortunately, not all the information is lost and some of it remains in the coupling between the relative acceleration and the relative attitude and angular velocities, described by elements A_{61} and A_{62} of matrix A_{full}. The reduced model, which is the one that has been used in the controller and estimator used for this research, will be given by

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} \\ \boldsymbol{y} &= C\boldsymbol{x} + D\boldsymbol{u} \end{aligned} \tag{2.37}$$

where

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\alpha}^{d_c d_t} \\ \boldsymbol{\omega}_{d_c}^{d_c d_t} \\ \boldsymbol{s}_{d_t}^{d_c d_t} \\ \dot{\boldsymbol{s}}_{d_t}^{d_c d_t} \end{bmatrix} \in \mathbb{R}^{12} \tag{2.38a}$$

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{T}_{d_c} \\ \boldsymbol{F}_{d_c} \end{bmatrix} \in \mathbb{R}^{6} \tag{2.38b}$$

$$A = \begin{bmatrix} 0 & A_{12} & 0 & 0 \\ A_{21} & A_{22} & 0 & 0 \\ 0 & 0 & 0 & A_{56} \\ A_{61} & A_{62} & A_{65} & A_{66} \end{bmatrix} \in \mathbb{R}^{12\times 12} \tag{2.38c}$$

$$B = \begin{bmatrix} 0 & 0 \\ B_{21} & 0 \\ 0 & 0 \\ B_{61} & B_{63} \end{bmatrix} \in \mathbb{R}^{12\times 6} \tag{2.38d}$$

2.5 Models of Actuators

In this thesis, the attitude and position actuation systems are modeled as independent and uncoupled. This is clearly an over-optimistic assumption, but a realistic model of the actuation system was not the goal of this research.

2.5.1 Reaction Control System

To approach and dock to the target, the chaser is equipped with a propulsion system. This type of systems, which propel spacecraft by ejecting fuel so to take advantage of the conservation of the linear momentum, are usually called Reaction Control Systems (RCS). Cold gas RCSs for nanosatellites are now mature technology and are soon to be used on ESA's RACE mission, in which an autonomous docking between two 6U CubeSats will be attempted [24]. In the ESA's mission the chaser will be equipped with two *NanoProp 6DOF* cold gas RCSs from GOMspace [24].

It is assumed each RCS provides 6 thrusters. The total 12 thrusters are arranged in a four-per-axis fashion. The thrusters offsets with respect to the Centre of Mass of the CubeSat will induce unwanted torques. Alternatively, this effect could be used to control also the three rotational degrees of freedom. In this research this effect is neglected, assuming that the thrusters are aligned with the respective axes, allowing only propulsion and producing no disturbance torque. Thanks to this simplifying assumption, the RCS modelled pruces a force which can be directly expressed as $\boldsymbol{F}_{b_c}$.

Recall that the P2P dynamics just defined requires the input forces to be expressed with respect to the docking frame. However, two assumptions have been made:

1. body frame $\mathcal{F}_{b_c}$ and docking frame $\mathcal{F}_{d_c}$ are aligned (2.4);
2. there is no misalignment between the thrust force produced and the body axes.

As a consequence, assuming the CubeSat structure can be modeled as a rigid body, the following holds:

$$\boldsymbol{F}_{d_c} = \boldsymbol{F}_{b_c} \tag{2.39}$$

The RCS is assumed to be controlled by means of a Pulse-Width-Pulse-Frequency (PWPF) modulator: the continuous command control signal is translated to an on-off signal [25]. The modulator features a Schmidt trigger, a lag network filter and feedback loop, as depicted in Figure 2.9.

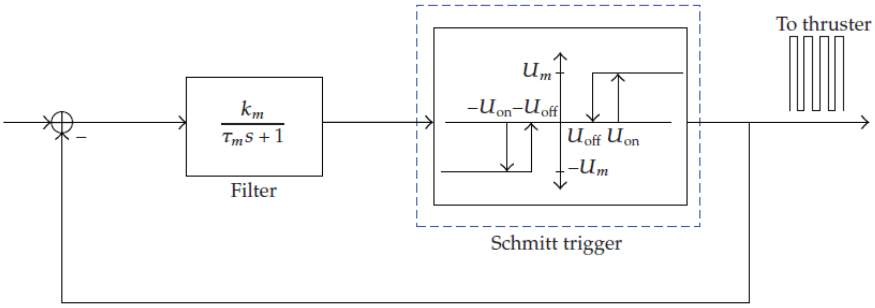


Figure 2.9: Pulse-Width-Pulse-Frequency modulator scheme [25]

2.5.2 Reaction Wheels

The chaser is also equipped with an actuation system that will control the attitude of the spacecraft. GOMspace provides many options already tested for nanosatellites, ranging from magnetorquer-based solutions [26] to RW-based ones [27].

In this research, a simplified solution is considered, assuming a Clyde Space RW210 reaction wheel per body axis can be installed in the chaser [28]. It is also assumed that the three wheels are perfectly aligned to the respective axis, without generating any disturbance torque.

Recall that the P2P dynamics has been developed in a general way, not including specific effects associated to the actuators chosen for the problem. The dynamic

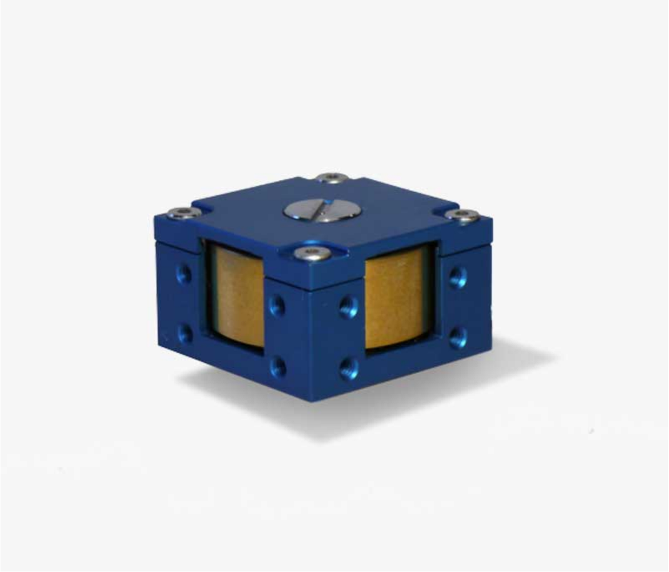


Figure 2.10: Clyde Space RW210 reaction wheel [28]

model obtained, for instance, can be used in association to any attitude actuation system. Chosen an RW-based solution, the gyroscopic coupling between reaction wheels and satellite angular velocity neglected in Section 2.4.1 must be accounted for in the actuator's model. Such coupling is described by the term $\boldsymbol{\omega}_{b_c}^{b_cI} \times \boldsymbol{H}_{b_c}^{RW}$. The actual torque actuating the chaser will thus be

$$\boldsymbol{T}_{b_c} = \boldsymbol{T}_{b_c}^{RW} - \boldsymbol{\omega}_{b_c}^{b_cI} \times \boldsymbol{H}_{b_c}^{RW} \tag{2.40}$$

where $\boldsymbol{T}_{b_c}^{RW}$ is the control torque requested by the LQR to the RW system. Recalling (2.11), the chaser's angular velocity with respect to the ECI frame, $\boldsymbol{\omega}_{b_c}^{b_cI}$, can be expressed as

$$\boldsymbol{\omega}_{b_c}^{b_cI} = \boldsymbol{\omega}_{b_c}^{b_co} + A_{b_co}\boldsymbol{\omega}_{o}^{oI} \tag{2.41}$$

where $\boldsymbol{\omega}_{o}^{oI}$ has been defined in (2.12). The chaser is modelled as a perfectly rigid body, so its angular velocity $\boldsymbol{\omega}_{b_c}^{b_co}$ is a property of every point of the structure and can be replaced with $\boldsymbol{\omega}_{d_c}^{d_co}$ (docking and body frame have the same orientation). Thanks to an angular velocities composition it is true that

$$\boldsymbol{\omega}_{d_c}^{d_co} = \boldsymbol{\omega}_{d_c}^{d_cd_t} + \boldsymbol{\omega}_{d_c}^{d_to} \tag{2.42}$$

where

- $\boldsymbol{\omega}_{d_c}^{d_cd_t}$ is the angular velocity part of the P2P dynamics state vector $\boldsymbol{x}$ defined in (2.38a);

- $\boldsymbol{\omega}_{d_c}^{d_t o}$ is the target's docking system angular velocity with respect to the orbital frame. It is zero thanks to the assumption of a target non rotating and perfectly aligned to its orbital frame.

So it is true that

$$\boldsymbol{\omega}_{d_c}^{d_c o} = \boldsymbol{\omega}_{d_c}^{d_c d_t} \tag{2.43}$$

The attitude matrix $A_{b_c o}$ appearing in (2.41) can be rewritten by means of a composition of DCMs

$$A_{b_c o} = A_{b_c d_c} A_{d_c o} = A_{b_c d_c} A_{d_c d_t} A_{d_t o} \tag{2.44}$$

where

- $A_{b_c d_c}$ is the identity matrix (again, docking and body frames have the same orientation);
- $A_{d_c d_t}$ is the relative attitude matrix of the two spacecraft, which can be obtained from the attitude angles contained in $\boldsymbol{x}$ used in (3.15);
- $A_{d_t o}$ is the identity matrix because of the assumption of a target perfectly aligned to its orbital frame.

So it is true that

$$A_{b_c o} = A_{d_c d_t} \tag{2.45}$$

In conclusion,(2.41) can be rewritten as

$$\boldsymbol{\omega}_{b_c}^{b_c I} = \boldsymbol{\omega}_{d_c}^{d_c d_t} + A_{d_c d_t} \boldsymbol{\omega}_{o}^{oI} \tag{2.46}$$

and substituted in (2.40)

$$\boldsymbol{T}_{b_c} = \boldsymbol{T}_{b_c}^{RW} - \left(\boldsymbol{\omega}_{d_c}^{d_c d_t} + A_{d_c d_t} \boldsymbol{\omega}_{o}^{oI} \right) \times \boldsymbol{H}_{b_c}^{RW} \tag{2.47}$$

The latter can be easily implemented in Simulink observing that the reaction wheels momenta $\boldsymbol{H}_{b_c}^{RW}$ can be computed by means of an integrator applied to the reaction wheels torque $\boldsymbol{T}_{b_c}^{RW}$, namely implementing the following relation

$$\boldsymbol{T}_{b_c}^{RW} = \dot{\boldsymbol{H}}_{b_c}^{RW} \tag{2.48}$$

Recall that the dynamic model requires torque inputs expressed in the docking frame, but a similar consideration as for the RCS can be done and $\boldsymbol{T}_{b_c}$ can be safely fed to the model.

No RW unloading system is modeled in the simulator designed in this research. Such system is of paramount importance to avoid saturation of the reaction wheels rotation speed. However, for the sake of the thesis, it can be neglected.

Notice that, to model the limited bandwidth of the control loop that will act on the RW system, the control torque generated by the LQR is first fed to a low pass filter and then to the model defined in (2.47).

2.6 Disturbances

The disturbances that have been taken into account are those with the most significant effects in LEO rendezvous missions [8]. In this specific research, the focus is on the Final Aapproach phase of the mission. Such phase lasts for a time-span of the order of tens of minutes, while the complete manoeuvre (Phasing, Homing, Closing, Final Approach) could last for a timespan of the order of days. Some of these fundamental disturbances have their main effect over long absolute trajectories. However, even though during Final Approach the biggest disturbance source is the aerodynamic pressure due to the residual atmosphere present in LEO, for sake of completeness also Earth's residual magnetic dipole and gravitational field are taken into account.

The simulator designed in this research, as already mentioned, considers only the actuation and control of the chaser. The target is assumed perfectly aligned to its local orbital frame and not moving with respect to it (2.19). Thus, the disturbances will only be considered for the chaser.

2.6.1 Aerodynamic Drag

The density of the atmosphere surrounding Earth decreases with a logarithmic scale as the altitude increases (Figure 2.11). In first approximation, at LEO altitudes, such density could be neglected. However, the impact it has on a light satellite such as a CubeSat, together with the high precision required in the considered mission, implies the necessity to model such disturbance.

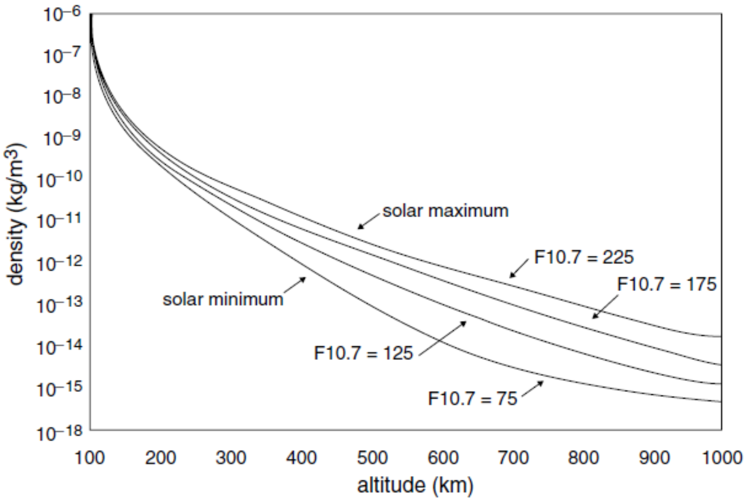


Figure 2.11: Atmosphere density logarithmically decreasing with the altitude [29]

The general aerodynamic force due to a residual atmosphere, expressed in the local orbital frame, is given by

$$\boldsymbol{F}_o^D = \frac{1}{2}\rho \boldsymbol{V}_o^{rel^2} S C_D \hat{\boldsymbol{V}}_o^{rel} \tag{2.49}$$

where:

- ρ is the air density, depending on the altitude;
- $\boldsymbol{V}_o^{rel}$ is the vector of relative velocity of the atmosphere with respect to the spacecraft, expressed in the orbital frame;
- S is the surface area of the spacecraft perpendicular to $\boldsymbol{V}_o^{rel}$, and for this research it will be the 1U docking side of the CubeSat;
- C_D is the drag coefficient associated to the surface S; for CubeSats, such surface can be safely approximated with a flat two-dimensional plate, implying $C_D = 2$ [6].

Assuming an ideal LEO, both chaser and target are orbiting in a uniform circular motion around Earth. The velocity is given by the product between the orbit radius r_O and the orbital mean motion ω_O. Assuming the atmosphere as static while the spacecraft traverse it, the squared relative velocity and the corresponding versor can be written as

$$\boldsymbol{V}_o^{rel^2} = \left(-r_O\omega_O\right)^2, \quad \hat{\boldsymbol{V}}_o^{rel} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{2.50}$$

The control forces and torques acting on the chaser are expressed in the docking frame of the satellite (2.37), so it is convenient to express also (2.49) in such frame:

$$\boldsymbol{F}_{d_c}^D = A_{d_c d_t} A_{d_t o} \boldsymbol{F}_o^D \tag{2.51}$$

where

- $A_{d_t o}$ is the absolute attitude of the target expressed in the docking frame (2.21), which in this research is assumed to be the identity matrix (see (2.19) and relative comments);
- $A_{d_c d_t}$ is the relative attitude between the two spacecraft, available from the state space model (2.37).

If the CoM of the spacecraft is not aligned with the centre of pressure of the drag force $\boldsymbol{F}_o^D$, the disturbance also generates a torque, which in the body frame can be expressed as:

$$\boldsymbol{T}_b^D = \boldsymbol{r}_b^{cp} \times \boldsymbol{F}_b^D \tag{2.52}$$

where the superscript *cp* stands for "centre of pressure".

Assuming the docking mechanism centred in the docking face of the chaser, the position of the centre of pressure in the body frame coincides with the position of the docking frame defined in (2.5a)

$$\boldsymbol{r}_b^{cp} = \boldsymbol{r}_{b_c}^{d_c} = \begin{bmatrix} 0.15 \\ 0 \\ 0 \end{bmatrix} \text{m} \tag{2.53}$$

To compute (2.52), it is required to modify the aerodynamic force in (2.51) from the docking frame to the body frame

$$\boldsymbol{F}_{b_c}^D = A_{b_c d_c} \boldsymbol{F}_{d_c}^D \tag{2.54}$$

However, from (2.4), $A_{b_c d_c}$ corresponds to the identity matrix, thus the torque can be directly computed. Furthermore, for the same reason, $\boldsymbol{T}_b^D$ doesn't change when expressed in the docking frame.

2.6.2 Magnetic Torque

The various pieces of electronic equipment inside a spacecraft orbiting around Earth generate a residual magnetic dipole $\boldsymbol{m}$ which interacts with the Earth's magnetic field $\boldsymbol{B}$, not negligible at LEO altitudes (Figure 2.12). This generates a torque which in general can be expressed as

$$\boldsymbol{T}^m = \boldsymbol{m} \times \boldsymbol{B} \tag{2.55}$$

The most accurate model for the computation of $\boldsymbol{B}$ are extremely complex. However, an approximated model can be found in [31], where the Earth's magnetic field in the local orbital frame is modeled as

$$\boldsymbol{B}_o = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix} = \frac{\mu}{r_O^3} \begin{bmatrix} \cos(\omega_O t) \sin(i_m) \\ -\cos(i_m) \\ 2\sin(\omega_O t)\sin(i_m) \end{bmatrix} \tag{2.56}$$

where

- $\mu = 7.9 \cdot 10^{15}$ Wb m is the magnetic field's dipole strength;
- i_m is the orbit inclination with respect to the magnetic equator;

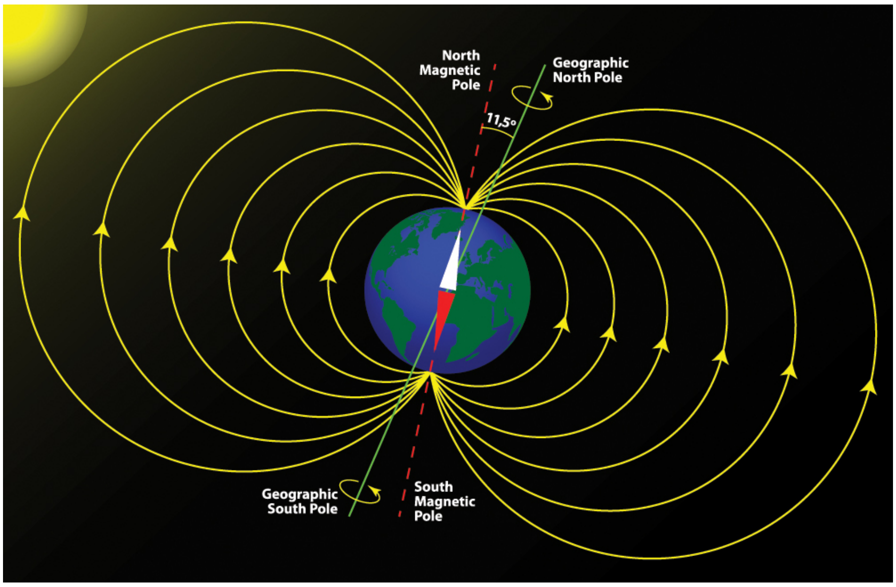


Figure 2.12: A schematic representation of the Earth's magnetic field [30]

- t is the time passed from the ascending-node crossing of the magnetic equator.

To compute (2.55) in the local orbital frame, an estimation of the spacecraft's residual dipole $\boldsymbol{m}_o$ is still required. An a-priori value for such parameter doesn't exist. The CubeSat should be constructed, then $\boldsymbol{m}_b$ should be measured and a correction by means of the absolute attitude of the spacecraft should be applied. However, Farissi et al. [32] estimated a worst-case scenario based on other studies, so to maintain a conservative approach, setting a value of $m_0 = 0.1$ Am^2 for each axis of the body frame.

To compute the disturbance torque in the body frame, (2.56) can be modified in a similar way as in (2.51), thus leading to

$$\boldsymbol{T}^m_{b_c} = \boldsymbol{m}_{b_c} \times \boldsymbol{B}_{b_c} = \begin{bmatrix} m_0 \\ m_0 \\ m_0 \end{bmatrix} \times A_{b_c d_c} A_{d_c d_t} A_{d_t o} \boldsymbol{B}_o \tag{2.57}$$

As in (2.54), $\boldsymbol{T}^m_{b_c}$ corresponds to the torque in the docking frame, being $A_{b_c d_c} = \mathbb{1}_3$.

2.6.3 Gravity Gradient

Earth is not a perfect sphere, but a geoid, implying a non-uniform distribution of mass and thus a complex gravity field potential. However, in first approximation, Earth can be considered a sphere with an homogeneous distribution of mass. The associated gravitational force decreases with the square of the distance, so whenever a spacecraft's principal axis of inertia is not aligned to the $\bar{V}$ axis of its local orbital frame, a disturbance torque is generated (Figure 2.13).

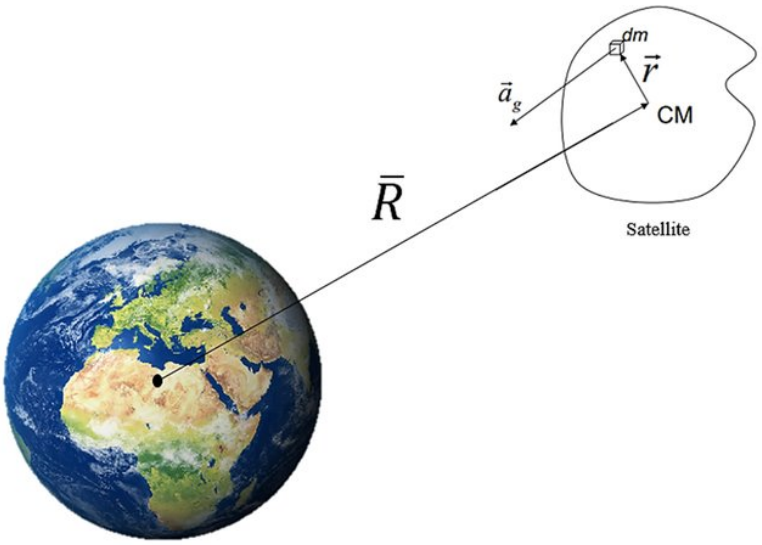


Figure 2.13: Intuitive explanation of the torque associated to the Earth's gravity gradient [33]

Such torque can be expressed as [34]

$$\boldsymbol{T}_o^g = 3\omega_O \hat{\boldsymbol{r}}_o^{b_t} \times I_b \hat{\boldsymbol{r}}_o^{b_t} \tag{2.58}$$

where

- ω_O is the orbital mean motion;
- $\hat{\boldsymbol{r}}_o^b$ is the versor corresponding to the spacecraft's CoM position with respect to the centre of Earth, expressed in the orbital frame ($\hat{\boldsymbol{r}}_o^{b_t} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$);
- I_b is the inertia tensor of the satellite.

Similarly to what has been done for the previous disturbances discussed, the torque in (2.58) must be computed in the docking frame of the chaser so that it can be correctly added to the model.

Chapter 3

Vision-Based Navigation System

The working principle of a Vision-Based Navigation (VBN) system is the following:

1. a picture is taken,

2. the fiducial markers on the picture are detected by means of an image processing algorithm,

3. the position in terms of pixels of such fiducial markers is used to estimate the state of the system:

    - either by means of a perspective problem algorithm [13] (most common approach in vision-based robotics),
    - or by means of a direct filtering of the measurement equations [35] [36],

4. the estimated state is used in a control algorithm to generate a control action,

5. the control action will modify the system and a new picture will be required to update the state estimation, thus restarting the cycle.

In this chapter, the VBN algorithm proposed by Dr. Pirat [6] is analyzed and adapted to two different LEDs patterns. The analytical solution, available thanks to the specific approach adopted by the algorithm, is derived for one of the patterns. In conclusion, the technique used to simulate LEDs pictures in the mission simulator is motivated and explained.

3.1 Measurement Model

3.1.1 Pinhole Camera Model

The best way to understand how monocular cameras can be used to perform state estimation is by means of the so-called pinhole camera model (Figure 3.1).

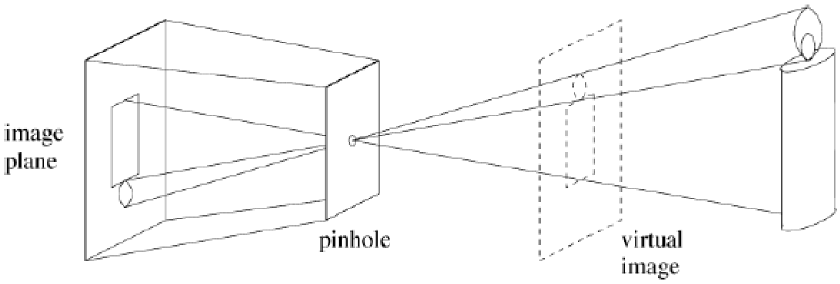


Figure 3.1: The pinhole camera model [37]

The general idea is that knowing the size D of an object in the real world and knowing the focal length f of the camera, that can be interpreted as the distance of the image plane from the pinhole, by measuring the size d on the image plane it is possible to compute the distance from the object observed (Figure 3.2)

$$R = f\frac{D}{d} \tag{3.1}$$

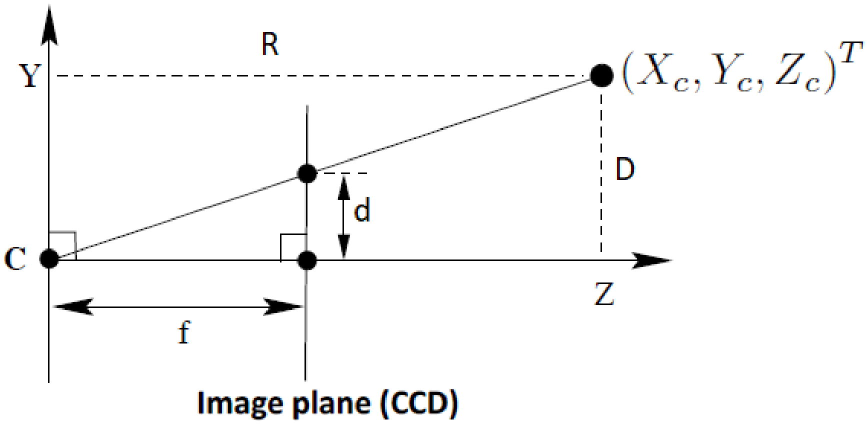


Figure 3.2: Range computation by means of the pinhole camera model [37]

In practice, when a more general and realistic situation is taken into account, considering for example perspective issues and line of sight angles, the pinhole model becomes more complex and requires the definition of additional parameters to be exhaustive (Figure 3.3).

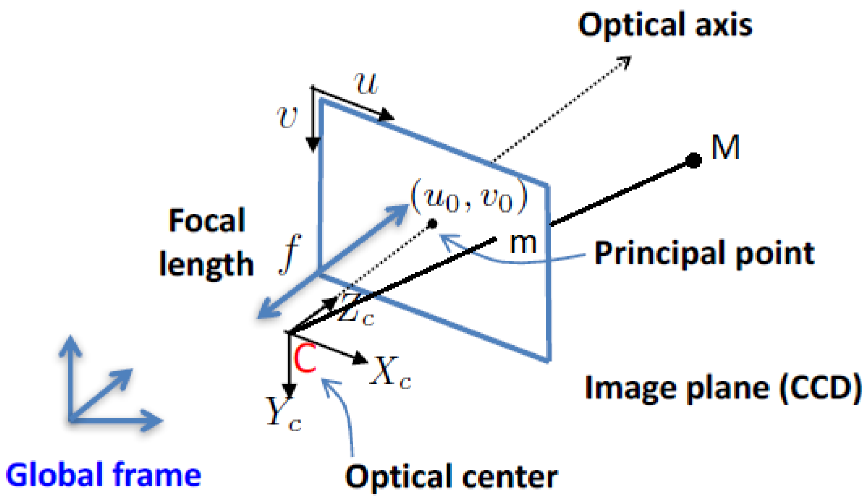


Figure 3.3: Parameters required in the most general case of the pinhole camera model [37]

- First of all, the image plane is represented in front of the camera centre C (pinhole), so to maintain the same orientation as the objects depicted.
- Three reference frames must be defined:
  - G) the three-dimensional global frame $\mathcal{F}_g$, that can be placed anywhere and oriented arbitrarily;
  - C) the three-dimensional camera frame $\mathcal{F}_c$, centred in the optical centre C, with the $\hat{X}_c - \hat{Y}_c$ plane parallel to the image plane, $\hat{X}_c$ pointing rightward, $\hat{Z}_c$ pointing as the optical axis and $\hat{Y}_c$ completing the triad;
  - IP) the two-dimensional image frame $\mathcal{F}_{ip}$, centred in the top left corner of the image plane, with axis $\hat{u}$ pointing rightward and axis $\hat{v}$ pointing downward, so that they are oriented as $\hat{X}_c$ and $\hat{Y}_c$ respectively. It follows that the coordinates in this frame cannot be negative and are bounded by the size in pixel of the Charged Coupled Device (CCD), namely the device which absorb light to generate the images placed inside the camera itself.
- The focal length f corresponds to the distance of the optical centre from the image plane, and it is an intrinsic parameter of the camera used.

- The principal point (u_0, v_0) corresponds to the centre of the image plane, namely the point of intersection between image plane and the optical axis of the camera; its coordinates in the camera frame $\mathcal{F}_c$ are given by $\begin{bmatrix}0 & 0 & f\end{bmatrix}^T$.
- A landmark M is observed by the camera and its coordinates in the global frame $\mathcal{F}_g$ are given by $\boldsymbol{r}_g^M = \begin{bmatrix}x_g^M & y_g^M & z_g^M\end{bmatrix}^T$.
- The projection of point M on the image plane is given by point m, whose coordinates $\begin{bmatrix}u^m & v^m\end{bmatrix}^T$ in the image frame $\mathcal{F}_{ip}$ can be measured in terms of pixels.

Let's assume that the coordinates of M in $\mathcal{F}_c$ are known: $\boldsymbol{r}_c^M = \begin{bmatrix}x_c^M & y_c^M & z_c^M\end{bmatrix}^T$. Thanks to (3.1), the coordinates of m can be obtained as:

$$\boldsymbol{r}_c^m = \begin{bmatrix}x_c^m \\ y_c^m \\ z_c^m\end{bmatrix} = \begin{bmatrix}f\frac{x_c^M}{z_c^M} \\ f\frac{y_c^M}{z_c^M} \\ f\end{bmatrix} \tag{3.2}$$

The pinhole camera model is based on a two-dimensional image plane: considering Figure 3.3 it is clear how no measurement can be actually taken in the direction of the optical axis. Let's define vector $\tilde{\boldsymbol{r}}_c^m = \begin{bmatrix}x_c^m & y_c^m & 1\end{bmatrix}^T$, which can be related to the coordinates of M, similarly as in (3.2), in a matricial form

$$z_c^M \tilde{\boldsymbol{r}}_c^m = \begin{bmatrix}f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1\end{bmatrix} \begin{bmatrix}x_c^M \\ y_c^M \\ z_c^M\end{bmatrix} \tag{3.3}$$

Our goal is now to pass to the image frame $\mathcal{F}_{ip}$, because it is the reference frame in which measurements of a picture are expressed. To do so it can be noticed that

- the first two components of $\tilde{\boldsymbol{r}}_c^m$ represent the position of m on the image plane, but with respect to its centre; furthermore, such components must be expressed in terms of pixels, which in general might not be square; the pixels dimensions in the two directions will thus be referred to as p_u and p_v;
- the principal point (u_0, v_0) acts as an offset;
- in principle, axes $\hat{u}$ and $\hat{v}$ might not be orthogonal, requiring the introduction of the skew parameter s which describes such non-orthogonality by correcting the pixel coordinate along the $\hat{u}$ direction; for digital cameras s is extremely small, and it can thus be neglected, but for sake of completeness for the moment it will be taken into account;

- as already mentioned, no measurement can be taken along the optical axis and thus z_c^M can be simply interpreted as a scaling parameter that will be called λ.

All these premises can be translated in the following equations

$$\begin{cases} u^m &= u_0 + k_u x_c^m + s\frac{y_c^m}{f} = u_0 + k_u f\frac{x_c^M}{\lambda} + s\frac{y_c^M}{\lambda}, \qquad k_u = \frac{1}{p_u} \\ v^m &= v_0 + k_v y_c^m = v_0 + k_v f\frac{y_c^M}{\lambda}, \qquad\qquad\qquad k_v = \frac{1}{p_v} \end{cases} \tag{3.4}$$

The latter represents the position of the pixel corresponding to M, thus corresponding to the measurement of the camera sensor. Similarly to what has been done in (3.3), Equation (3.4) can be rewritten in a matricial form in order to highlight the so called calibration matrix K, which contains the intrinsic parameter of the given camera:

$$\lambda \begin{bmatrix} u^m \\ v^m \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} x_c^M \\ y_c^M \\ z_c^M \end{bmatrix} = K\boldsymbol{r}_c^M, \qquad f_i = k_i f \quad i = u, v \tag{3.5}$$

The calibration matrix K, also known as the intrinsic parameters matrix, is computed offline by means of calibration grids of known shape and dimension.

Going back to the Vision-Based Navigation problem, vector $\boldsymbol{r}_c^M = \begin{bmatrix} x_c^M & y_c^M & z_c^M \end{bmatrix}^T$ is actually unknown. What instead is usually known is vector $\boldsymbol{r}_g^M$, namely the position of landmark M with respect to a global reference frame, unrelated to the camera position and orientation. The camera orientation with respect to the global frame $\mathcal{F}_g$ is described by the rotation matrix A_{cg}, which is unknown. The position of $\mathcal{F}_g$ with respect to the camera is instead described by vector $\boldsymbol{s}_c^{gc}$. The two elements just defined represent the so called extrinsic parameters of the problem, and (3.5) can be rewritten in terms of these parameters:

$$\lambda \begin{bmatrix} u^m \\ v^m \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} A_{cg} & \mid & \boldsymbol{s}_c^{gc} \end{bmatrix}}_{\text{extrinsic parameters}} \underbrace{\begin{bmatrix} x_g^M \\ y_g^M \\ z_g^M \\ 1 \end{bmatrix}}_{\tilde{\boldsymbol{r}}_g^M} = K \begin{bmatrix} A_{cg} & \mid & \boldsymbol{s}_c^{gc} \end{bmatrix} \tilde{\boldsymbol{r}}_g^M \tag{3.6}$$

Recall that, from the image plane, only u^m and v^m can be measured, so the third equation in (3.6) is actually a dummy equation, and the scaling parameter λ can just be assumed to be equal to 1.

It is easy to understand how this model could be used to solve the VBN problem:

- the global reference frame could be the target docking frame $\mathcal{F}_{d_t}$;

- the LEDs correspond to the landmarks, and their position in the target docking frame is known a priori;
- the camera frame is the chaser's navigation frame $\mathcal{F}_{n_c}$, and its position and orientation in the chaser's docking frame $\mathcal{F}_{d_c}$ are known a priori;
- the DCM appearing in the extrinsics of the problem contains the relative attitude matrix defined in Section 2.4.1:

$$A_{cg} \Rightarrow A_{n_c d_t} = A_{n_c d_c} \underbrace{A_{d_c d_t}}_{\text{P2P relative attitude}} \tag{3.7}$$

- the position of the global frame with respect to the camera frame contains the relative position of the two docking ports defined in Section 2.4.2:

$$\boldsymbol{s}_c^{gc} \Rightarrow \boldsymbol{s}_{n_c}^{d_t n_c} = \boldsymbol{s}_{n_c}^{d_c n_c} + A_{n_c d_t} \boldsymbol{s}_{d_t}^{d_t d_c} = \boldsymbol{s}_{n_c}^{d_c n_c} - A_{n_c d_c} A_{d_c d_t} \underbrace{\boldsymbol{s}_{d_t}^{d_c d_t}}_{\text{P2P relative position}} \tag{3.8}$$

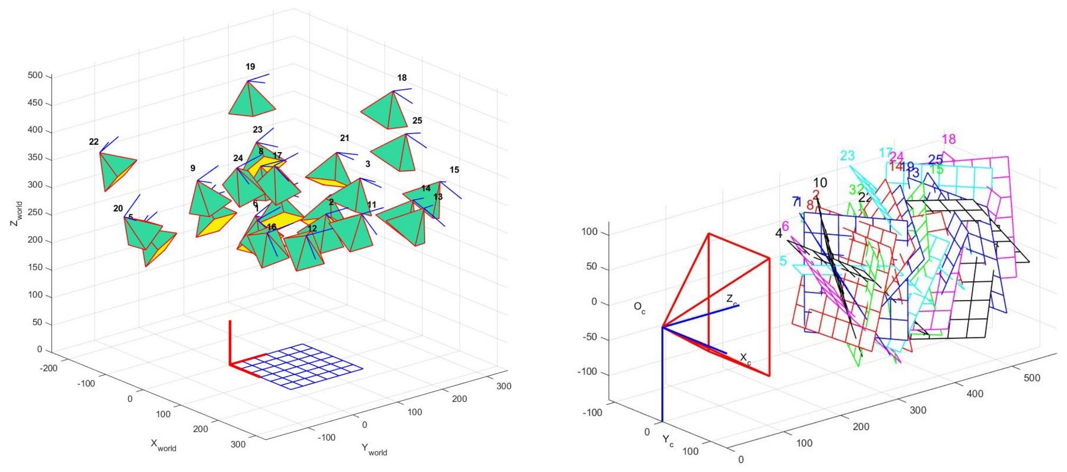
Some comments about the model just described are presented next:

- each fiducial marker to pixel correspondence can lead to two measurement equations;
- when both intrinsic and extrinsic parameters must be computed, the unknowns of the problem are eleven: six fiducial marker to pixel correspondences, leading to twelve equations, are needed to solve the problem;
- assuming the calibration matrix has been already computed offline, the problem has six unknowns: three correspondences can solve it, leading to four solutions; a fourth point is needed to disambiguate the solution;
- this type of model carries an intrinsic ambiguity in distinguishing between roto-translation of the camera with respect to fixed fiducial markers or vice versa (Figure 3.4), which is an extremely delicate issue during the final approach phase of a docking mission.

The model described in (3.6) can be solved, for example, with the perspective 3-points algorithm [13]. This type of approach involves the estimation of relative position and relative attitude in subsequent steps; such estimated variables must then be fed to a navigation filter so to account for the noise intrinsically affecting vision sensors, and only then they can be used to generate a control action.

Dr. Pirat [6] proposed a tailored solution which seeks to optimize the performance with respect to two main issues:

1. the ambiguity between rotation and translation, intuitively explained in Figure 3.4, accounted for by rewriting the model in (3.6) in an explicit way so that the Line of Sight (LoS) angles appear;



(a) Fiducial markers are assumed fixed **(b)** Monocular camera is assumed fixed

Figure 3.4: Ambiguity between a roto-translation of the camera with respect to fixed fiducial markers (a) and vice versa (b)

2. the two step process of estimating a state and then filtering the solution, accounted for by simultaneously performing state estimation and filtering inside an Extended Kalman Filter (EKF).

3.1.2 Tailored Model

In this section the tailored model developed in [6] is described and the measurement equations that will be used in the EKF are derived. Usually, when VBN algorithms are chosen for the Final Approach phase of a docking missions, multiple fiducial markers patterns with sizes increasing with the range are designed. In fact, when the two spacecraft are still distant, a larger pattern is required so that the image processing algorithm can properly detect the landmarks and allow a successful state estimation. As the spacecraft get closer, the finite Field of View (FoV) of the vision sensor constrains the maximum width and height of the pattern, so a smaller one is required to properly perform relative navigation during the final meters. In this research the focus is on this smaller and internal pattern, so to evaluate the most delicate part of the mission. The algorithm is thus described and analyzed only for the internal pattern, that will be used from a 5 m range to docking.

The model relies on the observation of a cross-shaped pattern of fiducial markers, with one out-of-plane marker to enhance the relative angles observation capability. The so called Line of Sight (LoS) angles describe the lateral (Azimuth, Az, angle) and vertical displacement (Elevation, El, angle) of the LEDs pattern's centre with respect to the camera (Figure 3.5).

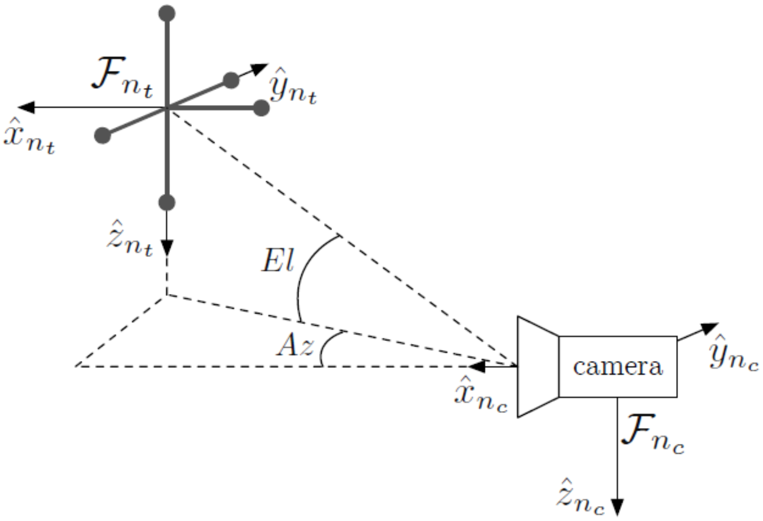


Figure 3.5: Cross-shaped pattern and LoS angles convention [6]

The navigation reference frames described in Section 2.2.5 can now be more precisely defined:

- the chaser's navigation frame $\mathcal{F}_{n_c}$ is centred in the principal point (u_0, v_0), defined in the previous section, and is oriented as the docking frame $\mathcal{F}_{d_c}$;
- the target's navigation frame $\mathcal{F}_{n_t}$ is centred in the centre of the pattern, so to easily define the LEDs' coordinates, and is oriented as the docking frame $\mathcal{F}_{d_t}$.

The cross-shaped pattern is made of five LEDs so to increase the robustness in case of failure of one of the sources of light. Furthermore, this allows the derivation of an analytical solution, explained in this chapter, which can be used as a watchdog of the state estimation performance of the EKF. A graphic representation of the problem scheme for the pattern designed by Dr. Pirat in [6] is depicted in Figure 3.6. This pattern will be referred to as Pattern 1, so to distinguish it from the other patterns tested in this thesis. The geometry and position of the three patterns will be thoroughly described in the following sections of this chapter.

The measurement equations will be non-linear and will relate the observation vector on the image plane of each LED to the state vector of the system. Such state vector is the one defined in (2.38a) for the P2P coupled dynamics

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\alpha}^{d_c d_t} & \boldsymbol{\omega}_{d_c}^{d_c d_t} & \boldsymbol{s}_{d_t}^{d_c d_t} & \dot{\boldsymbol{s}}_{d_t}^{d_c d_t} \end{bmatrix}^T \in \mathbb{R}^{12} \tag{3.9}$$

Notice that, because of how the navigation frames have just been defined (Figure 3.6), the camera model can directly compute $\boldsymbol{s}_{n_t}^{n_c n_t}$, not $\boldsymbol{s}_{d_t}^{d_c d_t}$. However, by means of frame transformations, the two vectors can be related.

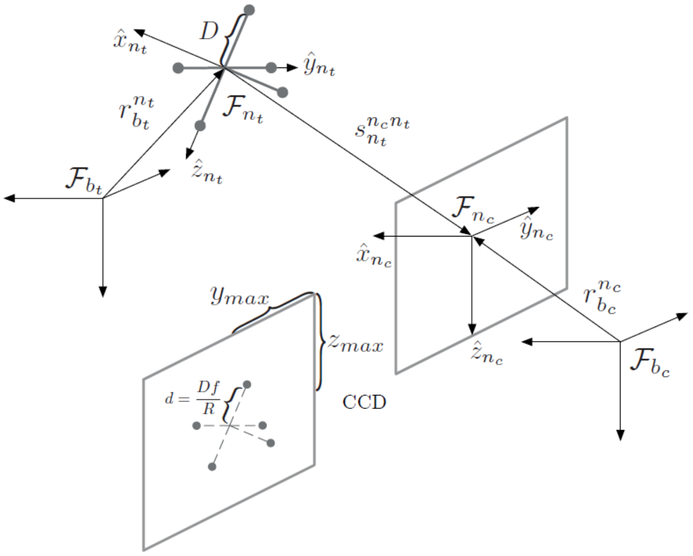


Figure 3.6: VBN scheme for Pattern 1 [6]

Let's define the known position of the LEDs in the target navigation frame

$$\boldsymbol{l}_{n_t}^i, i = 1, \ldots ,5 \tag{3.10}$$

Such vectors can be scaled by the range defined in the pinhole camera model, which can be written as $R = ||\boldsymbol{s}_{n_t}^{n_c n_t}||$, and the focal length f, which depend on the camera:

$$\boldsymbol{x}_{n_t}^i = \frac{f}{R}\boldsymbol{l}_{n_t}^i, \quad i = 1, \ldots ,5 \tag{3.11}$$

This operation somehow reflects the working principle described in (3.1): the known coordinates are scaled to the size they will have on the CCD. Notice that (3.11) implies two important assumptions: the pixels are orthogonal ($s = 0$) and square ($p_u = p_v$). Obviously vectors $\boldsymbol{x}_{n_t}^i$ now depend on the relative position, but will lead to a convenient set of measurement equations.

Let's define also the LEDs position vectors observed in the image plane. These can be defined as three parameters vectors, even though the first parameter, so the observation along the optical axis $\hat{x}_{n_c}$, will be a parameter with no physical meaning

$$\boldsymbol{x}_{n_c}^i = \begin{bmatrix} k_i & p_y^i & p_z^i \end{bmatrix}^T, \quad i = 1, \ldots ,5 \tag{3.12}$$

The second and third components of the vector, p_y and p_z, correspond to the LED position in unit length (not in pixels) in the image plane, with respect to the principal point.

Finally, let's define the position of the pattern's centre on the image plane as

$$\boldsymbol{x}^c_{n_c} = \begin{bmatrix} k_c & p^c_y & p^c_z \end{bmatrix}^T \tag{3.13}$$

Recalling that the target's navigation frame is centred in the pattern centre, a first form of the measurement equation can now be written as

$$\boldsymbol{x}^i_{n_c} = A_{n_c n_t}\boldsymbol{x}^i_{n_t} + \boldsymbol{x}^c_{n_c}, \quad i = 1, \ldots, 5 \tag{3.14}$$

where the relative attitude matrix between the two navigation frames, $A_{n_c n_t}$, appears to map the scaled position of the LEDs in the target frame to the chaser frame. Recall that $A_{n_c n_t}$ is equal to the P2P relative attitude matrix $A_{d_c d_t}$: docking and navigation frames have the same orientation in both satellites. A graphic explanation of the measurement equations described in (3.14), relative to Pattern 1, is given in Figure 3.7. It must kept in mind that the first component, for each i, of the measurement equation, has no physical meaning.

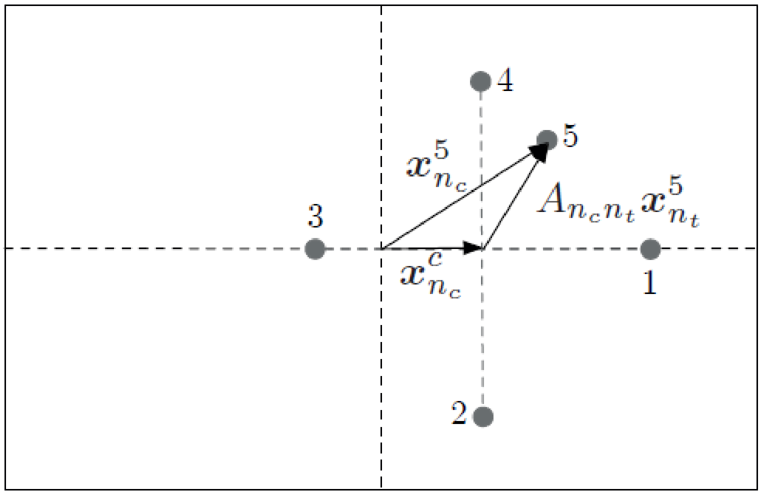


Figure 3.7: Graphical explanation of the measurement equations described in 3.14 [6]

It is important to understand that an ambiguity between pitch/yaw rotations and LoS angles is intrinsically involved in the P2P coupled dynamics. The trick to account for this ambiguity relies on the choice, made in Section 2.4.1, to use the 1-2-3 Euler sequence to define the relative attitude matrix. The rotation matrix (2.15) associated to such sequence, when applied in equation (3.14), intrinsically implies that the roll angle correction is applied first and it is thus kept decoupled from the other two angles. In fact, because of the form of the x-axis fundamental rotation matrix (2.2a), the multiplication $R_1(\phi)\boldsymbol{x}^i_{n_t}$ only corrects the second and third parameters of vector $\boldsymbol{x}^i_{n_t}$, which belong to the only observable rows of (3.14). Performing the correction in a different order would imply a loss of information associated to the roll angle, which would pointlessly correct the first parameter of $\boldsymbol{x}^i_{n_t}$. As a consequence the navigation frames relative attitude matrix is defined as

$$A_{n_c n_t} = R_{123}(\phi, \theta, \psi) = R_3(\psi) R_2(\theta) R_1(\phi) \tag{3.15}$$

The aforementioned ambiguity can now be explicitly accounted for in the measurement equation (3.14), which can be written as

$$\boldsymbol{x}^i_{n_c} = A_{n_c n_t}(\phi, \theta + El, \psi + Az)\boldsymbol{x}^i_{n_t} + \boldsymbol{x}^c_{n_c}, \quad i = 1, \ldots, 5 \tag{3.16}$$

The latter is a function of the relative attitude, the relative position of the navigation frames (used in the definition of R inside of vectors $\boldsymbol{x}^i_{n_t}$), the camera parameters and the pattern's physical parameters. The last step is to relate vector $\boldsymbol{x}^c_{n_c}$ and the LoS angles to the P2P variables of the problem. This operation can be performed thanks to the pinhole camera model.

Recall that the pattern centre corresponds to the origin of the navigation frame $\mathcal{F}_{n_t}$. Let's start by considering the top view of the camera model when the pattern centre is observed not aligned to the optical axis. It leads to a relation between the Azimuth angle and the camera parameters (Figure 3.8). The following holds

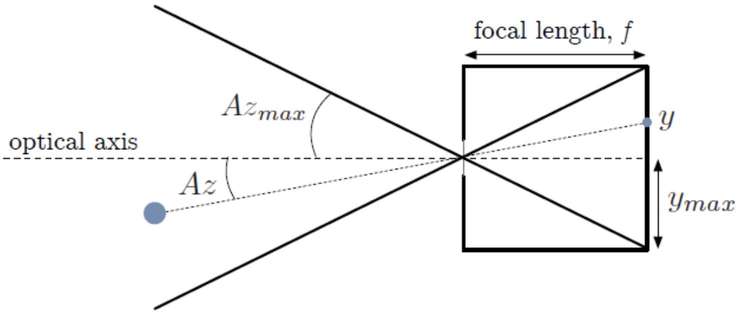


Figure 3.8: Top view of the pinhole camera model when the pattern centre (blue dot) is not aligned to the optical axis. The $\hat{y}_{n_c}$ axis points upward in the picture [6]

for the Azimuth angle

$$\tan(Az) = \frac{p^c_y}{f} \tag{3.17a}$$

$$\tan(Az_{max}) = \frac{y_{max}}{f} \Rightarrow f = \frac{y_{max}}{\tan(Az_{max})} \tag{3.17b}$$

Similarly, for the elevation angle, it holds that:

$$\tan(El) = -\frac{p_z^c}{f} \tag{3.18a}$$

$$\tan(El_{max}) = \frac{z_{max}}{f} \Rightarrow f = \frac{z_{max}}{\tan(El_{max})} \tag{3.18b}$$

Parameters y_{max} and z_{max} represent the image plane size in unit length, which depends on the number of pixels. Angles Az_{max} and El_{max} are the maximum values of the FoV along the $\hat{y}_{n_c}$ and $\hat{z}_{n_c}$ directions respectively. The negative sign for the tangent of the Elevation angle (3.18a) is given by the fact that a positive value (see the positive convention from Figure 3.5) implies the pattern centre to be seen in the negative half of the image plane with respect to $\hat{z}_{n_c}$. So for $El > 0°$ it follows that $sign(p_z^c) = -1$, so a negative sign is compliant with the chosen convention.

The pattern centre coordinates can thus be expressed as

$$p_y^c = \tan(Az)\frac{y_{max}}{\tan(Az_{max})} \tag{3.19a}$$

$$p_z^c = -\tan(El)\frac{z_{max}}{\tan(El_{max})} \tag{3.19b}$$

The LoS angles, thanks to some trigonometric considerations, can be expressed with respect to the components of vector $\boldsymbol{s}_{n_c}^{n_t n_c}$, which describes the position of the target's navigation frame with respect to the chaser's, expressed in $\mathcal{F}_{n_c}$. Such relations are given by

$$\tan(Az) = \frac{s_{n_c,y}^{n_t n_c}}{s_{n_c,x}^{n_t n_c}} \tag{3.20a}$$

$$\tan(El) = \frac{-s_{n_c,z}^{n_t n_c}}{\sqrt{s_{n_c,x}^{n_t n_c^2} + s_{n_c,y}^{n_t n_c^2}}} \tag{3.20b}$$

Vector $\boldsymbol{s}_{n_c}^{n_t n_c}$ can be related to the state variable $\boldsymbol{s}_{d_t}^{d_c d_t}$ by means of frame transformations:

1. first such vector can be inverted and rotated

$$\boldsymbol{s}_{n_c}^{n_t n_c} = -A_{n_c n_t}(\phi, \theta, \psi)\boldsymbol{s}_{n_t}^{n_c n_t} \tag{3.21}$$

2. because of the relations between docking and navigation frames (2.4) the following holds:

$$A_{n_c n_t} = \underbrace{A_{n_c d_c}}_{=\mathbb{1}_3} A_{d_c d_t} \underbrace{A_{d_t n_t}}_{=\mathbb{1}_3} \Rightarrow A_{n_c n_t} = A_{d_c d_t} \tag{3.22}$$

3. recalling the vectors describing the position of the navigation frames, defined in Section 2.2.5, vector $\boldsymbol{s}_{n_t}^{n_c n_t}$ can be written so that the state variable appears:

$$\boldsymbol{s}_{n_t}^{n_c n_t} = \underbrace{A_{d_t n_t}}_{=\mathbb{1}_3}\Bigg[\underbrace{\Big(\boldsymbol{s}_{d_t}^{d_c d_t} - A_{d_c d_t}^T \boldsymbol{r}_{d_c}^{d_c n_c}\Big)}_{\boldsymbol{s}_{d_t}^{n_c d_t}} - \boldsymbol{r}_{d_t}^{n_t d_t}\Bigg] \tag{3.23}$$

4. substituting (3.22) and (3.23) in (3.21) leads to:

$$\boldsymbol{s}_{n_c}^{n_t n_c} = \boldsymbol{r}_{d_c}^{d_c n_c} + A_{d_c d_t}\Big(\boldsymbol{r}_{d_t}^{n_t d_t} - \underbrace{\boldsymbol{s}_{d_t}^{d_c d_t}}_{\text{relative state}}\Big) \tag{3.24}$$

It can also be observed that $||\boldsymbol{s}_{n_c}^{n_t n_c}|| = ||\boldsymbol{s}_{n_t}^{n_c n_t}||$, since the two vectors describe the same distance. So the range that had been expressed as $R = ||\boldsymbol{s}_{n_t}^{n_c n_t}||$ can also be expressed as $R = ||\boldsymbol{s}_{n_c}^{n_t n_c}||$. This way, all the quantities considered so far can be expressed with respect to the P2P relative position and attitude. The non-linear tailored measurement equation 3.16 can now be written in its final form as

$$\boldsymbol{x}_{n_c}^i = A_{d_c d_t}(\phi, \theta + El, \psi + Az)\boldsymbol{x}_{n_t}^i + \begin{bmatrix} 0 \\ \tan(Az)\frac{y_{max}}{\tan(Az_{max})} \\ -\tan(El)\frac{z_{max}}{\tan(El_{max})} \end{bmatrix}, \quad i = 1, \ldots, 5 \tag{3.25}$$

The latter can be symbolically derived using MATLAB®'s Symbolic Math Toolbox. It is now possible to define the ten components measurement vector mentioned in 2.37 as

$$\boldsymbol{y} = \begin{bmatrix} p_y^1 \\ p_z^1 \\ p_y^2 \\ p_z^2 \\ \vdots \\ p_y^5 \\ p_z^5 \end{bmatrix} \in \mathbb{R}^{10} \tag{3.26}$$

such that:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\alpha}^{d_c d_t}, \boldsymbol{s}_{d_t}^{d_c d_t}) \tag{3.27}$$

A linearization of the measurement equation (3.25) around the operational point defined in Section 2.4.2 leads to the computation of matrices C and D (2.37). First of all, it can be noticed how matrix D is a zero-matrix: there is no direct effect of the control inputs $\boldsymbol{u}$ over the measurement vector $\boldsymbol{y}$. For what regards matrix $C \in R^{10\times12}$, instead, its form is not unique as for A and B, but it depends on the pattern used. Furthermore, the Extended Kalman Filter implemented for this research allows the definition of non-linear systems: the linearization of the measurement model is iteratively performed based on the current best state estimate, and a direct computation of matrix C is thus not required.

3.2 LEDs' Patterns

The patterns tested in this research are three. The first one has been designed and used, together with the algorithm just presented, by Dr. Pirat in his Ph.D thesis [6]; the peculiarity of his solution is the integration of both pattern and camera in the docking area of target and chaser respectively (Figure 3.9).

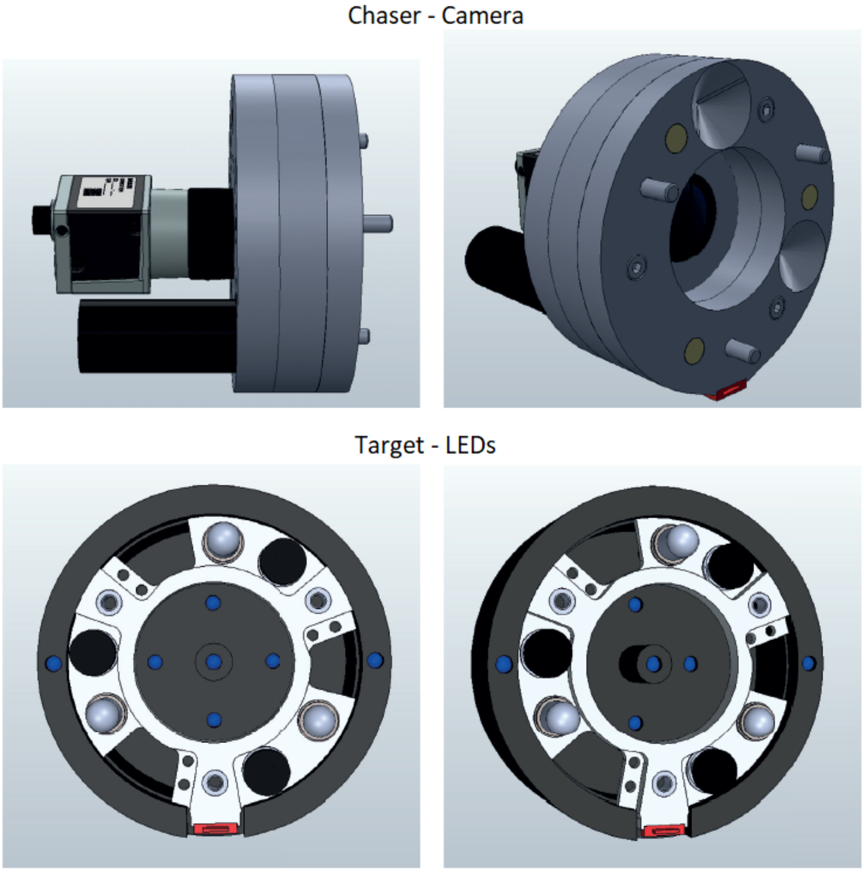


Figure 3.9: Camera and LEDs integrated in the respective docking mechanism [6]

The other two solutions, instead, are designed so that no hardware integration in the docking mechanism is required. This allows the proposed VBN solution to be taken into account for any 3U CubeSat docking mission that will be planned in the future, without requiring a specific docking system. From the hardware point of view, the docking mechanism can be totally independent from the navigation sensor suite required for the Final Approach phase: integrating five LEDs and a camera in empty areas of the docking faces of two CubeSats is definitely easier then performing such integration in a complex docking mechanism.

Satellites docking systems are usually designed so that the final centimeters of the translation are carried out by the mechanism itself by means of an electromagnetic action or a mechanical actuation (i.e. a screw and a nut). This phase is referred to as Soft Docking, and it is followed by the Hard Docking phase which starts as soon as perfect alignment is achieved and the mechanism can thus be locked mechanically. As a consequence, the VBN system is used until the range condition that allows Soft Docking is met. In this research, such range condition is set to be at $s^{d_c d_t}_{d_t,x} = -0.05$ m.

3.2.1 Pattern 1

Even though in [6] 6U CubeSats are considered, both docking mechanism and metrology system are designed so to be contained within a volume smaller than 0.5U and to fit in a 1U surface. The pattern is a symmetric 4-LEDs cross, with a fifth central out-of-plane LED (Figure 3.10b). Each fiducial marker is at the same distance $D = 2$ cm from the pattern centre, which is inserted in a 3 cm niche inside the docking face of the target; the camera as well is placed in a cavity, 4 cm deep. According to the geometric scheme of the internal and external patterns depicted in Figure 3.10a, and considering the target's docking system depicted in Figure 3.9, the solution as it is cannot fit in a 10×10 cm panel. However, considering a small reduction of the distance of the external LEDs from the centre of the pattern, it can be assumed that this solution fits in a 3U CubeSat.

The LEDs positions in the target's navigation frame, defined in (3.10), are the following

$$l^1_{n_t} = \begin{bmatrix} 0 & D & 0 \end{bmatrix}^T \tag{3.28a}$$

$$l^2_{n_t} = \begin{bmatrix} 0 & 0 & D \end{bmatrix}^T \tag{3.28b}$$

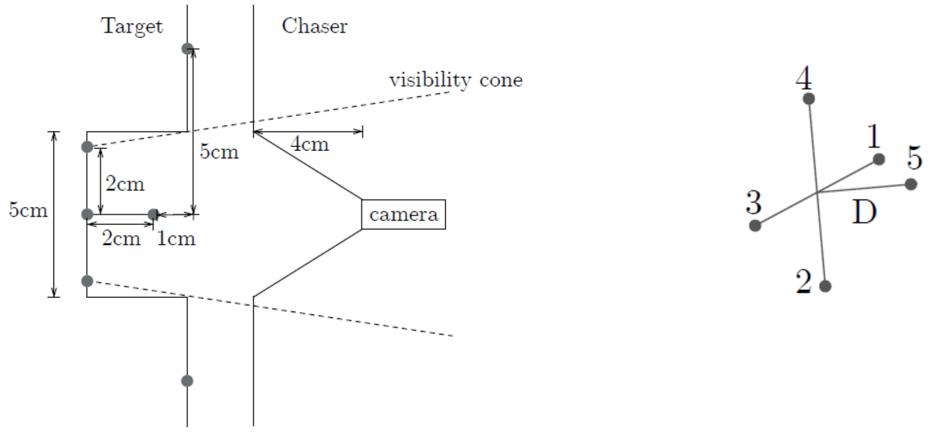
$$l^3_{n_t} = \begin{bmatrix} 0 & -D & 0 \end{bmatrix}^T \tag{3.28c}$$

$$l^4_{n_t} = \begin{bmatrix} 0 & 0 & -D \end{bmatrix}^T \tag{3.28d}$$

$$l^5_{n_t} = \begin{bmatrix} -D & 0 & 0 \end{bmatrix}^T \tag{3.28e}$$

Recall that the measurement model (3.25) uses a scaled version of these vectors: $\boldsymbol{x}^i_{n_t} = \frac{f}{R}\boldsymbol{l}^i_{n_t}, \quad i = 1, \ldots, 5.$

Being both pattern centre and camera placed in cavities inside the respective satellites, the positions of the navigation frames with respect to the docking frames



(a) Top view of the geometric disposition of the LEDs **(b)** 3D view and numbering of Pattern 1

Figure 3.10: Pattern 1 [6]

required in (3.24) are:

$$\boldsymbol{r}_{d_c}^{d_c n_c} = \begin{bmatrix} 0.04 & 0 & 0 \end{bmatrix}^T \text{ m} \tag{3.29a}$$

$$\boldsymbol{r}_{d_t}^{n_t d_t} = \begin{bmatrix} 0.03 & 0 & 0 \end{bmatrix}^T \text{ m} \tag{3.29b}$$

In this research, the monocular camera simulated is the same as in [6] (Figure 3.11): its performances, also paired with the image processing algorithm adopted, are tested and thus are reliable.

Figure 3.11: Basler ACE camera acA3800-10um [6]

It is a Basler ACE camera acA3800-10um [38] which features a sensor size of 2764 × 3856 pixels, with a pixel size is 1.67 µm. It is paired with lenses having a 4 mm focal length [39]. The combination of camera and lense yields a total FoV of ≈ 60°. The size of the sensor is 70 × 29 × 29 mm (0.03U), so the surface occupied can be modelled as a 29 × 29 mm square.

3.2.2 Pattern 2

The solutions relative to Pattern 2 and Pattern 3, as already stated, have been developed so that no integration with the docking systems is required. To achieve such goal, the docking face of the chaser must have enough room for both docking system and camera. It was decided to assume that the docking mechanism used is centred in the panel and occupies an area not larger then a circle with radius $r = 2.5$ cm. This choice fits with the 2.9×2.9 cm square that must be devoted to the monocular camera, that will be centred at a distance $C = 3.4$ cm along each axis, leaving also some empty space to allow hardware integration in the satellite 3.12.

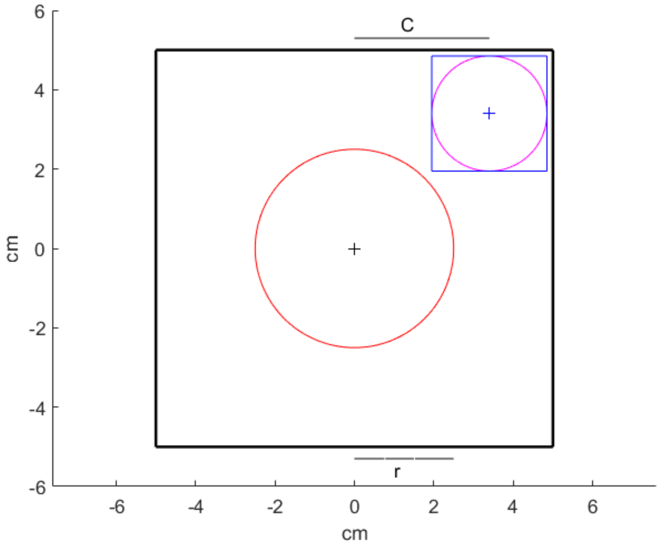
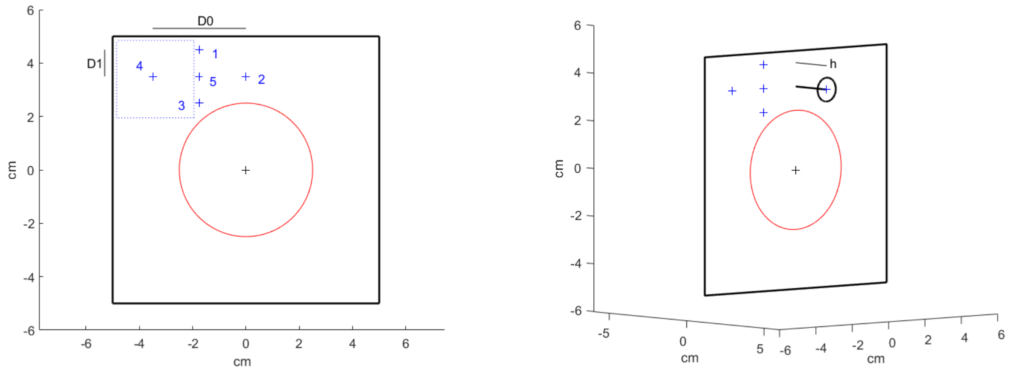


Figure 3.12: Chaser's docking face with no camera-docking system integration

In order to have the camera approximately in front of the pattern, the choice of top right corner for the camera implies that the LEDs must be placed on the top left corner of the target's docking face. Modern LEDs are very small, and can fit in a circular area with a diameter of approximately 6 mm [40]. Some space between LEDs and sides of the docking faces must thus be taken into account. In the design of Pattern 2, the goal has been the introduction of the lowest possible number of parameters (Figure 3.13):

- the centre of the pattern is placed 3.5 cm above the centre of the docking face ($D0 = 3.5$ cm);
- the centre of the pattern is placed 1.75 cm to the left of the centre of docking face ($D2 = \frac{D0}{2}$);



(a) Frontal view and numbering of Pattern 2

(b) 3D view of Pattern 2

Figure 3.13: Pattern 2

- LED number 1 is placed 1 cm above the pattern centre ($D1 = 1$ cm);
- LED number 2 protrudes by 2 cm from docking face ($h = 2$ cm).

The parameters introduced are three ($D0$, $D1$ and h, $D2$ is function of $D0$), while for Pattern 1 only one parameter is required. It can also be noticed how the origin of the target's navigation frame $\mathcal{F}_{n_t}$ coincides with the position of LED number 5. In Figure 3.13a the blue dotted line represents the projection on the target's docking face of the camera position when the state vector is zeroed, $\boldsymbol{x} = \mathbf{0}$, so to give an idea of where the LEDs will appear in the image plane in the final centimeters. Two important considerations must be done:

- because of the out-of-plane LED (number 2), this solution can only work if the final configuration after Hard Docking leaves at least 2cm between the two satellites, which depends on how the docking mechanism works;
- LED number 2 is both out-of-plane and the most to the right of the camera, outside of the projection of the camera over the docking face: perspective effects imply that such LED is the fiducial marker that has the higher risk to escape from the camera FoV in case of a large misalignment or yaw rotation in the final centimeters.

This two bullets make Pattern 2 not as reliable as Pattern 1, making this solution hardly applicable to a real mission. However, studying its performance is still useful to understand how pattern's parameters affect the docking mission.

As it has been done for Pattern 1, the position of the LEDs in the navigation frame can now be defined

$$l^1_{n_t} = \begin{bmatrix}0 & 0 & -D1\end{bmatrix}^T \tag{3.30a}$$

$$l^2_{n_t} = \begin{bmatrix}-h & \frac{D0}{2} & 0\end{bmatrix}^T \tag{3.30b}$$

$$l^3_{n_t} = \begin{bmatrix}0 & 0 & D1\end{bmatrix}^T \tag{3.30c}$$

$$l^4_{n_t} = \begin{bmatrix}0 & -\frac{D0}{2} & 0\end{bmatrix}^T \tag{3.30d}$$

$$l^5_{n_t} = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T \tag{3.30e}$$

The position of the navigation frames with respect to the docking frames are instead

$$\boldsymbol{r}^{d_c n_c}_{d_c} = \begin{bmatrix}0 & C & C\end{bmatrix}^T \tag{3.31a}$$

$$\boldsymbol{r}^{n_t d_t}_{d_t} = \begin{bmatrix}0 & -\frac{D0}{2} & -D0\end{bmatrix}^T \tag{3.31b}$$

3.2.3 Pattern 3

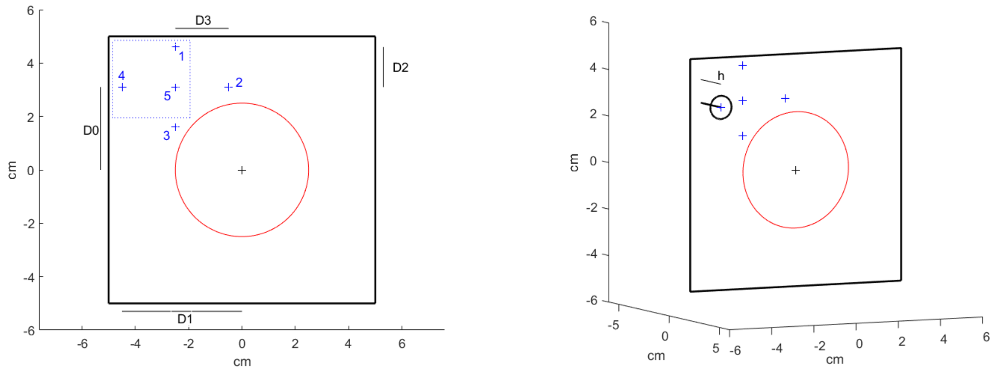
The design of Pattern 3 seeks to solve the issues associated to LED number 2, described in the Section 3.2.2, still not requiring hardware integration of pattern and camera in the docking area of the two satellites.

The camera position in the chaser's docking panel is the same as the one depicted in Figure 3.12. However, similarly as for Pattern 1, the camera is placed in a 3cm cavity inside the docking panel ($H = 3$cm): this choice strongly reduces the risk of LEDs exiting the FoV of the camera in the final centimeters and intrinsically provides protection to the camera lenses.

For what regards the pattern, it has a similar structure as for Pattern 2, but it is moved slightly to the left so to allow a bigger spacing of LEDs 1 and 3 with respect to the central one (LED number 5). Furthermore, the out-of-plane marker will be LED number 4 (Figure 3.14).

The parameters introduced for the definition of Pattern 3 are six, including the camera's cavity size H:

- the centre of the pattern is placed 3.1cm above the centre of the docking face ($D0 = 3.1$cm);
- the centre of the pattern is placed 2.5cm to the left of the centre of the docking face ($D4 = D1 - D3$);
- LED number 4 is placed 4.5cm to the left of the centre of the docking face ($D1 = 4.5$cm), so to laterally maximize the area devoted to the pattern;



(a) Frontal view and numbering of Pattern 3

(b) 3D view of Pattern 3

Figure 3.14: Pattern 3

- LED number 1 is placed 1.5cm above the pattern centre ($D2 = 1.5\text{cm}$), so to vertically maximize the area devoted to the pattern;
- LED number 2 is placed 2cm to the right of the pattern centre ($D3 = 2\text{cm}$);
- LED number 4 protrudes by 2cm from docking face ($h = 2\text{cm}$).

Also for this solution the origin of the target's navigation frame $\mathcal{F}_{n_t}$ coincides with the position of LED number 5. Furthermore, thanks to the camera's cavity and the change to LED number 4 as the out-of-plane marker, this solution can work also for systems in which the Hard Docking condition implies zero spacing between the two docking faces. This solution, as it will be shown, represents a valid alternative to Pattern 1.

The position of the LEDs in the navigation frame can now be defined

$$l^1_{nt} = \begin{bmatrix} 0 & 0 & -D2 \end{bmatrix}^T \tag{3.32a}$$

$$l^2_{nt} = \begin{bmatrix} 0 & D3 & 0 \end{bmatrix}^T \tag{3.32b}$$

$$l^3_{nt} = \begin{bmatrix} 0 & 0 & D2 \end{bmatrix}^T \tag{3.32c}$$

$$l^4_{nt} = \begin{bmatrix} -h & -D3 & 0 \end{bmatrix}^T \tag{3.32d}$$

$$l^5_{nt} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{3.32e}$$

Notice how parameters $D0$ and $D1$ don't actually appear in the LEDs positions with respect to $\mathcal{F}_{n_t}$. However, such parameters are needed to define the navigation frame position

$$\boldsymbol{r}_{d_c}^{d_c n_c} = \begin{bmatrix} H & C & C \end{bmatrix}^T \tag{3.33a}$$

$$\boldsymbol{r}_{d_t}^{n_t d_t} = \begin{bmatrix} 0 & -D1 + D3 & -D0 \end{bmatrix}^T \tag{3.33b}$$

3.3 Analytical Solution

Thanks to the choice of using a cross-shaped pattern made of 5 LEDs, an analytical solution to the measurement equation (3.25) exists [6]. Such solution provides direct access to the state of the system, but the state obtained by means of this approach is noisy, and so it must still be filtered before using it for the generation of the control inputs. Furthermore, in case of failure of one of the LEDs, the analytical solution cannot be computed, thus making this approach less reliable then the EKF one. In fact the EKF, together with receiving the pixels positions measurements, also contains the state space model of the system, and it is thus able to perform state estimation even if one or more of the LEDs fails. Of course, if such failure should happen, the estimation accuracy would worsen. However, the analytical solution can be used as a watchdog to detect the potential divergence of the navigation filter, and could be used in case of filter's failure as a contingency navigation mode.

In this research, the analytical solutions associated to Pattern 2 and Pattern 3 have been derived taking inspiration from Pirat's derivation for Pattern 1 [6]. Such solutions have been used to test the validity of the Image Generation algorithm that will be presented in the last section of this chapter. However, they are not actively used in the simulator, which instead is only based on the EKF approach. Since the derivation of the analytical solution is very similar for each of the patterns, it is here described only for Pattern 3, which, as it has been shown, is the most complex in terms of parameters involved.

3.3.1 Pattern 3

The measurement equation (3.25) can be used to derive an analytical solution. Let's consider a situation in which the coordinates of the pixels corresponding to the five LEDs have been measured. Ten measurements, two for each LED, are now available: $\boldsymbol{x}_{n_c}^i, \quad i = 1, \ldots, 5$. First of all, a vector describing the difference between the measured coordinates and the pattern centre coordinate can be defined

$$\boldsymbol{x}'^i \equiv \boldsymbol{x}_{n_c}^i - \boldsymbol{x}_{n_c}^c, \quad i = 1, \ldots, 5 \tag{3.34}$$

Notice how the coordinates of the pattern centre correspond to the coordinates of LED number 5, so for $i = 5$ it is actually true that $\boldsymbol{x}'^5 \equiv \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$. As a consequence, it is convenient to rewrite (3.34) as

$$\boldsymbol{x}'^i \equiv \boldsymbol{x}^i_{n_c} - \boldsymbol{x}^5_{n_c}, \quad i = 1, \ldots, 4 \tag{3.35}$$

The measurement equation (3.25) can now be rewritten as follows

$$\boldsymbol{x}'^i = A_{d_c d_t}(\phi, \theta + El, \psi + Az)\boldsymbol{x}^i_{n_t}, \quad i = 1, \ldots, 4 \tag{3.36}$$

The latter gives eight equations with six unknowns, namely the six variables of the P2P relative attitude and relative position. In principle, it is an over-determined system and no solution exists. Anyhow, let's write explicitly the eight equations.

Matrix $A_{d_c d_t}(\phi, \theta + El, \psi + Az)$ is obtained by means of a standard 1-2-3 Euler sequence, so by multiplying in the correct order, described in Equation (3.15), the three fundamental rotation matrices (2.2). Its form is well known and can be looked up in many books, such as Crassidis' [23], so it is here omitted for sake of brevity. The eight equations take the following form

$$x'^1_y = -\frac{fD2}{R}\Big(\cos(Az+\psi)\sin(\phi) + \sin(Az+\psi)\sin(El+\theta)\cos(\phi)\Big) \tag{3.37a}$$

$$x'^1_z = -\frac{fD2}{R}\cos(El+\theta)\cos(\phi) \tag{3.37b}$$

$$x'^2_y = \frac{fD3}{R}\Big(\cos(Az+\psi)\cos(\phi) - \sin(Az+\psi)\sin(El+\theta)\sin(\phi)\Big) \tag{3.37c}$$

$$x'^2_z = -\frac{fD3}{R}\cos(El+\theta)\sin(\phi) \tag{3.37d}$$

$$x'^3_y = \frac{fD2}{R}\Big(\cos(Az+\psi)\sin(\phi) + \sin(Az+\psi)\sin(El+\theta)\cos(\phi)\Big) \tag{3.37e}$$

$$x'^3_z = \frac{fD2}{R}\cos(El+\theta)\cos(\phi) \tag{3.37f}$$

$$\begin{aligned} x'^4_y = {} & \frac{fh}{R}\cos(El+\theta)\sin(Az+\psi) - \\ & -\frac{fD3}{R}\Big(\cos(Az+\psi)\cos(\phi) - \sin(Az+\psi)\sin(El+\theta)\sin(\phi)\Big) \end{aligned} \tag{3.37g}$$

$$x'^4_z = \frac{fD3}{R}\cos(El+\theta)\sin(\phi) - \frac{fh}{R}\sin(El+\theta) \tag{3.37h}$$

By observing the equations relative to pixels number 1 and number 3 ((3.37a), (3.37b) and (3.37e), (3.37f)), it can be noticed that they are the same set of two equations modulo a π rotation around the optical axis. This is reasonable given how the two LEDs are symmetrically disposed with respect to the centre of the pattern, as depicted in Figure 3.14. As a consequence, the linearly independent equations are six, so a single solution exists.

The first step is to use the pattern centre relation with the LoS angles (Equation (3.19)), to compute the Azimuth angle

$$Az = \arctan\left(\frac{p_y^c \tan\left(Az_{max}\right)}{y_{max}}\right) \tag{3.38}$$

and the Elevation angle

$$El = \arctan\left(\frac{-p_z^c \tan\left(El_{max}\right)}{z_{max}}\right) \tag{3.39}$$

Then, subtracting Equation (3.37f) from Equation (3.37d), angle ϕ can be computed

$$\frac{x_z'^2}{x_z'^3} = \frac{-D3}{D2}\tan\left(\phi\right) \Rightarrow \phi = \arctan\left(\frac{D2x_z'^2}{-D3x_z'^3}\right) \tag{3.40}$$

The latter implies an important existence condition: $\phi \neq \frac{\pi}{2}$. However, during the Final Approach phase, a 90° rotation around the roll axis of the spacecraft is extremely unrealistic, so the critical condition should be safely avoided.

By subtracting Equation (3.37b) from Equation (3.37h) and substituting the Elevation and ϕ angles just computed, angle θ can be obtained

$$\begin{aligned} \frac{x_z'^4}{x_z'^1} =& \frac{D3}{-D2}\tan\left(\phi\right) + \frac{h}{D2\cos\left(\phi\right)}\tan\left(El+\theta\right) \\ &\Rightarrow \theta = \arctan\left(\frac{\cos\left(\phi\right)}{h}\left(D2\frac{x_z'^4}{x_z'^1} + D3\tan\left(\phi\right)\right)\right) - El \end{aligned} \tag{3.41}$$

Also in this case an existence condition should be stated: $El + \theta \neq \frac{\pi}{2}$. Such situation corresponds to a condition in which the pattern centre position, defining the elevation angle, is well outside the FoV of the camera. In particular, it is like it would be seen vertically above the centre of the image plane. This condition is not expected to happen during the Final Approach phase, which requires the whole pattern to be in the FoV of the camera at all time.

Substituting (3.39), (3.41) and (3.40) into Equation (3.37d), the range R can be computed

$$R = -\frac{fD2}{x_z'^1}\cos\left(El+\theta\right)\cos\left(\phi\right) \tag{3.42}$$

Finally, summing together Equations (3.37g) and (3.37c), which share a common term but with opposite sign, and substituting all the variables already computed, angle ψ can be obtained

$$
\begin{aligned}
x_y'^4 + x_y'^2 =& \frac{fh}{R}\cos\left(El+\theta\right)\sin\left(Az+\psi\right) - \cancel{\frac{fD3}{R}\left(\cdots\right)} + \cancel{\frac{fD3}{R}\left(\cdots\right)} \\
\Rightarrow \psi =& \arcsin\left(\frac{R\left(x_y'^4 + x_y'^2\right)}{fh\cos\left(El+\theta\right)}\right) - Az
\end{aligned}
\tag{3.43}
$$

Thanks to the definition of spherical coordinates, the LoS angles combined with the range can be used to compute the target's navigation frame position with respect to the chaser's one

$$
\boldsymbol{s}_{n_c}^{n_T n_c} = \begin{bmatrix} R\cos\left(Az\right)\cos\left(El\right) \\ R\sin\left(Az\right)\cos\left(El\right) \\ -R\sin\left(El\right) \end{bmatrix}
\tag{3.44}
$$

The true P2P relative attitude matrix, $A_{d_c d_t}(\phi,\theta,\psi)$, not affected by the LoS angles, can be calculated substituting (3.40), (3.41) and (3.43) in (3.15).

Finally, using the navigation frames positions with respect to the corresponding docking frames, defined in (3.33), Equation (4) can be inverted so to compute the P2P relative position vector

$$
\boldsymbol{s}_{d_t}^{d_c d_t} = \boldsymbol{r}_{d_t}^{n_t d_t} + A_{d_c d_t}^T\left(r_{d_c}^{d_c n_c} - \boldsymbol{s}_{n_c}^{n_T n_c}\right)
\tag{3.45}
$$

3.4 Image Processing

While the chaser moves during the Final Approach phase, it takes pictures to perform a Vision-Based Navigation. These pictures must be processed with some kind of algorithm, such as a SIFT [14] or a RANSAC [11] algorithm, in order to recognize the LEDs and measure the corresponding position in the image plane. The image-processing step is quite delicate because it must be able to account for environmental noise sources such as light reflection, Earth's albedo or direct Sun light. Furthermore, such step is quite expensive from the computational point of view, which might be an issue for CubeSats, in which the hardware capabilities are limited.

To counteract the computational load issue, Dr. Pirat in [6] decided to use the Blob analysis algorithm of the MATLAB ® Computer Vision System toolbox, and performed laboratory tests to asses the performance and robustness of such algorithm when paired to the camera chosen for the research [38] [39]. The tests demonstrate how the pixel detection is successful in all the operating ranges and in presence of simulated environmental noise. Furthermore, the detection error in pixels is estimated.

For this research, there has been no possibility to perform similar tests in a laboratory so to evaluate the detection capabilities of the two new patterns (Pattern 2 and Pattern 3). However, being the camera simulated the same as in [6], and being the new patterns comparable to Pirat's, it is reasonable to assume that the image processing performance are still satisfactory, thus allowing to skip such step in the proposed docking simulator.

For sake of completeness, the working principle of the Blob analysis algorithm will be now explained. Then, the technique used to generate noisy pixels positions is described.

3.4.1 Blob Analysis

The working principle of this algorithm is to compute the coordinates of blobs of pixels in a binary image, obtained by thresholding grayscale images. A grayscale image is such that at each pixel can be assigned a value varying between 0 and 1, with 0 meaning black pixel and 1 meaning white pixel. Ideally, each of the pictures taken by the chaser should be black, a part for five withe pixels, corresponding to the five LEDs. In practice, the diffusion properties of light impliy that each LED generates a blob of pixels which is whiter towards the centre and fades to black towards the edges. When the VBN system is initialized at a range of -5m, for example, the LEDs are most likely seen as a unique blob of pixels, approximately at the centre of the image, which varies in shades of greys.

The thresholding process is required to divide bigger blobs in sub-blobs that can help distinguish different but partially overlapped sources of light. The binary image is obtained with these technique, which works like a sort of high pass filter: a threshold t between 0 and 1 is chosen and every pixel "whiter" then the threshold is set to be perfectly white, while the others are set to be black.

Once the thresholding is complete, a series of white blobs in a sea of black is obtained. The coordinates of each of these blobs are computed by the Blob analysis algorithm. The geometric features of the pattern are then used to understand which group of five blobs is actually the LEDs' pattern. Such features are, for example, the length ratios of LEDs position with respect to the centre, or the angles between specific segments defined by the pattern. This features have expected values independent from the range, so are the perfect tool to detect the pattern. Furthermore, an a priori state estimation coming from previous iterations of the algorithm can be used to define a Region of Interest (ROI) in which it is expected to observe the LEDs. This is extremely useful when more then one set of five blobs is recognized as a possible pattern: by computing the distance of each set from the a priori estimate, it is possible to identify the group of five blobs which most likely corresponds to the pattern

Once the LEDs coordinates are obtained, they are fed to the EKF to perform or update the state estimation by means of the measurement equation described in 3.25.

The tuning of the threshold is of paramount importance, appreciable by looking at Figure 3.15.

- A high value of t can be useful to cancel out all the low intensity sources of light noise, implying that only the brightest pixels are actually captured. At the same time, it can worsen the pixel position estimation performance. In fact, CCDs are intrinsically affected by a white noise, which is typically 1 pixel [6]; this effect can be counteracted by detecting objects with a larger light footprint, so to mitigate the impact of this noise by using a bigger number of pixels for the computation of the blob's coordinates. In far range, which is the most delicate for LEDs detection, each LED occupies just a few pixels: a high value of t leads to a very noisy image.
- A low value of t, however, might imply that a blob corresponding to more then one LED is converted to a single white blob, impeding the detection of all the light sources.

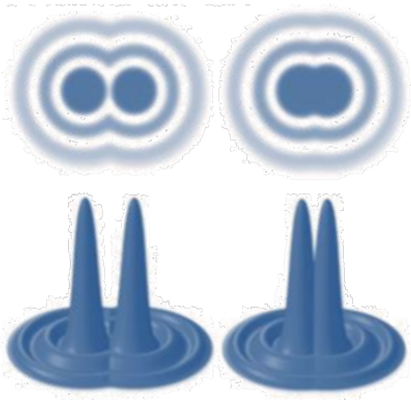


Figure 3.15: Visualization of the problem associated to light diffusion: correctly thresholding an image has a key role in the LEDs detection process [41]

Another important parameter associated to the just cited issues is the light intensity of the LEDs:

- a high intensity is desirable to detect LEDs in far range;
- a too high intensity, however, makes it impossible to distinguish between LEds when they are seen close to each other.

The most delicate moment for the detection is certainly when the VBN system is being initialized (poor a prioiri estimate) and the two satellites are still far apart (LEDs are seen very close to each others). Pirat tuned all the parameters such that the detection is successful in far range also in presence of environmental disturbances, so it can be safely assumed that the proposed image processing algorithm would work also with slightly different patterns.

A summary of the described algorithm is for sake of clarity presented in Table 3.1.

Table 3.1: The LEDs detection algorithm

Prediction	Define an ROI using an a prioir state estimation
Centroid detection	1. Apply the thresholding
	2. Select the blobs belonging to the ROI
Geometrical features	1. Compute the geometrical features for each group of five blobs
	2. Select the set of blobs whose geometrical features match with the expected ones
LEDs selection	Select the pattern with the smallest distance from the a priori one

3.4.2 Image Generation

The solution adopted in this research to obtain the pixel measurements needed to perform relative navigation is to create fake images using the real relative state available in the simulator. To do so, the symbolic version of the measurement equation (3.25) is fed with the P2P relative attitude vector $\boldsymbol{\alpha}^{d_c d_t}$ and the P2P relative position vector $\boldsymbol{s}_{d_t}^{d_c d_t}$. These two vectors are taken from a plant that simulates the complete nonlinear version of the P2P coupled dynamics 2.33, thus representing the real relative state and not an estimation.

As a consequence, the pixels obtained at this stage correspond to the exact positions of the LEDs in the image plane given the current relative state: using these values is highly unrealistic and would obviously imply a close to perfect state estimation when fed to the EKF. To make the solution more realistic, the images are corrupted with a white noise source to simulate both the CCD error and the blobs coordinates computation error. This two sources of noise will be accounted for in a single parameter modeled as a zero-mean Gaussian white noise, which will be hereafter referred to as Pixel Error

$$\varepsilon_{pix} \sim \mathcal{N}(0, \sigma_{pix}^2) \tag{3.46}$$

where σ_{pix} is the standard deviation associated to the distribution. The pixels coordinates $p_{y,real}^i$ and $p_{z,real}^i$, with $i = 1, \ldots, 5$, are corrupted by adding to each p^i a number picked randomly from the the distribution (3.46)

$$p_{noisy}^i = p_{real}^i + \varepsilon_{pix} \tag{3.47}$$

One of the scopes of this research has been to understand how ε_{pix} affects the Final Approach phase by changing the value of the associated standard deviation. A summary of the Image Generation algorithm implemented is described in Table 3.2.

Table 3.2: The Image Generation algorithm

State extraction	Extract vectors $\boldsymbol{\alpha}^{d_c d_t}$ and $\boldsymbol{s}_{d_t}^{d_c d_t}$ from the full P2P relative state vector $\boldsymbol{x}$
Generate ideal pixels	Use $\boldsymbol{\alpha}^{d_c d_t}$ and $\boldsymbol{s}_{d_t}^{d_c d_t}$ in Equation 3.25 to generate the ideal pixels positions p_{real}^i
Generate noisy pixels	Corrupt the ideal pixels by adding to each p_{real}^i a Pixel Error ε_{pix}, so to generate the noisy positions p_{noisy}^i

Simulated images

Here, some images of the three patterns generated with the algorithm just described are presented. The nominal standard deviation of the Pixel Error, following in a conservative way the results obtained by means of the test performed in [6], is set to

$$\bar{\sigma}_{pix} = 0.1 \text{ pixel} \tag{3.48}$$

and is accounted for following (3.47). As a consequence, the the accuracy of the analytical solution at various ranges can be assessed.

For each pattern, four images are here presented, obtained from the relative states described in Table 3.3.

Table 3.3: Relative states used for the generation of test images

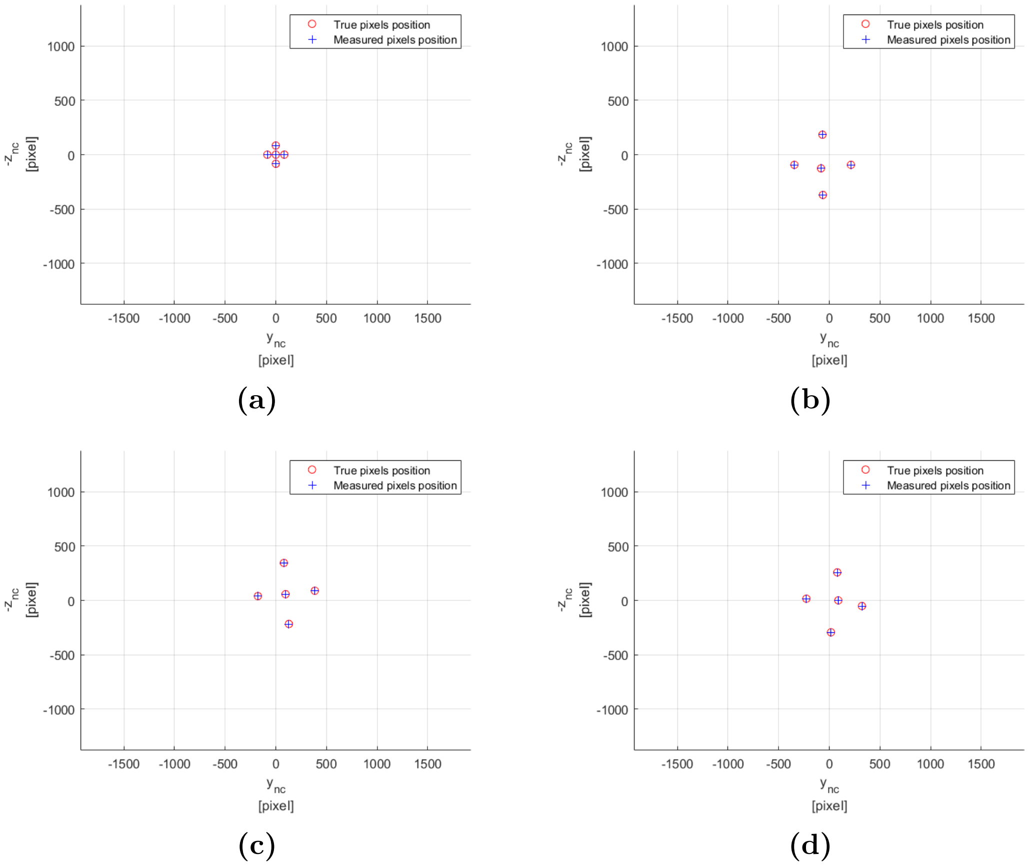
Image (a)	$\boldsymbol{\alpha}^{d_c d_t} = \begin{bmatrix} 0° \\ 0° \\ 0° \end{bmatrix}$	$\boldsymbol{s}_{d_t}^{d_c d_t} = \begin{bmatrix} -0.5 \\ 0 \\ 0 \end{bmatrix} \text{m}$
Image (b)	$\boldsymbol{\alpha}^{d_c d_t} = \begin{bmatrix} 0° \\ 0° \\ 0° \end{bmatrix}$	$\boldsymbol{s}_{d_t}^{d_c d_t} = \begin{bmatrix} -0.1 \\ 0.01 \\ -0.02 \end{bmatrix} \text{m}$
Image (c)	$\boldsymbol{\alpha}^{d_c d_t} = \begin{bmatrix} 5° \\ -6° \\ -7° \end{bmatrix}$	$\boldsymbol{s}_{d_t}^{d_c d_t} = \begin{bmatrix} -0.1 \\ 0 \\ 0 \end{bmatrix} \text{m}$
Image (d)	$\boldsymbol{\alpha}^{d_c d_t} = \begin{bmatrix} -7° \\ 5° \\ 6° \end{bmatrix}$	$\boldsymbol{s}_{d_t}^{d_c d_t} = \begin{bmatrix} -0.5 \\ -0.02 \\ 0.01 \end{bmatrix} \text{m}$

The four images for the three patterns can be seen in Figures 3.16, 3.17 and 3.18. The accuracy of the analytical solution for each pattern is described in Tables 3.4, 3.5 and 3.6.

Analyzing the accuracy of the analytical solution is already possible to understand which of the three patterns will perform better. Let's observe that

- for all patterns, the accuracy in the computation of the position improves as the range diminish, which is perfectly reasonable because the closer the satellites are, smaller is the impact of the detection error,
- in far range, the roll angle computation for Pattern 2 is considerably less accurate then the computation of the other two angles,
- Pattern 1 seems to perform better then the other two, especialy for what regards the angle computation,
- Pattern 2 performs similarly to Pattern 1 in terms of position computation, but is less accurate for the computation of the angles.

The lower spacing between the LEDs, which is characteristic of Pattern 2, worsen the observability of the relative orientation angles.

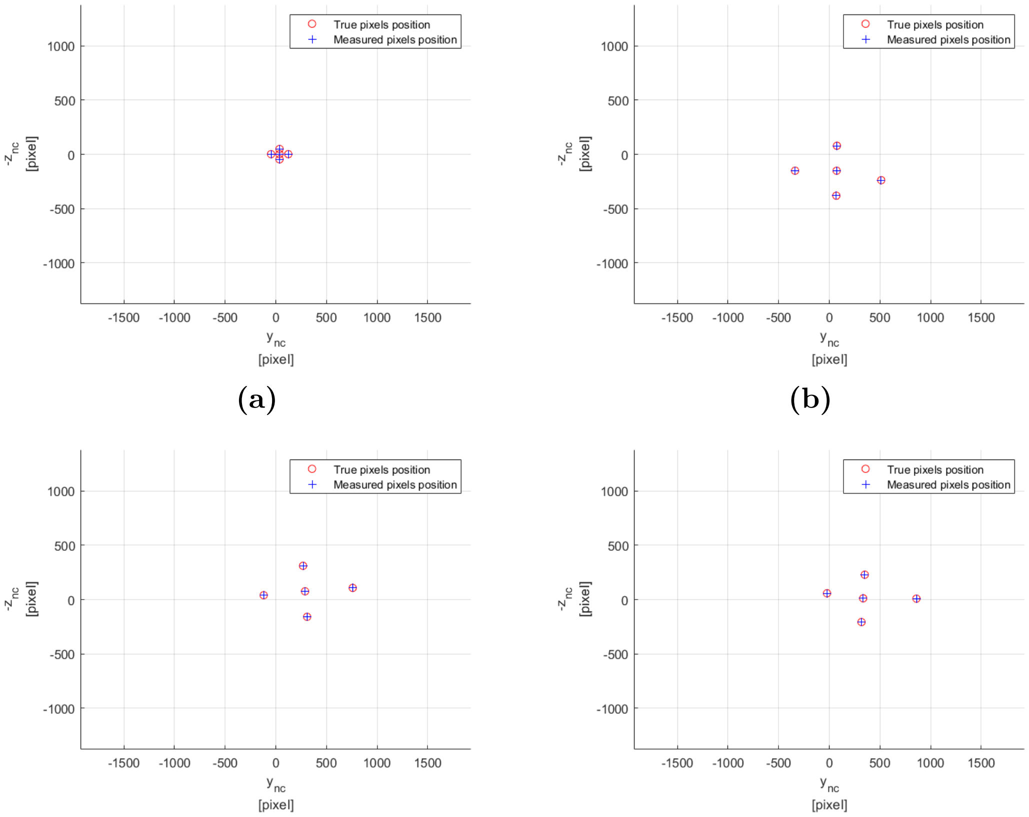


(a) **(b)** **(c)** **(d)**

Figure 3.16: Pattern 1 - See Table 3.3 for the relative state used for each picture

Table 3.4: Patter 1 - Accuracy of the analytical solution for the four images in Figure 3.16

Image (a)	$\varepsilon_\alpha = \begin{bmatrix} 0.1480° \\ -0.0081° \\ -0.0261° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 1.9 \cdot 10^{-5} \\ 2.7 \cdot 10^{-4} \\ -1.0 \cdot 10^{-4} \end{bmatrix} \mathrm{m}$
Image (b)	$\varepsilon_\alpha = \begin{bmatrix} -0.0081° \\ 0.0120° \\ 0.0203° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 6.3 \cdot 10^{-5} \\ -5.4 \cdot 10^{-5} \\ 3.8 \cdot 10^{-5} \end{bmatrix} \mathrm{m}$
Image (c)	$\varepsilon_\alpha = \begin{bmatrix} -0.0088° \\ 0.0077° \\ -1.3° \cdot 10^{-4} \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 2.2 \cdot 10^{-5} \\ -8.4 \cdot 10^{-6} \\ 2.0 \cdot 10^{-6} \end{bmatrix} \mathrm{m}$
Image (d)	$\varepsilon_\alpha = \begin{bmatrix} 0.0104° \\ -0.0101° \\ -0.0203° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} -8.2 \cdot 10^{-5} \\ -8.5 \cdot 10^{-5} \\ 9.6 \cdot 10^{-5} \end{bmatrix} \mathrm{m}$



(a)

(b)

(c)

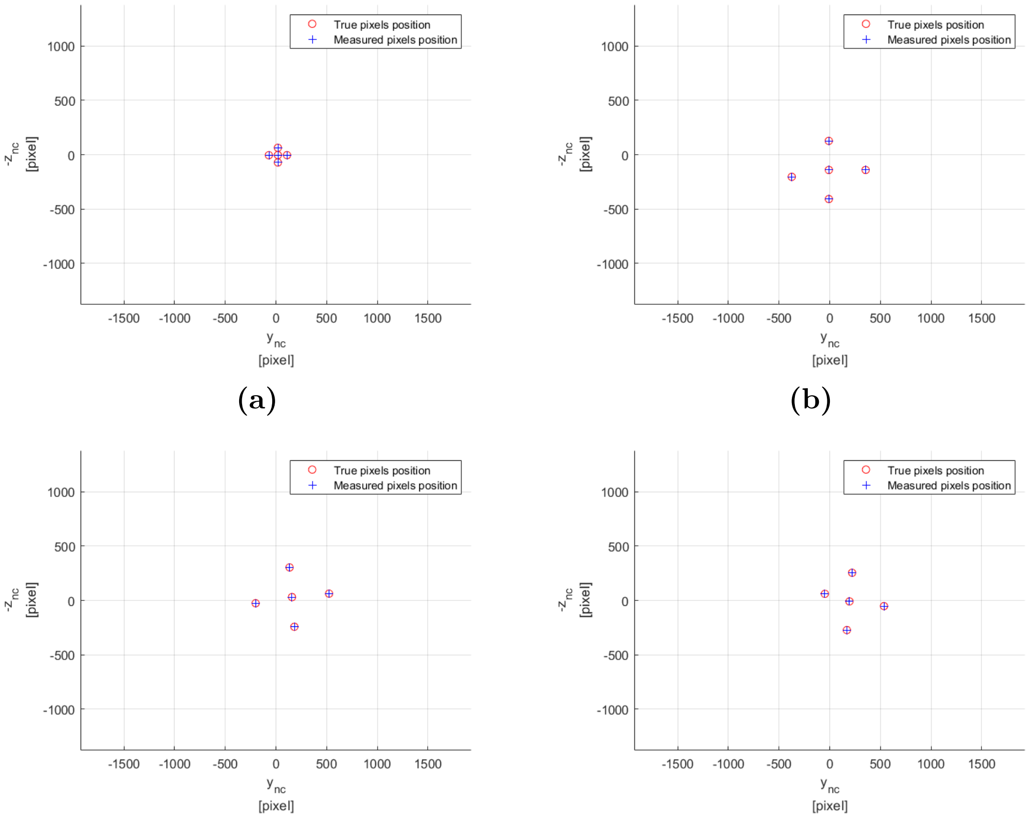
(d)

5

Figure 3.17: Pattern 2 - See Table 3.3 for the relative state used for each picture

Table 3.5: Pattern 2 - Accuracy of the analytical solution for the four images in Figure 3.17

Image (a)	$\varepsilon_\alpha = \begin{bmatrix} -0.3680° \\ 1.0613° \\ -0.3950° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} -4.3 \cdot 10^{-3} \\ 3.2 \cdot 10^{-3} \\ 9.3 \cdot 10^{-3} \end{bmatrix} \mathrm{m}$
Image (b)	$\varepsilon_\alpha = \begin{bmatrix} -0.1929° \\ 0.2978° \\ -0.3494° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 7.2 \cdot 10^{-4} \\ 5.5 \cdot 10^{-4} \\ 3.0 \cdot 10^{-4} \end{bmatrix} \mathrm{m}$
Image (c)	$\varepsilon_\alpha = \begin{bmatrix} 0.0876° \\ -0.1066° \\ -0.3264° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 2.7 \cdot 10^{-5} \\ 4.5 \cdot 10^{-4} \\ -1.3 \cdot 10^{-4} \end{bmatrix} \mathrm{m}$
Image (d)	$\varepsilon_\alpha = \begin{bmatrix} 0.1306° \\ -0.3563° \\ -0.1492° \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} -2.0 \cdot 10^{-4} \\ 1.5 \cdot 10^{-4} \\ -6.4 \cdot 10^{-4} \end{bmatrix} \mathrm{m}$



(a)

(b)

(c)

(d)

Figure 3.18: Pattern 3 - See Table 3.3 for the relative state used for each picture

Table 3.6: Pattern 3 - Accuracy of the analytical solution for the four images in Figure 3.18

Image (a)	$\varepsilon_\alpha = \begin{bmatrix} -0.1236^\circ \\ 0.2078^\circ \\ -0.0760^\circ \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 1.7 \cdot 10^{-3} \\ -6.9 \cdot 10^{-4} \\ 1.8 \cdot 10^{-3} \end{bmatrix} \mathrm{m}$
Image (b)	$\varepsilon_\alpha = \begin{bmatrix} -0.0269^\circ \\ -0.0202^\circ \\ 0.0309^\circ \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 1.2 \cdot 10^{-4} \\ 3.6 \cdot 10^{-5} \\ 6.1 \cdot 10^{-5} \end{bmatrix} \mathrm{m}$
Image (c)	$\varepsilon_\alpha = \begin{bmatrix} -0.0036^\circ \\ 0.0137^\circ \\ -0.0194^\circ \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 1.2 \cdot 10^{-5} \\ -2.6 \cdot 10^{-5} \\ 2.7 \cdot 10^{-5} \end{bmatrix} \mathrm{m}$
Image (d)	$\varepsilon_\alpha = \begin{bmatrix} 0.0150^\circ \\ 0.0249^\circ \\ 0.0492^\circ \end{bmatrix}$	$\varepsilon_s = \begin{bmatrix} 1.2 \cdot 10^{-4} \\ 3.6 \cdot 10^{-5} \\ 6.1 \cdot 10^{-5} \end{bmatrix} \mathrm{m}$

Chapter 4

Estimation and Control

The estimation and control technique used for this research will be now presented. The choice of adopting a classical approach is due to the focus of the thesis in the performance of the Vision-Based algorithm rather then on the optimization of estimation and control. In future works, the implementation of nonlinear observers and/or controllers could be investigated so to understand whether such solutions yield improvements in state estimation accuracy and systems control.

In the first section the working principle of Extended Kalman Filters is explained. Then, the Linear Quadratic Regulator problem is described, with particular attention to the handover phase that is required for the initialization of the VBN algorithm.

4.1 Extended Kalman Filter

An Extended Kalman Filter (EKF), as its own name says, is an extension of the most common sequential state estimator for stochastic processes, the Kalman Filter (KF). The basic working principle of a KF is to perform state estimation for linear systems that can be assumed are effected by non-correlated, zero-mean Gaussian white noises. In the linear case, a Gaussian input yields a Gaussian output. This is not necessarily true for nonlinear models, requiring an extension given by the EKF. A complete and thorough derivation of both type of filters can be found in [42].

Let's consider a generic nonlinear model of the form

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) + G(t)\boldsymbol{w}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{h}(\boldsymbol{x}(t), t) + \boldsymbol{v}(t) \end{aligned} \tag{4.1}$$

where the nonlinear functions $\boldsymbol{f}$ and $\boldsymbol{h}$ are assumed to be continuously differentiable, and the noises vectors are non-correlated, zero-mean Gaussian white noises

$$\boldsymbol{w}(t) \sim \mathcal{N}(0, Q) \tag{4.2a}$$

$$\boldsymbol{v}(t) \sim \mathcal{N}(0, R) \tag{4.2b}$$

$$\mathbb{E}\{\boldsymbol{w}(t)\boldsymbol{v}^T(t)\} = 0 \tag{4.2c}$$

If the assumption that the estimated state is sufficiently close to the real state is true, such that the error dynamics can be described by means of a linearized first-order Taylor expansion of the model in (4.1), than the Extended Kalman Filter algorithm can be applied. In fact, such algorithm relies on subsequent estimations of the state using known inputs $\boldsymbol{u}$ and measurements $\boldsymbol{y}$ that are fed to the model, which gets linearized around the current estimate. In other words, an *a priori* estimate is used in the model itself to generate an *a posteriori* estimate, given the current inputs and measurements.

The model (4.1) resembles the combination of the P2P dynamics from (2.33) and the measurement equations from (3.27). However, even though the P2P dynamics model is continuous, measurements in real systems are available in a discrete fashion. In fact, the pixels positions in the image plane are not continuously available, but are computed only after the picture is taken and the execution of the image processing algorithm described in Table 3.1 is completed. Even in the simulator implemented in this thesis, which skips the image processing step as described at the end of previous chapter, the measurements will be available discretely. For this reason, a discrete measurement model more accurately describes the situation

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k, \qquad \boldsymbol{v}_k \sim \mathcal{N}(0, R_k) \tag{4.3}$$

where k describes the current time step. The Kalman filtering technique that will be now described, which features a continuous state transition model and a discrete measurement model, is referred to as *continuous-discrete* Extended Kalman Filter.

4.1.1 Continuous-Discrete Extended Kalman Filter

In Section 2.4.2, a linearization of the P2P dynamics around a single operational point describing the docking conditions yielded matrices A and B. In the most general form of the EKF algorithm, instead, the linearization is performed iteratively around the current estimate for both dynamic and measurement models.

Let's define the state estimate $\hat{\boldsymbol{x}}$ such that

$$\tilde{\boldsymbol{x}} = \hat{\boldsymbol{x}} - \boldsymbol{x} \tag{4.4}$$

where $\tilde{\boldsymbol{x}}$ is the state estimation error. It is possible to distinguish between the *a priori* estimate, $\hat{\boldsymbol{x}}^-$, and the *a posteriori* estimate $\hat{\boldsymbol{x}}^+$. Its discrete version will be $\hat{\boldsymbol{x}}_k$.

The state transition matrix F and the measurement matrix H_k, in general, can be defined as

$$F(\hat{\boldsymbol{x}}) \equiv \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}},\boldsymbol{u}} \tag{4.5a}$$

$$H_k(\hat{\boldsymbol{x}}_k) \equiv \left.\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_k} \tag{4.5b}$$

The first step of the algorithm is the initialization step and will be performed just once. In the following items, superscripts $^-$ and $^+$ indicate quantities obtained from the current estimate and the updated estimate respectively.

1. Assuming a first state estimation is available (initial conditions), $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_0$, an initial covariance matrix can be defined as the expectation of the initial estimation error
$$P_0 = \mathbb{E}\{\tilde{\boldsymbol{x}}_0 \tilde{\boldsymbol{x}}_0^T\} \tag{4.6}$$

2. The discrete measurement matrix $H_k^-(\hat{\boldsymbol{x}}_k^-)$ can be obtained from a linearization around the current estimate.

3. The gain, needed in the update step to balance the contribute of a new measurement, can be now computed as
$$K_k^- = P_k^- H_k^{-T} \left[H_k^- P_k^- H_k^{-T} + R_k\right]^{-1} \tag{4.7}$$

4. The state estimate and the model covariance can now be updated
$$\begin{aligned} \hat{\boldsymbol{x}}_k^+ &= \hat{\boldsymbol{x}}_k^- + K_k^- \left[\boldsymbol{y}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-)\right] \\ P_k^+ &= \left[\mathbb{I} - K_k^- H_k^-\right] P_k^- \end{aligned} \tag{4.8}$$

5. The dynamic model and the covariance are propagated forward in time as follows
$$\begin{aligned} \dot{\hat{\boldsymbol{x}}} &= \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u}) \\ \dot{P} &= FP + PF^T + GQG^T \end{aligned} \tag{4.9}$$
where F is obtained from a linearization of the dynamic model around the current best estimate.

6. Repeat from step 2.

The computational burden of this algorithm can be reduced by performing an a priori linearization of the dynamic model around a given operational point and use this linearization rather then performing such costly operation every time a new best estimate is available. This would correspond, for example, to using matrices A and B defined in (2.38c) and (2.38d).

The algorithm is summarized in Table 4.1.

Table 4.1: The continuous-discrete Extended Kalman Filter algorithm

Define model	$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + G(t)\boldsymbol{w}(t), \quad \boldsymbol{w}(t) \sim \mathcal{N}(0, Q)$
	$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(0, R_k)$
Initialize	$\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_0$
	$P_0 = \mathbb{E}\{\tilde{\boldsymbol{x}}_0 \tilde{\boldsymbol{x}}_0^T\}$
Compute gain	$K_k^- = P_k^- H_k^{-T} \Big[H_k^- P_k^- H_k^{-T} + R_k \Big]^{-1}$
	where $H_k(\hat{\boldsymbol{x}}_k^-) \equiv \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}}\Big\rvert_{\hat{\boldsymbol{x}}_k^-}$
Update	$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + K_k^- \Big[y_k - h_k(\hat{\boldsymbol{x}}_k^-) \Big]$
	$P_k^+ = \Big[\mathbb{I} - K_k^- H_k^- \Big] P_k^-$
Propagate	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})$
	$\dot{P} = FP + PF^T + GQG^T$
	where $F(\hat{\boldsymbol{x}}) \equiv \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big\rvert_{\hat{\boldsymbol{x}}, \boldsymbol{u}}$

4.2 Linear Quadratic Regulator

A Linear Quadratic Regulator (LQR) is a very efficient and widely used type of linear controller because it allows to perform in a convenient way a state feedback control of a system by computing a gain that optimally balances the trade-off described by a cost function [43].

Let's consider a linear dynamic model in state space form, similar to the one obtained linearizing the P2P dynamics (2.37)

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u} \tag{4.10}$$

A deterministic integral quadratic cost function can be defined

$$J = \int_0^\infty \Big(\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{u}^T R \boldsymbol{u} \Big) dt \tag{4.11}$$

where Q and R are weight matrices which balance the trade-off in the optimization between the performance of the controller and the effort required respectively.

The optimal control action for (4.10) given (4.11), so the control action which best balances performance and effort according to the weights Q and R by minimizing the cost function J, is given by the full-state feedback action

$$\boldsymbol{u} = -K_r \boldsymbol{x} \tag{4.12}$$

where

$$K_r = R^{-1} B^T X \tag{4.13}$$

and X is a symmetric, positive semi-definite matrix which solves the following algebraic Riccati equation

$$A^T X + XA - XBR^{-1} B^T X + Q = 0 \tag{4.14}$$

The weight matrices Q and R must be positive semi-definite. The common approach is to define them as diagonal matrices, so that each element of the diagonal describes the weight given to the corresponding state or control input. What is really important in the definition of these parameters is the relative size of the values in the two diagonals, and not the absolute values. As a consequence, a possible controller design approach is to set all the elements of one diagonal to 1 and then tune the second diagonal until the desired performances are achieved.

This type of controller can be used to control the linearized version of the P2P coupled dynamics defined in (2.37). Such model is an approximation, since it is obtained from a first order Taylor expansion. However, it can be shown that the nonlinearity involved are small, so it is reasonable to expect that the LQR controller can yield satisfactory performance [6].

Notice that the full state feedback control law defined in (4.12) requires the real full state of the system to be available. This, for real systems, is never possible: the state has to be somehow measured or estimated. In this research, as already described in Section 4.1.1, such estimation is performed by means of an Extended Kalman Filter, so the control law that has actually been implemented is of the form

$$\boldsymbol{u} = -K_r \hat{\boldsymbol{x}} \tag{4.15}$$

where the estimated state $\hat{\boldsymbol{x}}$ is used.

This type of solution, in which the state estimated by a Kalman filter is then used in a Linear Quadratic Regulator, is usually referred to as Linear Quadratic Gaussian (LQG). LQG controllers benefit of an important property, the *separation principle*, which states that the poles of the estimator and of the controller do not interact and the two blocks can thus be designed and tested independently.

To conclude, the reference error $\boldsymbol{\varepsilon_x}$ must be defined. In fact, the control action defined in (4.15), if used as it is, will control the state of the system directly to zero. However, for safety reasons it might be required to divide the Final Approach translation in independent segments (see Chapter 5), so to have some intermediate station keeping points. As a consequence, in order to generate the proper control action, the controller must be fed with a reference error defined as

$$\boldsymbol{\varepsilon_x} = \boldsymbol{x}_{desired} - \hat{\boldsymbol{x}} \tag{4.16}$$

so that the LQR will control the reference error to zero.

The control action in (4.15) becomes

$$\boldsymbol{u} = K_r \boldsymbol{\varepsilon_x} \tag{4.17}$$

This type of solution is usually referred to as Linear Quadratic Tracker, but in this research it will be simply referred to as Linear Quadratic Regulator.

4.3 GNC Handover

A very delicate and crucial phase of the mission studied in this research is the moment of the so called handover. The chasing CubeSat starts its journey at a distance from the target of the order of tens of kilometers (Figure 4.1). During the first part of the RVD mission, its goal is to reduce such relative distance to the order of hundreds of meters by using lower and faster orbits to diminish the phase angle between the two spacecraft. This part ends with the Closing phase. During Closing the two satellites communicate by means of an ISL system and the target is thus able to reduce the relative distance thanks to a CDGPS solution (see Section 1.1.1). For what regards the attitude control, in this phase the two spacecraft simply control their absolute attitude with respect to their local orbital frame by means of sensors specifically designed for this purpose such as Sun Sensors or Star Trackers.

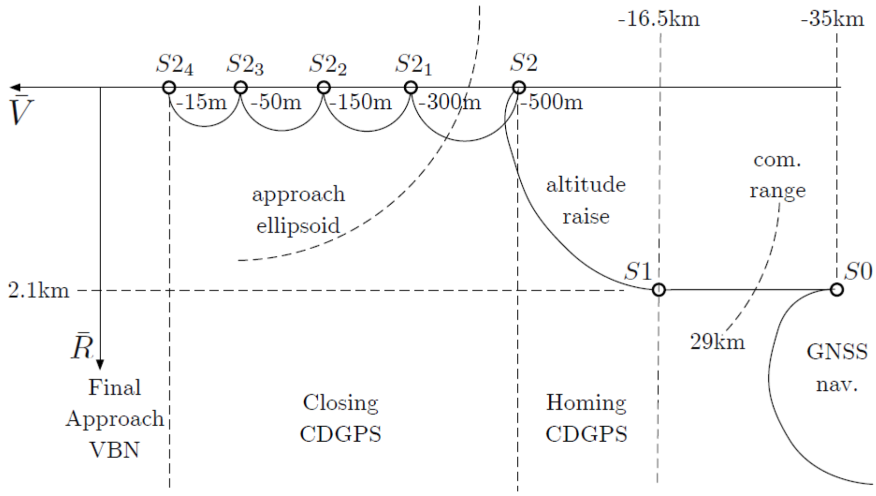


Figure 4.1: The complete RVD mission designed and simulated in [6]

Then, the Final Approach (FA) phase starts, and the point in which this happens will be referred to as $SK0$, where the acronym SK stands for Station Keeping. In the transition between Closing and FA, the sensors used to perform relative navigation change, so there will be an interval of time in which an handover between the GNC system that guided the chaser all the way to $SK0$ and the VBN system occurs. The chaser will remain in $SK0$ until the handover is complete, and this explains why it is referred to as a station keeping point.

$SK0$ can be defined by means of a nominal vector which describes the relative distance of the two spacecraft in such point. For this research it is assumed the following

$$\boldsymbol{r}_{d_t}^{SK0} = \bar{\boldsymbol{s}}_{d_t}^{d_c d_t} = \begin{bmatrix} -5 \\ 0 \\ 0 \end{bmatrix} \mathrm{m} \tag{4.18}$$

Furthermore, the desired nominal relative state for $SK0$ can be defined. Such nominal state can be described with the following conditions: the two spacecraft are aligned with the same attitude and have no residual relative angular nor linear velocities.

$$\bar{\boldsymbol{\alpha}}^{d_c d_t} = \begin{bmatrix} 0^\circ \\ 0^\circ \\ 0^\circ \end{bmatrix} \qquad \bar{\boldsymbol{\omega}}_{d_t}^{d_c d_t} = \begin{bmatrix} 0^\circ \\ 0^\circ \\ 0^\circ \end{bmatrix} \frac{1}{\mathrm{s}} \qquad \dot{\bar{\boldsymbol{s}}}_{d_t}^{d_c d_t} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \frac{\mathrm{m}}{\mathrm{s}} \tag{4.19}$$

The nominal conditions just described will be used for the initialization of the EKF, as explained in Table 4.1, so to set the initial state estimate $\boldsymbol{x}_0$.

However, the GNC system that brings the chaser in $SK0$ is not ideal, so it will not be able to achieve the nominal state just described. A position and attitude error ($\Delta \boldsymbol{s}$ and $\Delta \boldsymbol{\alpha}$), together with some residual linear and angular velocities ($\Delta \dot{\boldsymbol{s}}$ and $\Delta \boldsymbol{\omega}$), will always be present in $SK0$. An estimate of this deviation from the nominal initial conditions is needed for the definition of the initial covariance P_0 of the EKF, as explained in Table 4.1.

The amplitude of this errors and of the residual velocities can strongly affect the initialization process of the VBN system: the ROI used in the LEDs detection algorithm described in Table 3.1 will be larger if the expected Δ are big, and this implies an increase in the detection error ε_{pix}. As a consequence, the estimation performance of the EKF worsen, implying the generation of an imprecise control action (4.15). The combination of this effects might even lead to an abrupt divergence of the chaser which could result in a collision of the two spacecraft. Alternatively, the initialization process could completely fail and the docking mission might have to be aborted.

Chapter 5

Final Approach Simulator

In this chapter all the details relative to the simulations performed are presented, together with the results obtained.

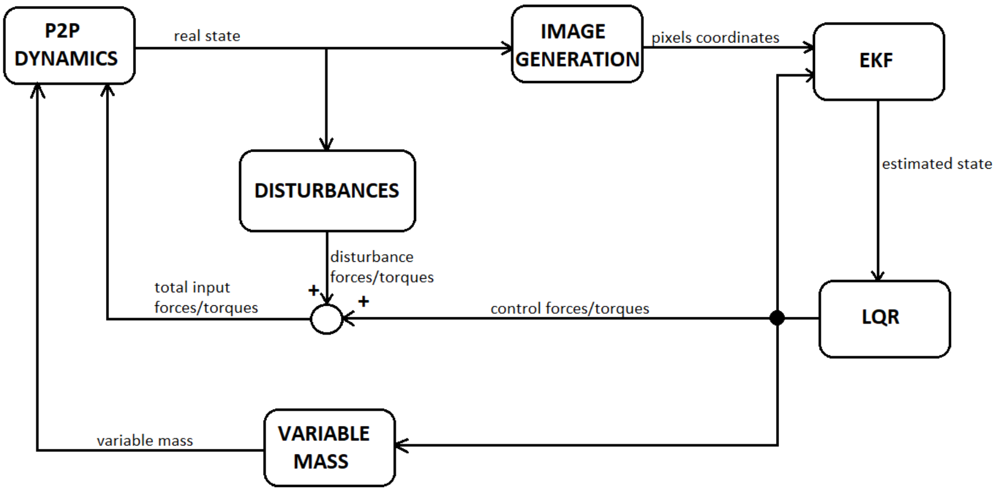


Figure 5.1: The Final Approach simulator

First of all the mission scenario, i.e. the description of how the Final Approach translation is executed, is described. Then, the orbital and CubeSat parameters used are presented and motivated. It follows a brief description of the simulator designed, depicted in Figure 5.1, with an explanation of the blocks it features and the tuning parameters used for the EKF and the LQR. Finally, the simulations performed to investigate handover performance, pattern performance and the role of the pixel detection error, are described, and the obtained results are commented.

5.1 Mission Scenario

A Vision-Based Navigation (VBN) system is the best tool to perform relative navigation of two spacecraft in close proximity because it can give the highest navigation accuracy. The SPHERES nanosatellites tested on board the ISS, which use the algorithm developed in [9], are able to achieve the following close range accuracy [36]:

- error lower then 0.5 cm in the approach direction when the chaser is in a 1 m range from the target,
- error lower then 0.5 cm for the lateral displacements when the chaser in a 30 cm range from the target, slightly higher then 0.5 cm when the chaser is between 30 cm and 60cm from the target,
- error lower then 1° for each axis when the chaser is in 60 cm range from the target.

A linear scaling analysis of the docking mechanism used by the European Space Agency's (ESA) Autonomous Transfer Vehicle (ATV) performed in [35] led to the requirement that a CubeSat docking mechanism shall be able to cope with a 1 cm and 2° misalignment for each axis. For what regards the accuracy required during the Final Approach phase, thus the translation prior to docking, a rule of thumb is that the estimation shall be better then 1% of the range while the control accuracy shall be better then 10% of the range.

The Final Approach phase will start from the Initial Station Keeping point $SK0$, defined in (4.18). In this research, an approach along the $\bar{V}$ axis of the LVLH frame defined in Section 2.2.2 is considered, and the translation will be in a quasi straight-line fashion. The chaser keeps its position while the handover is being performed and the EKF is converging. Recalling the definition of the reference error given in (4.16), the translation will start only when both the following conditions are met:

- the reference error for the relative position remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_s} \in [-5{,}5]$cm for at least 60 consecutive seconds,
- the reference error for the relative velocity remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_{\dot{s}}} \in [-1{,}1]\mathrm{cm\,s^{-1}}$ for at least 60 consecutive seconds.

By imposing such condition, the LQG system has time to converge and stabilize the satellite around $SK0$, so that when the translation starts the state estimation is already sufficiently accurate.

The translation is then divided in six consecutive steps, which consequently give rise to 6 station keeping points along the $\bar{V}$ axis, as depicted in Figure 5.2. Let's

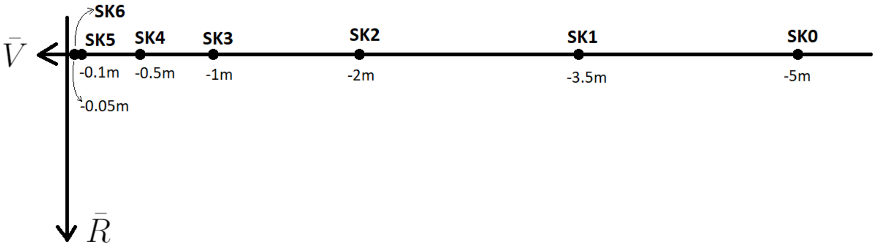


Figure 5.2: The Final Approach mission scenario

call the first two SK points, namely $SK1$ and $SK2$, Far Rage Points (FRPs). The next three points, $SK3$, $SK4$ and $SK5$, will instead be referred to as Close Range Point (CRPs). Finally, $SK6$ is referred to as the Final Point. The navigation rules applied are listed below.

- The chaser remains in a FRP until both the following conditions are met:
  - the reference error for the relative position remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_s} \in [-5,5]\text{cm}$ for at least 30 consecutive seconds,
  - the reference error for the relative velocity remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_{\dot{s}}} \in [-1,1]\text{cm}\,\text{s}^{-1}$ for at least 30 consecutive seconds.
- The chaser remains in a CRP until both the following conditions are met:
  - the reference error for the relative position remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_s} \in [-1,1]\text{cm}$ for at least 30 consecutive seconds,
  - the reference error for the relative velocity remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_{\dot{s}}} \in [-1,1]\text{mm}\,\text{s}^{-1}$ for at least 30 consecutive seconds.
- The chaser remains in $SK6$ until both the following conditions are met:
  - the reference error for the relative position remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_s} \in [-5,5]\text{mm}$ for at least 120 consecutive seconds,
  - the reference error for the relative velocity remains in a range defined as $\boldsymbol{\varepsilon}_{\boldsymbol{x}_{\dot{s}}} \in [-0.5,0.5]\text{mm}\,\text{s}^{-1}$ for at least 120 consecutive seconds.

When the condition for $SK6$ is finally met, the Soft Docking described at the beginning of Section 3.2 can start. Notice that the requirements imposed for the Final Station Keeping point are different from the misalignment constraints defined at the beginning of this section. In fact, the misalignment depends on the real relative state, which is not available in a real mission and is thus unconstrainable. To satisfy the accuracy required at docking, the best option is to impose strict requirements to the reference error, which is based on the estimated state, so to avoid the risk that a residual estimation error $\tilde{x}$ makes a successful docking unfeasible.

A proper tuning of the LQR controller weight matrices is of paramount importance to bound to 10% of the range the overshoot, which will intrinsically happen every time a new Station Keeping point is reached. The weight matrices will also have a strong role in the stabilization of the chaser around each SK point.

5.2 Simulator's Input

All the input parameters used in the simulator designed in MATLAB ® and Simulink will be now presented.

5.2.1 Orbital Parameters

The orbital parameters used are listed in Table 5.1.

Table 5.1: Orbital parameters [22]

Orbit altitude	$z_O = 400000$ m
Earth's radius	$r_E = 6378.137$ km
Orbital radius	$r_O = z_O + r_E = 6778137$ m
Standard gravitational parameter	$\mu_E = 398600.4405$ km^3/s^2
Orbital mean motion	$\omega_O = \sqrt{\frac{\mu_E}{r_O^3}} \approx 0.001131$ rad s^{-1}
Orbit inclination	$i_m = \frac{\pi}{4}$

5.2.2 CubeSat Parameters

Both chaser and target are considered parallelepipeds, as depicted in Figure 5.3. To simplify the estimation of the inertia tensor, the mass distribution is assumed to be homogeneous for both satellites. Recalling the definition of the body frame $\mathcal{F}_b$ (see Section 2.2.3), the following three lengths can be defined

$$\begin{aligned} l_x &= 0.3 \text{ m} \\ l_y &= 0.1 \text{ m} \\ l_z &= 0.1 \text{ m} \end{aligned} \tag{5.1}$$

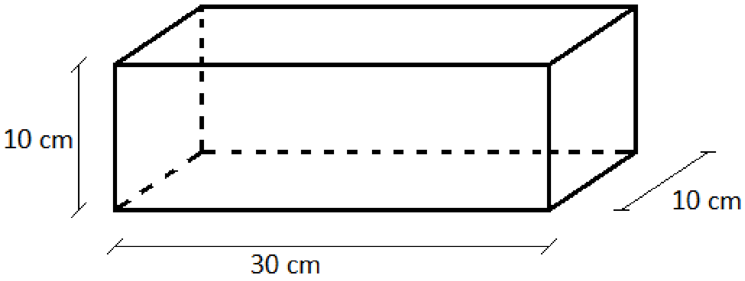


Figure 5.3: CubeSat shape and dimension

Because of the homogeneous mass distribution, the inertia tensor expressed in the body frame will be diagonal

$$I_b = \begin{bmatrix} J_{x0} & 0 & 0 \\ 0 & J_{y0} & 0 \\ 0 & 0 & J_{y0} \end{bmatrix} \tag{5.2}$$

where

$$\begin{aligned} J_{x0} &= m\frac{l_y^2 + l_z^2}{12} \\ J_{y0} &= m\frac{l_x^2 + l_z^2}{12} \\ J_{z0} &= m\frac{l_x^2 + l_y^2}{12} \end{aligned} \tag{5.3}$$

Where the mass m considered is the same wet mass for both target and chaser: $m_c = m_t = 4$ kg. Recall that the simulated P2P dynamics uses the variable mass model described in Appendix A. Recalling the relative attitude dynamics defined in Section 2.4.1, the inertia tensor must be modified so that it is expressed with respect to the docking frame by means of Equation (2.20), which is here copied for sake of clarity

$$I_d = A_{db}\left(I_b + m\left[||\boldsymbol{r}_b^d||^2\mathbb{1}_3 - \boldsymbol{r}_b^d\boldsymbol{r}_b^{d^T}\right]\right)A_{db}^T$$

The DCM matrix which maps vector from the body frame to the docking frame, A_{db}, corresponds the identity matrix (2.4). As a consequence, being I_b diagonal, I_d will remain diagonal.

5.2.3 Actuator Parameters

The actuators models have been defined in Section 2.5. Here, the parameters used in the simlulator are presented.

Reaction Control System

Recall the assumption that two *NanoProp 6DOF* cold gas RCSs from GOMspace [24], scaled to 3U CubeSat form factors, are available (see Section 2.5.1 for details of the model). Each of the two RCS provides 6 thrusters with the capability of pruducing a 1 mN thrust each.

The parameters used to model the RCS are listed in Table 5.2.

Table 5.2: RCS parameters [24]

Thrust	4 mN per axis
Minimum time ON	25 ms
Specific impulse	$I_{sp} = 50$ s

The PWPF modulator used in the RCS model has been tuned and the parameters used in the simulations are listed in Table 5.3.

Table 5.3: PWPF modulator parameters

Filter's gain	$k_m = 10$
Filter's time constant	$\tau_m = 2$
ON threshold	$U_{ON} = 0.002$ N
OFF threshold	$U_{OFF} = 0.001$ N
Output amplitude	$U_m = 0.004$ N

Reaction Wheels

The Reaction Wheel model used for this research has been described in Section 2.5.2, assuming a Clyde Space RW210 reaction wheel per body axis can be installed in the chaser [28]. Being a simplified model, in which no reaction wheel unloading system is implemented and no limitation of the angular speed of the wheels and of the maximum torque obtainable are considered, no tuning had to be done.

The only parameter considered is the maximum torque, but it has been used in the LQR tuning process (see Section 5.2.4). According to the data-sheet provided by Clyde Space, the maximum torque per wheel is

$$T_{max} = 0.1 \text{ mN m} \tag{5.4}$$

5.2.4 Estimation and Control

The parameters and the solutions adopted in the simulator to implement the state estimator and the controller are here described.

Extended Kalman Filter

The Extended Kalman Filter has been tuned until its convergence performance is optimal, given the nominal standard deviation of the Pixel Errror defined (3.48). The optimality is considered as the best stabilization time during the handover in $SK0$.

At the end of Section 4.1.1 it has been described how a solution that reduces the computations involved in the continuous-discrete EKF involves the use of a pre-linearized state transition model rather then the non-linear one. Such solution has been adopted in this research. As a consequence, given a set of initial conditions and an initial covariance, both relative attitude and relative position are only determined by the respective dynamics [6]. It follows that the process covariance associated to angles and position must be set to zero.

The parameter used are listed in Table 5.4.

Table 5.4: EKF parameters

Initial condition	$\bar{\boldsymbol{\alpha}}^{d_c d_t} = \begin{bmatrix}0^\circ\\0^\circ\\0^\circ\end{bmatrix} \quad \bar{\boldsymbol{\omega}}^{d_c d_t}_{d_t} = \begin{bmatrix}0^\circ\\0^\circ\\0^\circ\end{bmatrix}\,\mathrm{s}^{-1}$
	$\bar{\boldsymbol{s}}^{d_c d_t}_{d_t} = \begin{bmatrix}-5\\0\\0\end{bmatrix}\,\mathrm{m} \quad \bar{\dot{\boldsymbol{s}}}^{d_c d_t}_{d_t} = \begin{bmatrix}0\\0\\0\end{bmatrix}\,\mathrm{m\,s^{-1}}$
Initial covariance	$P_0 = 10^{-3} \cdot \mathbb{1}_{12}$
Process covariance	$Q = \begin{bmatrix}\mathbf{0}_3 & 50 & 50 & 50 & \mathbf{0}_3 & 50 & 50 & 50\end{bmatrix} \cdot \mathbb{1}_{12}$
Noise covariance	$R_k = 5 \cdot 10^{-1}\mathbb{1}_{10}$

Linear Quadratic Regulator

The LQR is actually composed of eight different blocks which control translations and station keeping operations (see Figure 5.2):

- a first LQR must stabilize the chaser in $SK0$, and will be referred to as *LQR0*;
- the next two will operate the chaser to the FRPs ($SK1$ and $SK2$), will be tuned in the same way and will be jointly referred to as *LQR1*,
- the next four will operate the chaser to the CRPs ($SK3$, $SK4$, $SK5$ and $SK6$), will be tuned in the same way and will be jointly referred to as *LQR2*,
- the final controller stabilizes the chaser in $SK6$ with a stronger action, so to meet the accuracy required before Soft Docking, and will be referred to as *LQR3*.

A logical control unit handles the navigation planning task, regulating the operation of the eight controllers. Each controller is tuned so that the overshoot at each SK point never exceeds 10% of the range.

Matrices Q and R are diagonal. The first describes the weight given to the control of the states of the system. The second one describes the weight given to the optimization of the control action. The tuning process has been done with the following approach:

- set the R matrix by means of two 3-parameter vectors, function of the maximum torque and force:

$$r_T = \frac{1}{T_{max}^2} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.5a}$$

$$r_F = \frac{1}{F_{max}^2} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.5b}$$

$$R = \text{diag}\left(\begin{bmatrix} r_T & r_F \end{bmatrix}\right) \tag{5.5c}$$

  where function *diag()* describes a diagonal matrix having in its principal diagonal the vector received as argument,

- initialize four 3-parameter vectors, describing the weight given to each of the state variables, to be all equal to 1:

$$q_\alpha = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.6a}$$

$$q_\omega = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.6b}$$

$$q_s = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.6c}$$

$$q_{\dot{s}} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{5.6d}$$

$$R = \text{diag}\left(\begin{bmatrix} q_\alpha & q_\omega & q_s & q_{\dot{s}} \end{bmatrix}\right) \tag{5.6e}$$

- regulate the four q vectors until the desired performance is achieved.

The R matrix is never changed and is the same for all four regulators: *LQR0*, *LQR1*, *LQR2* and *LQR3*. The parameters used for the controllers are listed in Table 5.5.

Table 5.5: LQR parameters

Inputs weight - all regulators	$r_T = \frac{1}{0.1^2 \text{mN m}} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$
	$r_F = \frac{1}{4^2 \text{mN}} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$
State weight - *LQR0*	$q_\alpha = \begin{bmatrix} 50 & 50 & 50 \end{bmatrix}$
	$q_\omega = \begin{bmatrix} 50 & 50 & 50 \end{bmatrix}$
	$q_s = \begin{bmatrix} 100 & 100 & 100 \end{bmatrix}$
	$q_{\dot{s}} = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
State weight - *LQR1*	$q_\alpha = \begin{bmatrix} 100 & 100 & 100 \end{bmatrix}$
	$q_\omega = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
	$q_s = \begin{bmatrix} 50 & 100 & 100 \end{bmatrix}$
	$q_{\dot{s}} = \begin{bmatrix} 1000 & 100 & 100 \end{bmatrix}$
State weight - *LQR2*	$q_\alpha = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
	$q_\omega = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
	$q_s = \begin{bmatrix} 300 & 500 & 500 \end{bmatrix}$
	$q_{\dot{s}} = \begin{bmatrix} 7770 & 10 & 10 \end{bmatrix}$
State weight - *LQR3*	$q_\alpha = \begin{bmatrix} 50000 & 50000 & 50000 \end{bmatrix}$
	$q_\omega = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
	$q_s = \begin{bmatrix} 1000 & 1000 & 1000 \end{bmatrix}$
	$q_{\dot{s}} = \begin{bmatrix} 10 & 10 & 10 \end{bmatrix}$

The LQR gain K_r is then generated following the mathematical model of the controller described in Section 4.2.

5.2.5 Handover Parameters

The handover residual states and velocities are generated randomly assuming they can be described by Gaussian distributions to be summed to the nominal state defined in Section 4.3

$$\Delta\boldsymbol{\alpha} \sim \mathcal{N}(0, H_\alpha) \tag{5.7a}$$
$$\Delta\boldsymbol{\omega} \sim \mathcal{N}(0, H_\omega) \tag{5.7b}$$
$$\Delta\boldsymbol{s} \sim \mathcal{N}(0, H_s) \tag{5.7c}$$
$$\Delta\dot{\boldsymbol{s}} \sim \mathcal{N}(0, H_{\dot{s}}) \tag{5.7d}$$

The initial conditions are thus generated as follows

$$\boldsymbol{\alpha}_0 = \bar{\boldsymbol{\alpha}}^{d_c d_t} + \Delta\boldsymbol{\alpha} \tag{5.8a}$$
$$\boldsymbol{\omega}_0 = \bar{\boldsymbol{\omega}}^{d_c d_t}_{d_t} + \Delta\boldsymbol{\omega} \tag{5.8b}$$
$$\boldsymbol{s}_0 = \bar{\boldsymbol{s}}^{d_c d_t}_{d_t} + \Delta\boldsymbol{s} \tag{5.8c}$$
$$\dot{\boldsymbol{s}}_0 = \bar{\dot{\boldsymbol{s}}}^{d_c d_t}_{d_t} + \Delta\dot{\boldsymbol{s}} \tag{5.8d}$$

The standard deviations defined are four, one for each set of state variables:

$$\sigma_{\boldsymbol{\alpha}} = 2° \tag{5.9a}$$
$$\sigma_{\boldsymbol{\omega}} = 0.5° \text{ s}^{-1} \tag{5.9b}$$
$$\sigma_{\boldsymbol{s}} = 0.1 \text{ m} \tag{5.9c}$$
$$\sigma_{\dot{\boldsymbol{s}}} = 0.01 \text{ m s}^{-1} \tag{5.9d}$$

A first set of randomly generated initial conditions has been created and saved, so to use it for all three patterns to test the impact of the Pixel Error in the stabilization around $SK0$ at the moment of handover. This set will be referred to as *IC1*, and is listed in Table 5.6.

Table 5.6: Initial Conditions *IC1*

Initial Attitude	$\boldsymbol{\alpha}_0 = \begin{bmatrix} 0.880° & 0.203° & 5.575° \end{bmatrix}^T$
Initial Angular Velocity	$\boldsymbol{\omega}_0 = \begin{bmatrix} -0.583° & -0.9271° & -0.5703° \end{bmatrix}^T \text{ s}^{-1}$
Initial Position	$\boldsymbol{s}_0 = \begin{bmatrix} -5.109 & -0.043 & -0.017 \end{bmatrix}^T \text{ m}$
Initial Linear Velocity	$\dot{\boldsymbol{s}}_0 = \begin{bmatrix} -0.001 & 0.003 & 0.002 \end{bmatrix}^T \text{ m s}^{-1}$

5.2.6 Simulink Model

The Simulink blocks shown in Figure 5.1 will now be briefly described.

- **P2P Dynamics**: this is the plant of the system, and implements the nonlinear version of the P2P coupled dynamics described in Section 2.4. It receives as input the control actions generated by the LQG system, summed to the disturbance forces and torques described in Section 2.6, and the variable mass. Its output is the true relative state, which is required to generate the images and to correctly model the disturbances.
- **Image Generation**: this block implements the algorithm described in Table 3.2. It receives as input the true relative state and generates as output the measured pixels position vector defined in (3.26).
- **EKF**: this is the continuous-discrete Extended Kalman Filter, realized with a Simulink built-in block, which receives as input the measurement vector defined in (3.26) and the control inputs generated by the LQR. It implements the algorithm described in Table 4.1, but with the pre-linearized version of the dynamic model. It is tuned with the parameters described in Table 5.4. The output of this block is the estimated P2P state.
- **LQR**: this is the controller of the system, and it features the control unit and the eight controllers described in Section 5.2.4. It receives as input the state estimated by the EKF, and produces the control forces and torques expressed in the chaser's docking frame, as required by the P2P dynamics.
- **Disturbances**: in this block, the three disturbances described in Section 2.6 are implemented. Its input is the real relative state, while it outputs the disturbance forces and torques to be summed to the control action and then passed to the plant.
- **Variabl Mass**: this block implements the variable mass model described in Appendix A. It receives as input the control force generated by the RCS. Its output is the current mass of the chaser, which is fed to the plant.

5.3 Simulations

5.3.1 Final Approach

For each pattern, a first simulation with the same initial conditions *IC1* (5.6) and the same nominal standard deviation for the Pixel Error (3.48) is performed. This simulation can be used to analyze the Root Mean Square (RMS) error of the state estimation performed by the EKF, the overall time required for the Final Approach and the position and pointing accuracy achievable at docking. Recall that, in general, an estimation error, is defined as $\boldsymbol{\varepsilon} = \boldsymbol{x} - \hat{\boldsymbol{x}}$. The final accuracy is computed as the true average position and true average relative attitude for the last 60 s, namely half of the time-span in which the chaser is kept stable in $SK6$.

A list which organizes figures and tables by pattern is here presented.

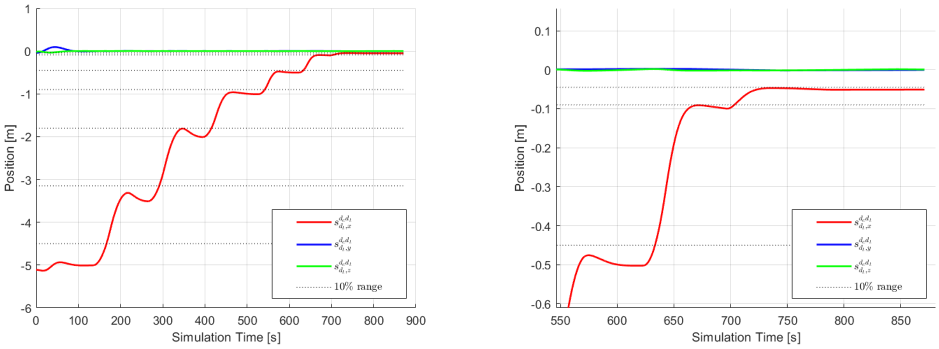
- **Pattern 1**:
  - true final approach relative position: Figure 5.4;
  - estimation error for the four vectors composing the P2P state: Figure 5.5,
  - pattern, with both true and measured positions, seen from $SK3$, $SK4$, $SK5$ and $SK6$: Figure 5.6,
  - enlargement of the pattern seen from $SK0$ and $SK3$ to show the spacing in pixels between LEDs: Figure 5.7,
  - RMS estimation errors, simulation time and final accuracy for the full mission: Table 5.7.
- **Pattern 2**:
  - true final approach relative position: Figure 5.8,
  - estimation error for the four vectors composing the P2P state: Figure 5.9,
  - pattern, with both true and measured positions, seen from $SK3$, $SK4$, $SK5$ and $SK6$: Figure 5.10,
  - enlargement of the pattern seen from $SK0$ and $SK3$ to show the spacing in pixels between LEDs: Figure 5.11,
  - RMS estimation errors, simulation time and final accuracy for the full mission: Table 5.8.
- **Pattern 3**:
  - true final approach relative position: Figure 5.12,
  - estimation error for the four vectors composing the P2P state: Figure 5.13,
  - pattern, with both true and measured positions, seen from $SK3$, $SK4$, $SK5$ and $SK6$: Figure 5.14,
  - enlargement of the pattern seen from $SK0$ and $SK3$ to show the spacing in pixels between LEDs: Figure 5.15,
  - RMS estimation errors, simulation time and final accuracy for the full mission: Table 5.9.

Analyzing the complete Final Approach manoeuvre it seems that the three patterns have comparable performance. They can all yield a position and pointing accuracy definitely better then the requirements mentioned at the beginning of this chapter: the pointing accuracy is better then 0.1°, while the lateral displacement is better then 1.2 mm. Furthermore, if the 60 s window used for the computation of the average relative attitude and position is reduced, the accuracy improves even more.

The simulation time is very similar for each of the patterns. After the first stabilization part, the mission requires the same time for each of the patterns. During the stabilization, as it will be explained in the following sections, Pattern 2 is the one which requires slighlty longer time.

The RMS estimation errors are considered for the complete FA. During the initialization, however, the estimation error is larger then in the rest of the FA, as it can be seen in Figures 5.5, 5.9, and 5.13. As a consequence, if the first 100 s are excluded from the computation of the RMS errors, their value decreases by at least one order of magnitude for each pattern.

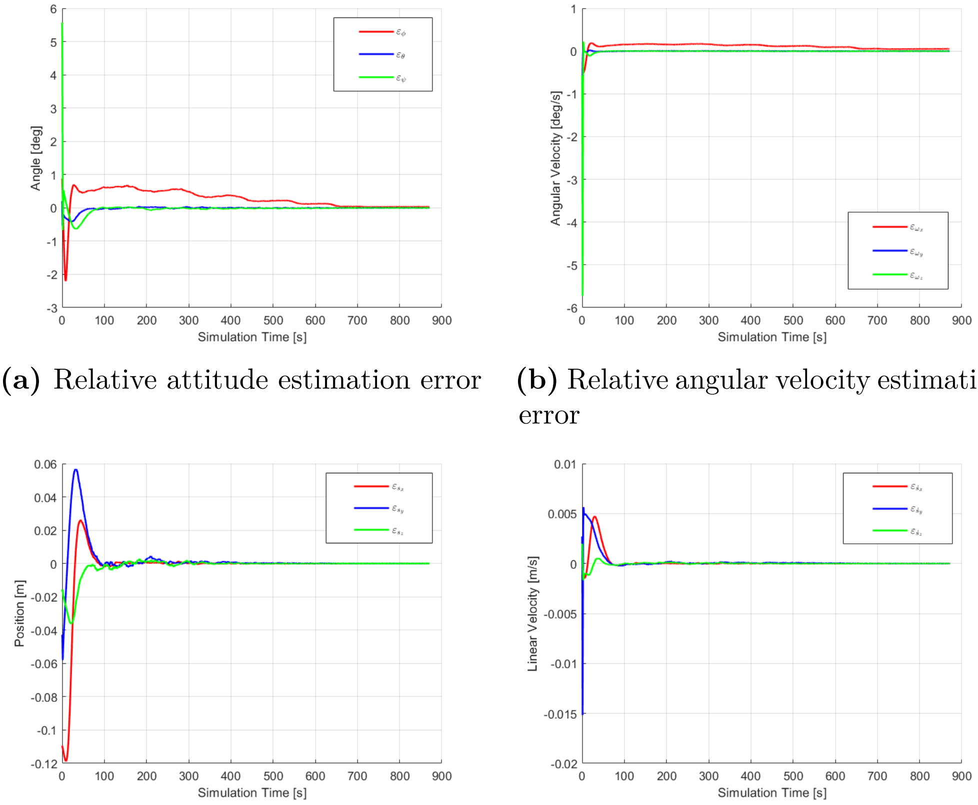
It is worth mentioning that LED number 2 of Pattern 2, as expected (see Section 3.2.2), is seen close to the edge of the image plane (Figure 5.10d). In case of a combination of lateral displacement and yaw angle rotation, resulting in a substantial move to the right of the observed pattern, the LED might escape the FoV of the camera, implying a reduction of the estimation performance. Inserting the camera in a cavity located on the docking side of the chaser, as it is done in the other two solutions, would limit this problem. However, this would also imply worst stabilization performance (see Section 5.3.2), making the choice of Pattern 2 not as appealing as the others.



(a) Relative position of the full Final Approach mission

(b) Enlargement of the final 50 cm

Figure 5.4: Pattern 1 - Final Approach



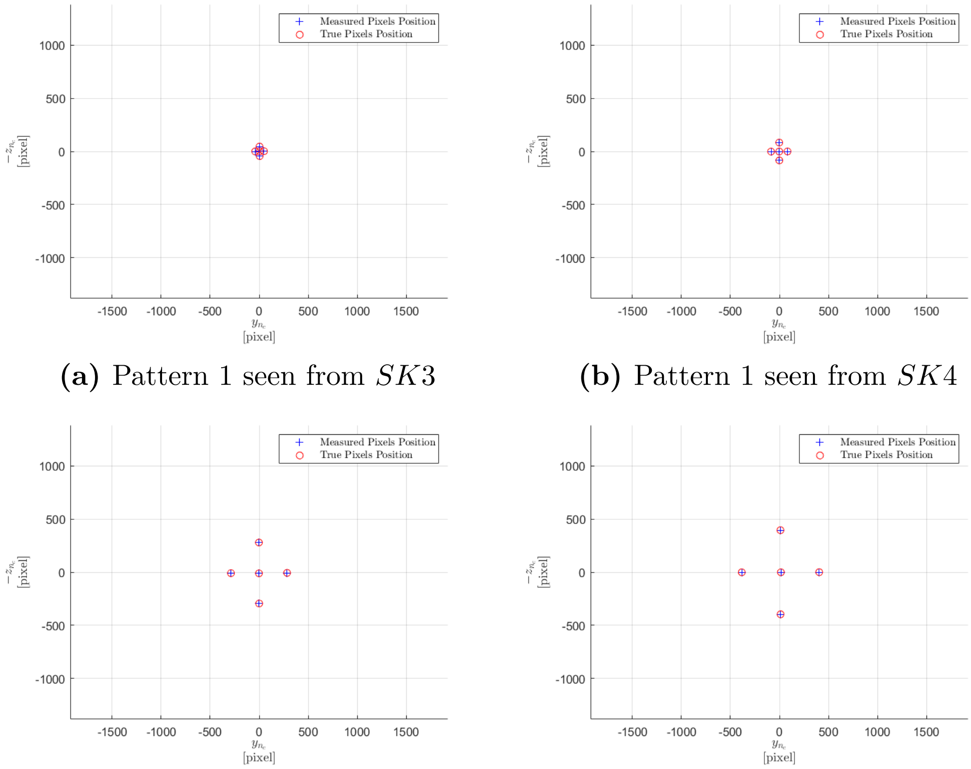
(a) Relative attitude estimation error

(b) Relative angular velocity estimation error

(c) Relative position estimation error

(d) Relative linear velocity estimation error

Figure 5.5: Pattern 1 - Estimation Error



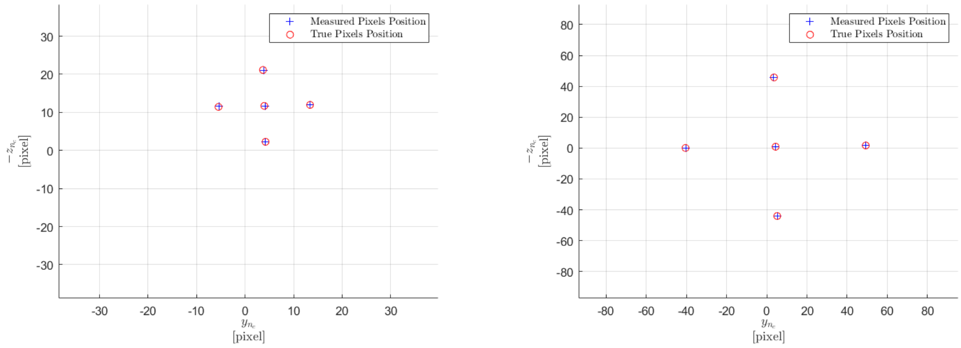
(a) Pattern 1 seen from $SK3$

(b) Pattern 1 seen from $SK4$

(c) Pattern 1 seen from $SK5$

(d) Pattern 1 seen from $SK6$

Figure 5.6: Pattern 1 - LEDs positions on the image plane from the last four station keeping points



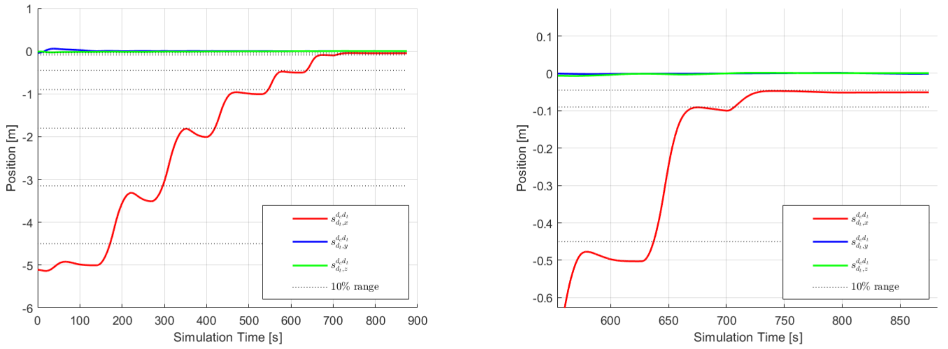
(a) Enlargement of Pattern 1 seen from $SK0$

(b) Enlargement of Pattern 1 seen from $SK3$

Figure 5.7: Pattern 1 - spacing in pixels between LEDs

Table 5.7: Pattern 1 - Final Approach performance

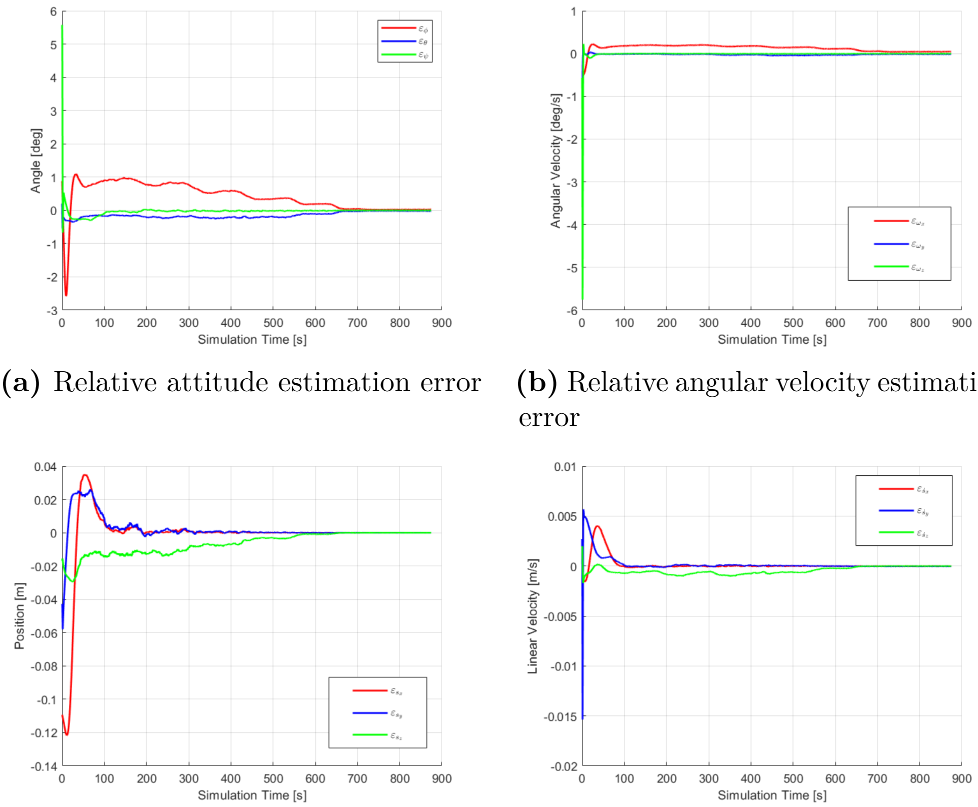
Attitude RMS error	$\boldsymbol{\varepsilon}_{RMS,\alpha} = \begin{bmatrix} 0.4099° \\ 0.0749° \\ 0.1892° \end{bmatrix}$
Angular velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\omega} = \begin{bmatrix} 0.1286° \\ 0.0317° \\ 0.1837° \end{bmatrix} \mathrm{s}^{-1}$
Position RMS error	$\boldsymbol{\varepsilon}_{RMS,s} = \begin{bmatrix} 0.0176 \\ 0.0106 \\ 0.0062 \end{bmatrix} \mathrm{m}$
Linear velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\dot{s}} = 10^{-3} \cdot \begin{bmatrix} 0.7 \\ 0.9 \\ \cdot 0.2 \end{bmatrix} \mathrm{m\,s^{-1}}$
Simulation time	$T_{sim} = 871$ s
Attitude final accuracy	$\boldsymbol{\alpha}_{accuracy} = \begin{bmatrix} 0.0810° \\ -0.0134° \\ -0.0183° \end{bmatrix}$
Position final accuracy	$\boldsymbol{s}_{accuracy} = \begin{bmatrix} -0.0512 \\ -0.0011 \\ 0.0002 \end{bmatrix} \mathrm{m}$



(a) Relative position of the full Final Approach mission

(b) Enlargement of the final 50 cm

Figure 5.8: Pattern 2 - Final Approach



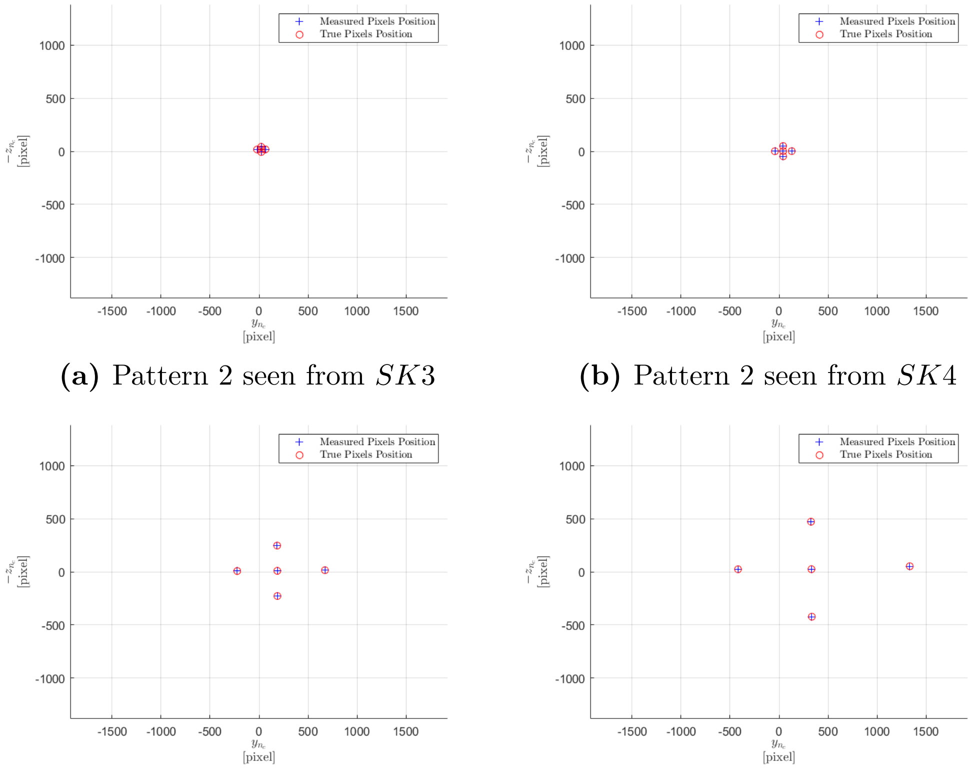
(a) Relative attitude estimation error

(b) Relative angular velocity estimation error

(c) Relative position estimation error

(d) Relative linear velocity estimation error

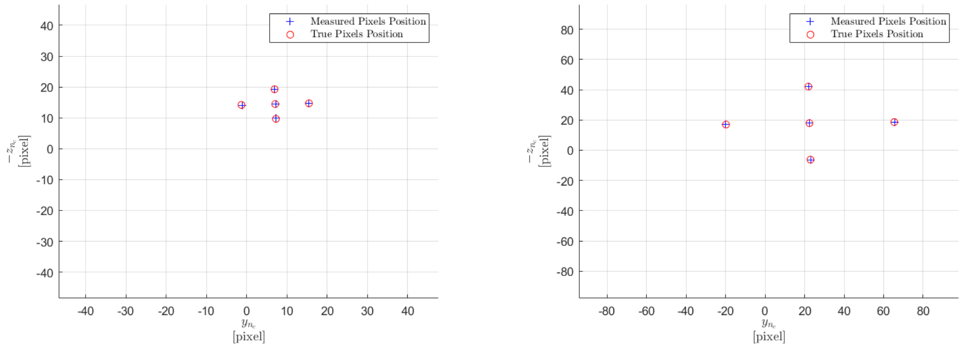
Figure 5.9: Pattern 2 - Estimation Error



(a) Pattern 2 seen from $SK3$ **(b)** Pattern 2 seen from $SK4$

(c) Pattern 2 seen from $SK5$ **(d)** Pattern 2 seen from $SK6$

Figure 5.10: Pattern 2 - LEDs positions on the image plane from the last four station keeping points

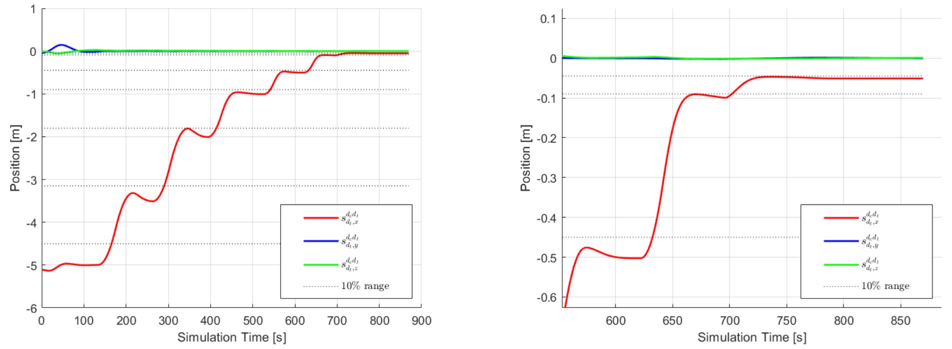


(a) Enlargement of Pattern 2 seen from $SK0$ **(b)** Enlargement of Pattern 2 seen from $SK3$

Figure 5.11: Pattern 2 - spacing in pixels between LEDs

Table 5.8: Pattern 2 - Final Approach performance

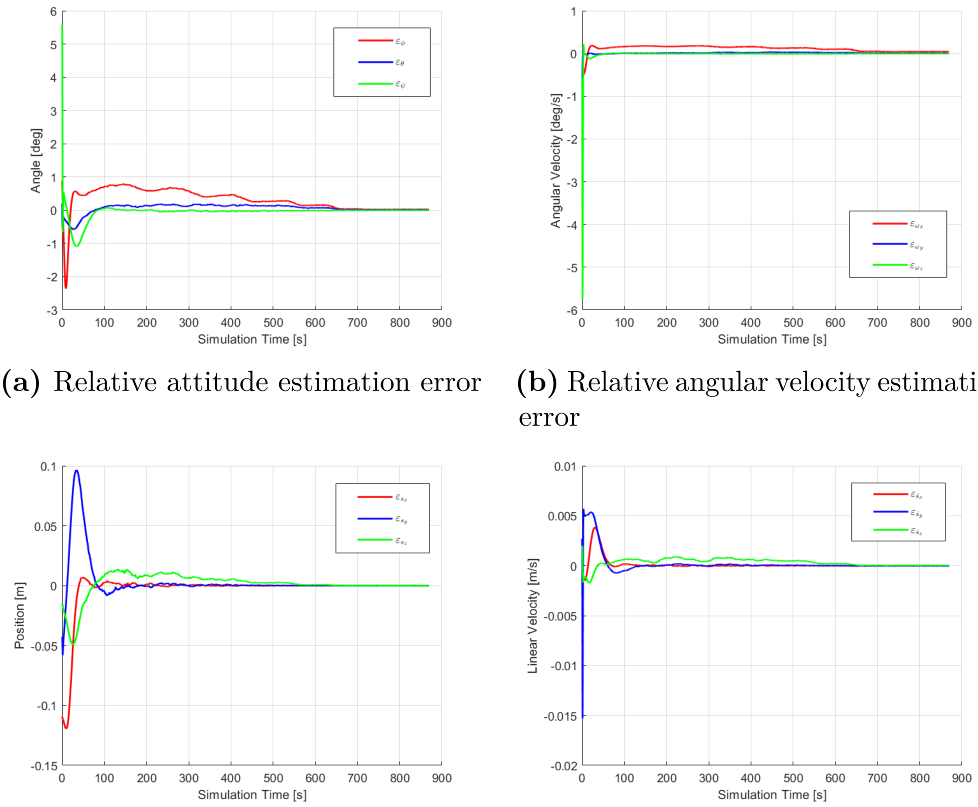
Attitude RMS error	$\boldsymbol{\varepsilon}_{RMS,\alpha} = \begin{bmatrix} 0.6021^\circ \\ 0.1691^\circ \\ 0.1706^\circ \end{bmatrix}$
Angular velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\omega} = \begin{bmatrix} 0.1519^\circ \\ 0.0400^\circ \\ 0.1836^\circ \end{bmatrix} \mathrm{s}^{-1}$
Position RMS error	$\boldsymbol{\varepsilon}_{RMS,s} = \begin{bmatrix} 0.0200 \\ 0.0077 \\ 0.0093 \end{bmatrix} \mathrm{m}$
Linear velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\dot{s}} = 10^{-3} \cdot \begin{bmatrix} 0.7 \\ 0.9 \\ 0.6 \end{bmatrix} \mathrm{m\,s}^{-1}$
Simulation time	$T_{sim} = 874$ s
Attitude final accuracy	$\boldsymbol{\alpha}_{accuracy} = \begin{bmatrix} 0.0743^\circ \\ -0.0729^\circ \\ -0.0186^\circ \end{bmatrix}$
Position final accuracy	$\boldsymbol{s}_{accuracy} = \begin{bmatrix} -0.0512 \\ 0.0005 \\ 0.0006 \end{bmatrix} \mathrm{m}$



(a) Relative position of the full Final Approach mission

(b) Enlargement of the final 50 cm

Figure 5.12: Pattern 3 - Final Approach



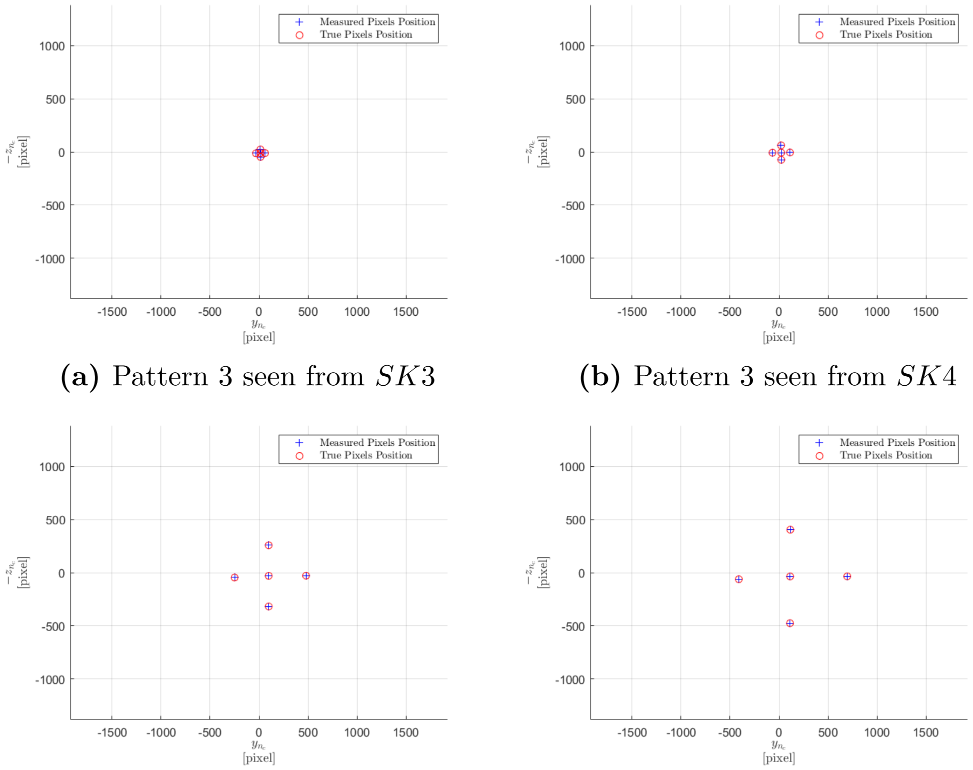
(a) Relative attitude estimation error

(b) Relative angular velocity estimation error

(c) Relative position estimation error

(d) Relative linear velocity estimation error

Figure 5.13: Pattern 3 - Estimation Error



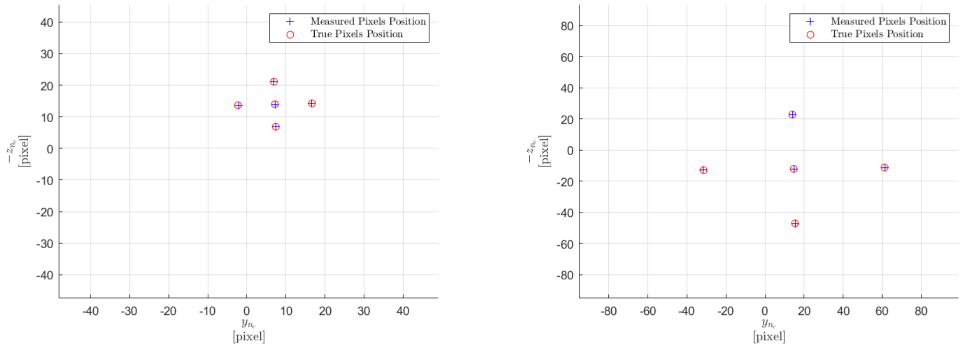
(a) Pattern 3 seen from $SK3$

(b) Pattern 3 seen from $SK4$

(c) Pattern 3 seen from $SK5$

(d) Pattern 3 seen from $SK6$

Figure 5.14: Pattern 3 - LEDs positions on the image plane from the last four station keeping points



(a) Enlargement of Pattern 3 seen from $SK0$

(b) Enlargement of Pattern 3 seen from $SK3$

Figure 5.15: Pattern 3 - spacing in pixels between LEDs

Table 5.9: Pattern 3 - Final Approach performance

Attitude RMS error	$\boldsymbol{\varepsilon}_{RMS,\alpha} = \begin{bmatrix} 0.4729° \\ 0.1496° \\ 0.2452° \end{bmatrix}$
Angular velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\omega} = \begin{bmatrix} 0.1370° \\ 0.0355° \\ 0.1841° \end{bmatrix} \mathrm{s}^{-1}$
Position RMS error	$\boldsymbol{\varepsilon}_{RMS,s} = \begin{bmatrix} 0.0182 \\ 0.0173 \\ 0.0102 \end{bmatrix} \mathrm{m}$
Linear velocity RMS error	$\boldsymbol{\varepsilon}_{RMS,\dot{s}} = 10^{-3} \cdot \begin{bmatrix} 0.6 \\ 1.1 \\ 0.5 \end{bmatrix} \mathrm{m\,s}^{-1}$
Simulation time	$T_{sim} = 870$ s
Attitude final accuracy	$\boldsymbol{\alpha}_{accuracy} = \begin{bmatrix} 0.0750° \\ 0.027° \\ -0.0141° \end{bmatrix}$
Position final accuracy	$\boldsymbol{s}_{accuracy} = \begin{bmatrix} -0.0512 \\ 0.0005 \\ -0.0004 \end{bmatrix} \mathrm{m}$

5.3.2 Handover Performance

The handover performance is evaluated by means of a Monte Carlo analysis obtained running 100 simulations for each pattern. For this task, the Sensitivity Analyzer application available in Simulink has been used. For every simulation, a different initial condition defined according to (5.8) is generated. The evaluation parameter is the time required by the system to stabilize the chaser around $SK0$ (see Section 5.1). The standard deviation of the Pixel Error is kept fixed to its nominal value for all simulations (3.48).

The probabilities of a certain stabilization time, given the randomly generated initial conditions, are shown in Figures 5.16, 5.17 and 5.18 for Pattern 1, Pattern 2 and Pattern 3 respectively. The performance at handover of the three patterns can be evaluated computing the mean and the standard deviation for each of the distributions. The results obtained are listed in Table 5.10.

This simulation shows that the sensitivity of the VBN algorithm to the initial conditions is comparable for the three patterns. Pattern 1 performs slightly better, while Pattern 2 has the worst performance, but this difference ($\Delta\mu_{2-1} = 14$ s) is small ($\approx 1.6\%$) compared to the time required for the full mission, which is around 870 s.

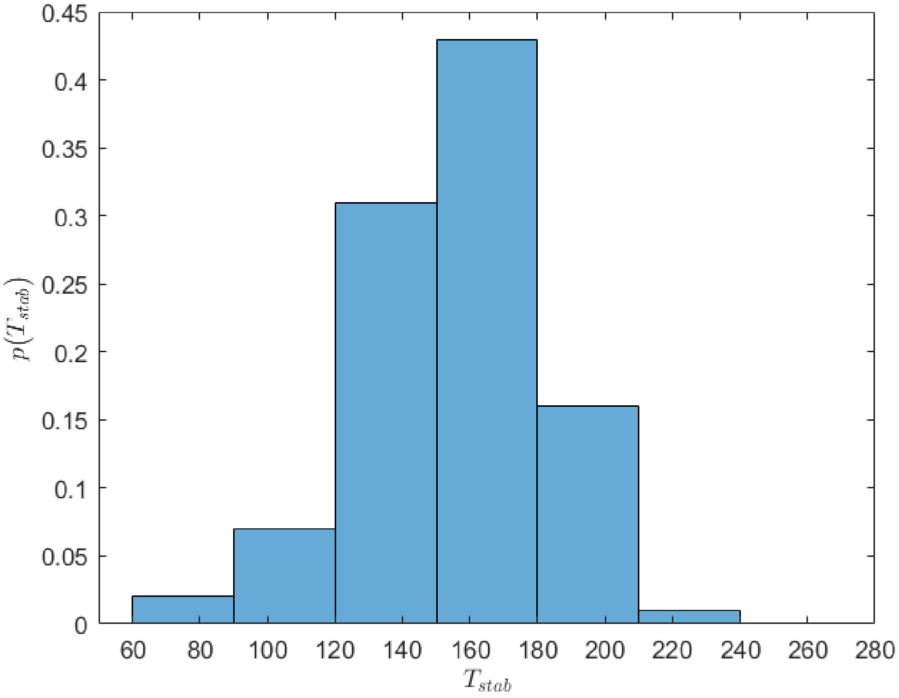


Figure 5.16: Pattern 1 - Probability distribution of the stabilization time for 100 sets of initial conditions, defined as in (5.8), (5.9)

Table 5.10: Mean and standard deviation of the Stabilization Time probability distributions

Pattern 1	$\mu_{H1} = 154$ s
	$\sigma_{H1} = 28.1$ s
Pattern 2	$\mu_{H2} = 168$ s
	$\sigma_{H2} = 37.0$ s
Pattern 3	$\mu_{H3} = 162$ s
	$\sigma_{H3} = 30.6$ s

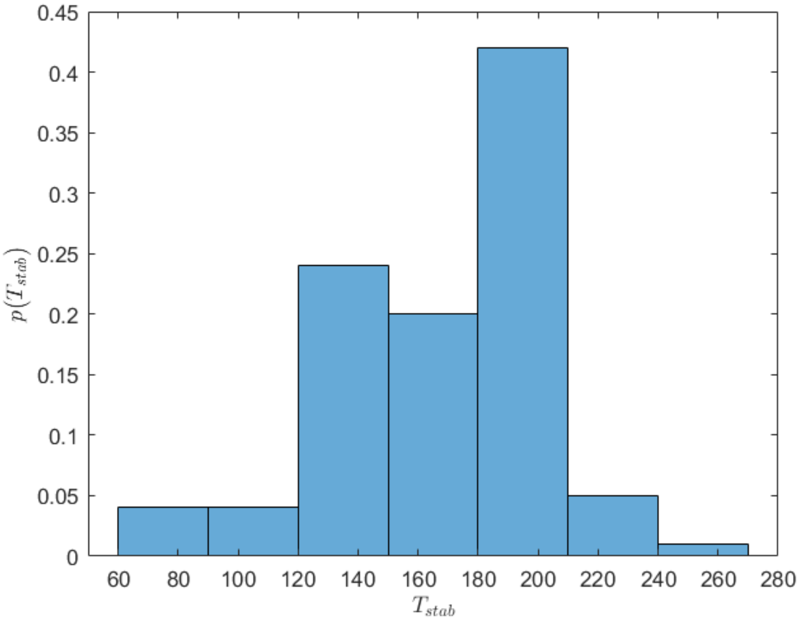


Figure 5.17: Pattern 2 - Probability distribution of the stabilization time for 100 sets of initial conditions, defined as in(5.8), (5.9)

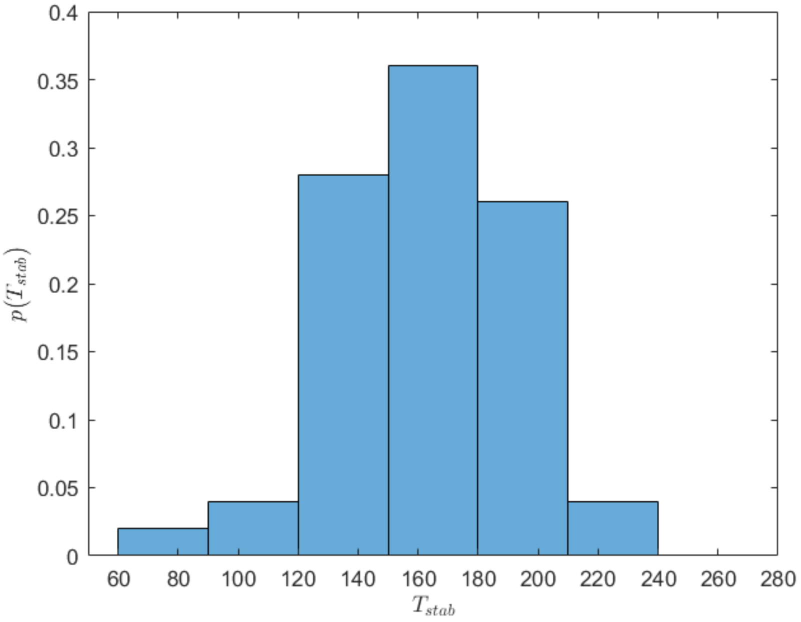


Figure 5.18: Pattern 3 - Probability distribution of the stabilization time for 100 sets of initial conditions, defined as in (5.8), (5.9)

5.3.3 Pixels Detection Error

The Pixel Error impact on the stabilization time is evaluated by means of a second Monte Carlo analysis, again composed of 100 simulations for each pattern. For every simulation, a different standard deviation of the Pixel Error is generated. This random values are uniformly distributed in the range [0.1,1]pixel. The evaluation parameter is again the time required for the stabilization around $SK0$ (see Section 5.1). The initial conditions used are the ones defined in Table 5.6. The stabilization time at the variation of σ_{pix} for the three patterns is depicted in figures 5.19, 5.20 and 5.21 for Pattern 1, Pattern 2 and Pattern 3 respectively.

An interesting behaviour in the stabilization time trend can be observed: such parameter increases in a step-wise fashion for certain values of σ_{pix}, values which are different for the three patterns.

- **Pattern 1**: the value of σ_{pix} that implies a consistent step in T_{stab} is very close to 1 pixel. After the step, the stabilization time becomes approximately 210 s.
- **Pattern 2**: shows three steps inside the chosen range of σ_{pix}:
  1. the first step, corresponding to $\sigma_{pix} \approx 0.55$ pixel, makes the stabilization time jump to around 200 s,
  2. the second step, corresponding to $\sigma_{pix} \approx 0.78$ pixel, makes the stabilization time jump to around 350 s,
  3. the third step, corresponding to $\sigma_{pix} \approx 0.96$ pixel, makes the stabilization time jump to around 375 s.
- **Pattern 3**: the step in T_{stab} can be observed for a value of $\sigma_{pix} \approx 0.84$ pixel. After the step, the stabilization time becomes approximately 210 s.

Recall that Pattern 2 is the one in which the LEDs composing the vertical axis of the cross-shape are less spaced between each other. As it can be seen in Section 3.2.2, there is only 1 cm between LED number 1, the top one, and LED number 5, the central one. The same holds for LED number 3, the bottom one. In Pattern 1 instead, the corresponding distance is 2 cm, while for Pattern 3 is 1.5 cm. It can be noticed that the trend in the spacing between the three Patterns corresponds to the trend in the performance just described in the bullet list.

In particular, Pattern 2 performs worst then the other two, confirming that the spacing between the LEDs is an important design parameter for a VBN algorithm of the sort described in this research. It is in fact simple to understand that, when the LEDs are seen just a few pixels apart (Figure 5.11a), an error in the detection of the centre of the blob has an higher impact in the state estimation process and thus in the stabilization of the satellite.

In the case of Pattern 1 and 3, only one major step can be observed, and occurs at high values of σ_{pix}. The stabilization time never goes above 210 s, and remains below 150 s for most of the values of σ_{pix}. For Pattern 2, instead, the first jump of T_{stab} is observed at relatively lower values of σ_{pix}, and it is followed by two further jumps. It reaches approximately 200 s when σ_{pix} is just above 0.5, and can become as high as 380 s when σ_{pix} approaches 1.

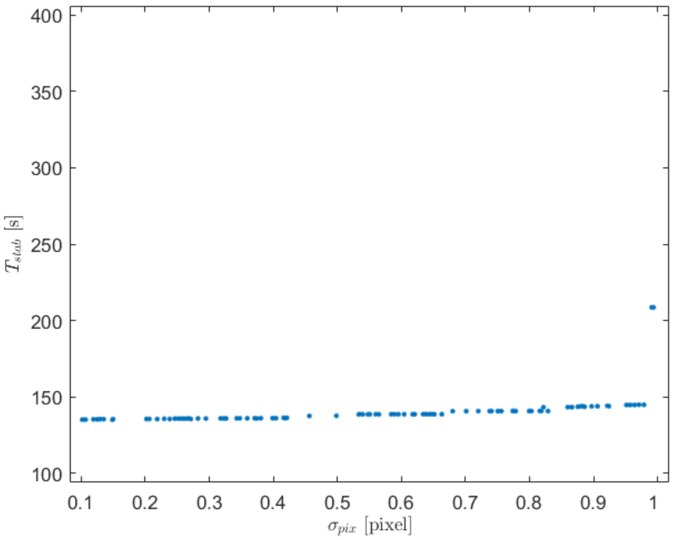


Figure 5.19: Pattern 1 - Stabilization time function of the standard deviation of the Pixel Error

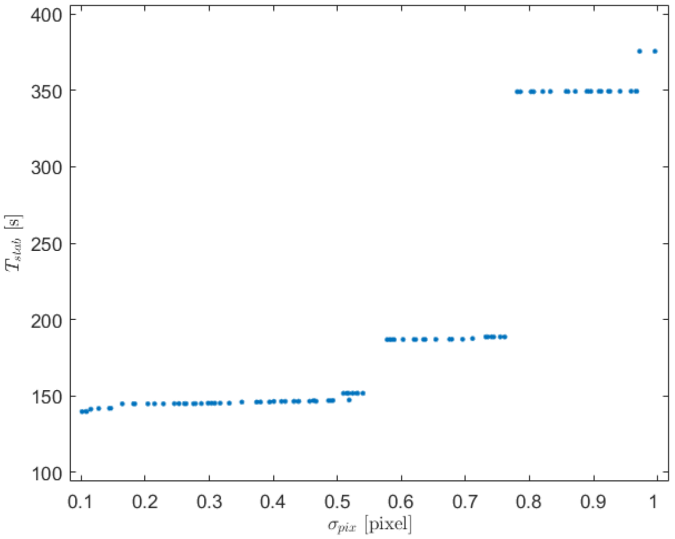


Figure 5.20: Pattern 2 - Stabilization time function of the standard deviation of the Pixel Error

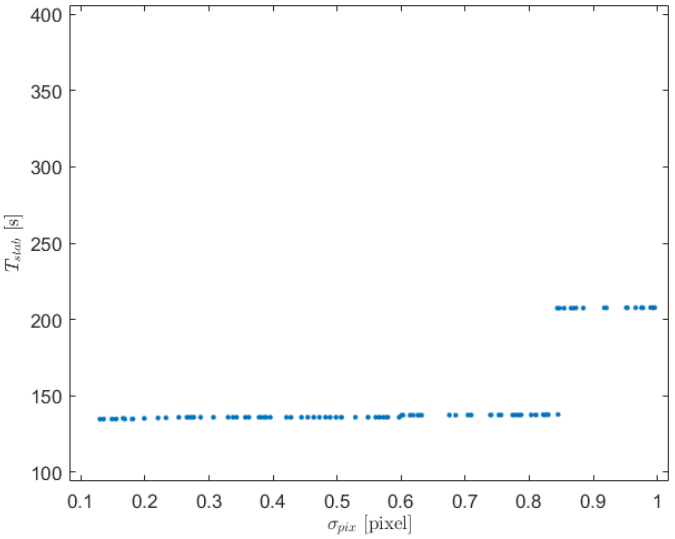


Figure 5.21: Pattern 3 - Stabilization time function of the standard deviation of the Pixel Error

Chapter 6

Conclusions and Future Works

The goal of this research has been to test a Vision-Based Navigation algorithm for the Final Approach (FA) phase of a 3U CubeSat Rendezvous and Docking mission in Low Earth Orbit (LEO). Three different LEDs patterns have been studied to asses the sensitivity of the algorithm with respect to the handover accuracy and to the pixel detection error. Pattern 1, taken from the existing literature, requires hardware integration with the docking mechanism. Pattern 2 and 3, instead, have been designed by the candidate to allow the use of a generic docking mechanism which must only fulfill an area requirement.

A Simulink®-based simulator describing the Port-To-Port (P2P) orbital coupled dynamics has been designed. A cold gas Reaction Control System is assumed to be used as translation actuator. A set of Reaction Wheels are assumed to be the attitude actuation system. The chaser's model accounts for the mass variations due to the fuel consumption. The true state of the chaser is used to simulate the images taken from the camera sensor. The fake images are needed to obtain the LEDs pixels positions measurements in the image plane. The true pixels positions are corrupted with a Gaussian white noise to simulate the effects of the image processing algorithm, which is required in a real application. A continuous-discrete Extended Kalman Filter uses this measurements to simultaneously perform state estimation and filtering. It uses a linearized version of the P2P coupled dynamics and the nonlinear version of the measurement equations. Eight LQR controllers are used to generate the control inputs during different phases of the FA. A logical control unit is used to coordinate the operations of the eight internal controllers. The true state of the system is also needed to simulate the three main disturbances affecting LEO missions: aerodynamic drag, magnetic torque and gravity gradient.

The first simulation tests the full FA for the three different patterns. The same initial conditions and the same LEDs detection error are used for each pattern. The simulation shows that the performance are comparable and that no pattern, under nominal conditions, seems to stand out. The mission time and the Root Mean Square estimation errors are practically the same for all the solutions. The overshoot requirement is fulfilled for all the intermediate steps of the translation, and for all the three patterns. Furthermore, the true state accuracy achieved at the end of the translation always fulfills with a good margin the docking precision requirements. However, the pattern design for the second proposed solution implies the right-most LED to be close to the edge of the Field of View of the camera during the final centimeters of the translation. If, as a consequence of a combination of lateral displacement and yaw angle rotation, the LED is not seen, the estimation accuracy worsens, consistently increasing the risk of mission failure. Such limiting condition is unlikely because the estimation and control accuracy in the final centimeters, when the LEDs are seen well spaced in the image plane, is high. Nevertheless, the other two solutions must be preferred to this one to achieve a more robust system.

The second simulation performed is a Monte Carlo analysis: one hundred sets of uncorrelated and randomly generated initial conditions are used to assess the handover performance. The random initial conditions are taken from Gaussian distributions. The parameter used to evaluate the handover performance is the stabilization time required by the GNC system to stabilize the chaser around the Station Keeping point from which the FA begins. The simulations show how the three patterns have similar sensitivity to the initial conditions variation. The sensitivity is evaluated computing the mean and the standard deviation of the stabilization time.

The third and final simulation performed is again a Monte Carlo analysis, but the randomly generated parameter is the standard deviation of the pixel detection error, σ_{pix}. One hundred values of this parameter are taken from a uniform distribution, and the indicator of performance is again the stabilization time, T_{stab}. The simulation shows a peculiar step behaviour in the growth of T_{stab} as σ_{pix} increases. In the case of Pattern 1 and 3, only one major step can be observed, and occurs at high values of σ_{pix}. For Pattern 2, instead, the first jump of T_{stab} is observed at relatively lower values of σ_{pix}, and it is followed by two further jumps. After the first jump, the stabilization time required by Pattern 2 is already comparable to the time required by the other two patterns in the worst-case scenario, so towards the right end of the uniform distribution chosen for σ_{pix}. After the third jump, Pattern 2 requires almost twice the time required in the worst-case by Pattern 1 and 3. The sensitivity of Pattern 2 with respect to an increase in the pixel detection error is definitely higher, making this solution less reliable then the others.

An explanation for the observed phenomenon is presented in what follows. In Pattern 2, the LEDs composing the vertical axis of the cross-shape are closer then in the other two solutions. This implies that, when the two satellites are still far apart, the LEDs are seen extremely close to each other in the image plane. Only a few pixels separate them in each picture, so a given pixel error yields a larger state estimation uncertainty. It follows that the spacing between the LEDs composing the pattern is a very important design parameter for the optimization of the VBN algorithm.

To conclude, the similarity of the performance of Pattern 1 and 3 suggests that it is possible to design a solution which yields satisfactory docking accuracy without requiring complex hardware integration. This makes the proposed Vision-Based Navigation algorithm applicable to a large variety of CubeSats. The most important feature of a pattern, to be accounted for at design and feasibility stages, is the spacing between the LEDs. The spacing must also account for the Field of View of the vision sensor, to avoid the risk of an LED exiting the image plane during the final centimeters of the mission.

In future works, a model for the absolute attitude control of the target should be added, so to better understand the robustness of the proposed new pattern to the coupled rotations and translations of the two satellites. The mission must also be extended to include the Guidance Navigation and Control system that will operate the satellite from the earlier phases of the rendezvous to the beginning of the Final Approach. This extension would help simulating the handover phase in a more realistic way. Finally, the mechanical design of a docking system which can fit in the area constraints defined in this research should be developed.

Appendix A

Variable Mass

Assuming constant propellant mass flow $\dot{m}_c$ ($\mathrm{m\,s^{-1}}$), constant total thrust force F_{tot}, and negligible start and stop transient, the following relation can be written [44]:

$$I_{sp} = \frac{F_{tot}}{\dot{m}_c g_0} \tag{A.1}$$

where:

- I_{sp}, known as specific impulse, represents the thrust per unit propellant weight flow rate, and it is expressed in seconds; it's a parameter depending on the thrusters used and on the type of fuel (see Section 2.5);
- F_{tot} is given by the sum of the force produced by all the thrusters present in the spacecraft;
- g_0 is Earth's gravitational constant, and the average value at sea level, $g_0 = 9.8066 \mathrm{m/s^2}$ is considered [44].

An equation describing the time variation of the mass can be written as:

$$\dot{m}_c = \frac{F}{I_{sp} g_0} \tag{A.2}$$

where the product $c = I_{sp} g_0$ is commonly referred to as *effective exhaust velocity*, and corresponds to the average velocity at which the propellant is being ejected from the spacecraft.

In Simulink, thanks to an integrator block in the Laplace domain, the constant thrust force generated by the RCS can be used to compute the time varying mass.

Bibliography

[1] Bradley W. Carroll. «The Delicate Dance of Orbital Rendezvous». In: *American Journal of Physics* 87 (2019), p. 627.

[2] *Cosmos 186 & 188*. URL: https://en.wikipedia.org/wiki/Kosmos_186_and_Kosmos_188.

[3] URL: https://www.esa.int/Science_Exploration/Human_and_Robotic_Exploration/Astronauts/Rendezvous_and_docking.

[4] H. Heidt, J. Puig-Suari, A. Moore, S. Nakasuka, and R. Twiggs. «CubeSat: A new Generation of Picosatellite for Education and Industry Low-Cost Space Experimentation». In: *AIAA/USU Conference on Small Satellites*. Aug. 2000.

[5] *CubeSat Design Specification Rev. 13*. Tech. rep. 13. San Luis Obispo, CA: California Polytechnic State University, Feb. 2014.

[6] C. Pirat. «Guidance, Navigation and Control for Autonomous Rendezvous and Docking of Nano-Satellites». PhD thesis. Lausanne: École Polythecnique Fédérale de Lausanne (EPFL), Dec. 2018. URL: https://infoscience.epfl.ch/record/261367.

[7] P. Bodin, R. Noteborn, R. Larsson, T. Karlsson, S. D'Amico, J. Ardaens, M. Delpech, and J. Berges. «The PRISMA Formation Flying Demonstrator: Overview and Conclusions from the Nominal Mission». In: *Advances in the Astronautical Sciences* 144 (Feb. 2012), pp. 441–460.

[8] W. Fehse. *Automated rendezvous and docking of spacecraft*. Cambridge University Press, 2003.

[9] B. E. Tweddle. «Computer Vision Based Navigation for Spacecraft Proximity Operations». MA thesis. Massachussets Institute of Technologies (MIT), Feb. 2010.

[10] Lance B. Gatrell, William A. Hoff, and Cheryl W. Sklair. «Robust image features: concentric contrasting circles and their image extraction». In: *Cooperative Intelligent Robotics in Space II*. Ed. by William E. Stoney. Vol. 1612. International Society for Optics and Photonics. SPIE, 1992, pp. 235–244.

[11] Martin A. Fischler and Robert C. Bolles. «Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography». In: 24.6 (1981).

[12] Francesco Sansone, Francesco Branz, and Alessandro Francesconi. «A relative navigation sensor for CubeSats based on LED fiducial markers». In: *Acta Astronautica* 146 (May 2018), pp. 206–215.

[13] L. Kneip, D. Scaramuzza, and R. Siegwart. «A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation». In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2011, pp. 2969–2976.

[14] D.G. Lowe. «Object recognition from local scale-invariant features». In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. 1999, pp. 1150–1157.

[15] Richard O. Duda and Peter E. Hart. «Use of the Hough transformation to detect lines and curves in pictures». In: *Commun. ACM* 15 (1972), pp. 11–15.

[16] Bong Wie, Haim Weiss, and Ari Arapostathis. «Quaternion feedback regulator for spacecraft eigenaxis rotation». In: *Journal of Guidance Control and Dynamics* 12 (May 1989), pp. 375–380.

[17] Camille Pirat, P.-A. Mäusli, Roger Walker, Finn Ankersen, and Volker Gass. «Guidance, Navigation and Control for Autonomous Cooperative Docking of CubeSats». In: 2018.

[18] Huihui Bai, Chunqing Huang, and Jianping Zeng. «Robust nonlinear H-output-feedback control for flexible spacecraft attitude manoeuvring». In: *Transactions of the Institute of Measurement and Control* 41.7 (2019), pp. 2026–2038.

[19] Gabriel Popescu. «Pixel geolocation algorithm for satellite scanner data». In: June 2014.

[20] Nicoletta Bloise, Elisa Capello, Matteo Dentis, and Elisabetta Punta. «Obstacle Avoidance with Potential Field Applied to a Rendezvous Maneuver». In: *Applied Sciences* 7 (Oct. 2017), p. 1042.

[21] Bruce Yost et al. *State-of-the-Art Small Spacecraft Technology*. NASA/TP—20210021263. Moffett Field, CA: Ames Research Center, Oct. 2021.

[22] Oliver Montenbruck and Eberhard Gill. *Satellite Orbits: Models, Methods and Applications*. Springer Berlin Heidelberg, 2000.

[23] F. Landis Markley and John L. Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, 2014.

[24] *GOMspace 6DOF Reaction Control System.* URL: https://gomspace.com/shop/subsystems/attitude-orbit-control-systems/nanoprop-6u-propulsion.aspx.

[25] Arantes Gilberto, Luiz Martins-Filho, and Adrielle Santana. «Optimal On-Off Attitude Control for the Brazilian Multimission Platform Satellite». In: *Mathematical Problems in Engineering* 2009 (Oct. 2009).

[26] *GOMspace 3-axis magnetorquer.* URL: https://gomspace.com/shop/subsystems/attitude-orbit-control-systems/nanotorque-gst-600.aspx.

[27] *GOMspace Reaction Wheel.* URL: https://gomspace.com/shop/subsystems/attitude-orbit-control-systems/nanotorque-gsw-600.aspx.

[28] *Clyde Space RW210 reaction wheel.* URL: https://www.aac-clyde.space/what-we-do/space-products-components/acds/rw210-15.

[29] W J Larson and J R Wertz. *Space Mission Analysis and Design.* 2. Microcosm and Kluwer Academic Publishers, Dordrecht, 1992.

[30] URL: https://www.technologiesinindustry4.com/2021/07/earth-magnetic-field.html.

[31] Mark L. Psiaki. «Magnetic Torquer Attitude Control via Asymptotic Periodic Linear Quadratic Regulation». In: *Journal of Guidance Control and Dynamics* 24 (2000), pp. 386–394.

[32] M. Salim Farissi, Stefano Carletta, Augusto Nascetti, and Paolo Teofilatto. «Implementation and Hardware-In-The-Loop Simulation of a Magnetic Detumbling and Pointing Control Based on Three-Axis Magnetometer Data». In: *Aerospace* 6.12 (2019). URL: https://www.mdpi.com/2226-4310/6/12/133.

[33] Ali Kasiri and Farhad Saberi. «Agile Multi-Targeting Spacecraft Control Via Backstepping-Sliding mode Approach». In: Dec. 2019.

[34] Rafał Wiśniewski and Mogens Blanke. «Fully magnetic attitude control for spacecraft subject to gravity gradient». In: *Automatica* 35.7 (1999), pp. 1201–1214.

[35] Camille Pirat, Finn Ankersen, Roger Walker, and Volker Gass. «Vision Based Navigation for Autonomous Cooperative Docking of CubeSats». In: *Acta Astronautica* 146 (2018), pp. 418–434.

[36] Lennon Rodgers, Simon Nolet, and David W. Miller. «Development of the miniature video docking sensor». In: *Modeling, Simulation, and Verification of Space-based Systems III.* Ed. by Pejmun Motaghedi. Vol. 6221. International Society for Optics and Photonics. SPIE, 2006, pp. 121–132.

[37] Vadim Indelman. *Vision-Aided Navigation course.* Haifa, Israel: Technion - Israel Institute of Technology, 2020.

[38] *Monocular camera.* URL: https://www.baslerweb.com/en/products/cameras/area-scan-cameras/ace/aca3800-14um/.

[39] *Monocular camera's lenses.* URL: https://www.baslerweb.com/en/products/vision-components/lenses/basler-lens-c125-0418-5m-p-f4mm/.

[40] *Modern LEDs dimension.* URL: https://it.rs-online.com/web/p/led/2285821/?cm_mmc=IT-PLA-DS3A-_-google-_-PLA_IT_IT_Display_e_optoelettronica_Whoop-_-(IT:Whoop!)+LED-_-2285821&matchtype=&pla-307757506505&gclid=CjwKCAiA7dKMBhBCEiwAO_crFO2yDGeEuS426CPSEbe0W_JjEb_P7Ui3A0oYl48CDDlqWxAw_WinyRoC4FUQAvD_BwE&gclsrc=aw.ds.

[41] URL: https://www.princetoninstruments.com/learn/camera-fundamentals/pixel-size-and-camera-resolution.

[42] John L. Crassidis and John L. Junkins. *Optimal Estimation of Dynamic Systems.* 2nd. Chapman & Hall/CRC, 2011.

[43] Sigurd Skogestad and Ian Postlethwaite. *Multivariable feedback control: Analysis and Design.* Wiley, 2005.

[44] George P. Sutton and Oscar Biblarz. *Rocket Propulsion Elements.* 9th. John Wiley & Sons, 2017.