A review of balancing methods for satellite simulators

Rodrigo Cardoso da Silva[a,*], Renato Alves Borges[a], Simone Battistini[b], Chantal Cappelletti[c]

[a] *Dept. of Electrical Engineering, Univ. of Brasília, Campus Darcy Ribeiro, Brasília-DF CEP 70910-900, Brazil*
[b] *Department of Engineering and Mathematics, Sheffield Hallam University, Sheffield, S1 1WB, UK*
[c] *Dept. of Mechanical, Materials, and Manufacturing Engineering, Univ. of Nottingham, Nottingham NG7 2RD, UK*

Abstract

This article presents a general review of the existing methods for balancing satellite simulators. The mathematical modeling for the kinematics and the dynamics of the satellite simulator is described and uniformed throughout the various methods. The available balancing methods are classified in a handful group of categories, which consider whether the analyzed method uses linear or nonlinear filtering, adaptive control or even classical regression methods. Details are given on the features which may be added to the dynamic model and how they are modeled, on the discretization methods used, on the models observability and on the solution convergence. Finally, the overall performance of each method is shown, as well as the circumstances in which each method presents advantages and disadvantages, guiding on which method to use.

Keywords: satellite simulator, balancing methods, filtering, adaptive control

1. Introduction

Attitude simulators are a great resource for designing and testing attitude determination and control algorithms [1] for spacecraft. They have been used for decades together with other simulators (magnetic field simulators [2], sun sensors [3]) to reproduce the operational conditions of a satellite in space and to experimentally validate attitude determination and control algorithms and devices [4, 5, 6]. The recent increase in the number of small satellite initiatives has made attitude simulators even more popular and accessible for research groups in universities and other institutions [3, 7, 8, 9].

In order to simulate the attitude motion of a satellite in space, many simulators rely on an air-bearing to sustain the platform where the spacecraft is positioned. Air-bearings allow to obtain an almost friction-less environment and the consequent possibility of torque-free rotational motion. In some cases, the air-bearing can be even extended to obtain force-free translational motion [10] to simulate docking and rendez-vous tasks together with attitude problems such as inertial pointing or detumbling.

A pre-requisite for conducting spacecraft attitude simulations is to have a neutral environment in terms of mechanical torques. This means that the platform that carries the spacecraft should be balanced so that no gravitational torques exist. When simulating 3D rotational movements, a simulator may present pendulum-like motion, as a consequence of the unbalance produced by the distance between the Center of Rotation (CR) and the Center of Mass (CM) of the platform. In order to properly reproduce attitude maneuvers in the Low Earth Orbit (LEO), for example, this torque should be reduced to values as low as 10^{-6} Nm [11, 12, 13].

Although in some cases balancing the platform can be performed manually, for example if the required accuracy is not high, an automatic balancing procedure is preferable because it grants more accuracy and less efforts. Balancing is, in fact, an operation that needs to be performed on the testbed every time the set-up of the experiment is changed, e.g. every time the spacecraft undergoes some modification (changing its position or structural elements). Therefore, an automatic method avoids a human operator to repeat the procedure multiple times.

This paper presents a general review of automatic balancing methods for spacecraft attitude simulators. The goal is to identify, analyze, and compare the main features and the limitations of these methods. This perspective will allow to show how some of these methods may be changed in order to create new algorithms that take into account specific characteristics of the simulators. Furthermore, the consequences of making simplifications in the mathematical model of the system will be presented.

The remaining sections of this work are organized as follows. In Sec. 2 the satellite simulator model will be described, including the kinematics and the dynamics equations that apply, the balancing problem will be defined and some design remarks are given. In Sec. 3 the performance

*Corresponding author.
Email addresses: rcsilva@lara.unb.br (Rodrigo Cardoso da Silva), raborges@unb.br (Renato Alves Borges), s.battistini@shu.ac.uk (Simone Battistini), chantal.cappelletti@nottingham.ac.uk (Chantal Cappelletti)

that may be attained by manually balancing a satellite simulator is firstly presented along with the disadvantages of adopting this strategy, followed by a systematization of the methods for identifying the physical parameters of the simulator - such as its CM and its inertia tensor parameters. In Sec. 4, techniques that depend on active control torques are presented, focusing on the use of the complete nonlinear model of the system. Finally, Sec. 5 provides manners of assessing the performance of a given balancing method and Sec. 6 concludes with final remarks.

2. Design remarks on models and hardware

Before delving into the details of the various balancing methods, it is important to define the standards which will be adopted in order to describe the simulator kinematics and dynamics. Regarding the simulator kinematics, there are many ways of representing the attitude of the testbed, such as Direct Cosine Matrices (DCMs), rotation matrices in various sequences, quaternions, dual quaternions and Euler-Rodrigues parameters. In this work, only two conventions are employed, the 3-2-1 sequence of the Euler angles and the quaternions. With the Euler angles, attitude is determined by three independent angles, ϕ (roll angle, around the body x-axis), θ (pitch angle, around the body y-axis) and ψ (yaw angle, around the body z-axis). Also, when discretizing the testbed models for simulation, it is important to know the rate at which these angles vary. This rate is given by

$$\dot{\mathbf{E}} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sec\theta\sin\phi & \sec\theta\cos\phi \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} , \tag{1}$$

where $\omega_{x,y,z}$ are the components of the angular velocity vector $\boldsymbol{\omega}$. This equation can be integrated along with the dynamic model equations in order to simulate the testbed motion.

The other convention used in this work is the rotation quaternion. The rotation quaternion follows the quaternion algebra for representing vector rotations. In this case, the representation of the transformation between two vectors $\mathbf{v}$ and $\mathbf{w}$ in the body and inertial frame can be given by the following equation:

$$\mathbf{w} = \mathbf{q} \otimes \mathbf{v} \otimes \mathbf{q}^* = \mathbf{Q}_i^b \cdot \mathbf{v} , \tag{2}$$

where $\mathbf{q}$ is the rotation quaternion, $\otimes$ the quarternion product, and $\mathbf{q}^*$ its conjugate, and $\mathbf{Q}_i^b$ is the 3×3 matrix-equivalent rotation operator whose entries are given as functions of the four components of the rotation quaternion, q_0, q_1, q_2 and q_3. Also, the rates at which each quaternion component varies are given by

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Omega}\mathbf{q} , \tag{3}$$

where $\boldsymbol{\Omega}$ is the 4×4 skew-symmetric matrix of $\boldsymbol{\omega}$. Special attention must be taken to the following remarks:

1. The adopted convention assumes that the q_0 component is the real part of the quaternion. In other works, instead, the components are numbered from 1 to 4, with q_4 being the real part.
2. The adopted convention assumes a *world-to-body* perspective. Further understanding on the differences between the *world-to-body* and the *body-to-world* perspectives and its respective mathematical consequences is given in [14].

In this work, the dynamic model of the testbed is given by the Euler Equations of Motion (EOM)

$$\begin{aligned} \frac{d\boldsymbol{\Gamma}}{dt} &= \mathbf{M} , \\ \dot{\boldsymbol{\Gamma}} &= \dot{\boldsymbol{\Gamma}}\Big|_b + \boldsymbol{\omega} \times \boldsymbol{\Gamma}|_b = \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \\ \mathbf{M} &= \mathbf{r} \times \mathbf{F}_{\text{G}} = \mathbf{r} \times m\mathbf{g}_{\text{b}} , \end{aligned} \tag{4}$$

where Γ is the angular momentum around the center of rotation, J is the inertia tensor, and the external torques are given solely by the gravitational torque as a consequence of the CM displacement vector $\mathbf{r}$, the testbed mass m, and the local gravity in the body frame $\mathbf{g}_{\text{b}}$. Throughout the text, depending on the balancing method, terms may be added to represent additional external torques or additional angular momentum, as a consequence of the adopted actuators or disturbances present in the system.

Another way of describing the movement of a testbed restricted to rotational movements is given by

$$\begin{aligned} \frac{d\boldsymbol{\Gamma}}{dt} &= \mathbf{M} , \\ \dot{\boldsymbol{\Gamma}} &= \mathbf{r} \times m\ddot{\mathbf{r}} + \boldsymbol{\omega} \times (\mathbf{r} \times m\dot{\mathbf{r}}) + \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}) , \end{aligned} \tag{5}$$

which is useful when performing the simplifications required by the static estimation and linear filtering methods. Once the equations have been simplified, the angular accelerations can be expressed as [15]

$$\dot{\boldsymbol{\omega}} = \begin{bmatrix} \frac{mg}{J_{\text{x}}}\left(-r_y\cos\phi\cos\theta + r_z\sin\phi\cos\theta\right) \\ \frac{mg}{J_{\text{y}}}\left(r_x\cos\phi\cos\theta + r_z\sin\theta\right) \\ \frac{mg}{J_{\text{z}}}\left(-r_x\sin\phi\cos\theta - r_y\sin\theta\right) \end{bmatrix} , \tag{6}$$

which will be referred to as the "Euler Simplified Model" (ESM) throughout the text .

Fig. 1 shows the relevant parameters of a spacecraft attitude simulator and its Automatic Balancing System (ABS). The origin O of the inertial frame axes ($\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$) coincides with the the origin of the body frame ($\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$) and the Center of Rotation (CR) of the system and the $\mathbf{Z}$ axis parallel to the gravity vector $\mathbf{g}$. The ABS is composed of Movable Mass Units (MMU) responsible for controlling the CM position, which is indicated by the $\mathbf{r}$. In this work, it is assumed that there are three MMUs in the system, each one aligned with the body frame axes, weighting $\mathbf{m}_i$, $i \in (x, y, z)$ and capable of moving up to

$\mathbf{d}_i$, $i \in (x, y, z)$, although it is also possible to achieve the same balancing capabilities with other configurations, such as in [16].

Regarding the balancing performance an ABS may achieve, the system mass properties and the displacement attained by the MMUs determine an upper and a lower bound to how much the CM offset may be changed [11]. The CM displacement vector $\Delta\mathbf{r}_{CM}$ has its components defined by

$$\Delta\mathbf{r}_{CM,i} = \frac{m_i \mathbf{d}_i}{M}, i \in (x, y, z), \tag{7}$$

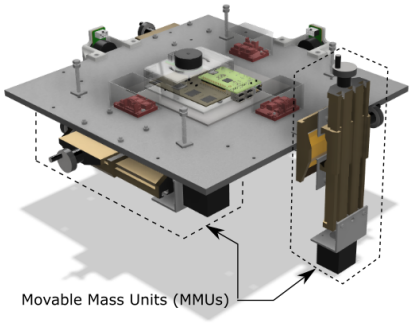
in which $M = \sum_i m_i + m_{sim}$, with m_{sim} being the mass of the simulator without the MMUs. As one may notice, the ideal maximum balancing attainable by the ABS $\Delta\mathbf{r}_{CM,max}$ is achieved when all the MMUs start the balancing procedure in one extremity of its excursion and finish in the opposite extremity, moving $\mathbf{d}_{i,max}$ each. This maximum balancing determines how the simulator must be pre-balanced with fixed masses in order for the ABS to be capable of compensating the residual unbalancing. On the other hand, the resolution of the MMUs, *i.e.* the minimum displacement $\mathbf{d}_{i,min}$ that each MMU may perform, determines the minimum CM offset $\mathbf{r} = \Delta\mathbf{r}_{CM,min}$ that may be attainable by the ABS, corresponding to a minimum residual gravitational torque $\tau_{G,min} = \Delta\mathbf{r}_{CM,min} \times M\mathbf{g}$. As emphasized in [11], this is only an ideal minimum, since the balancing quality may be affected by backlash in the mechanical system, axes misalignment, measurement errors or even little variations in the simulator components (such as cables).

This paper compares the balancing methods of many different testbeds found in the literature. Table 1 presents them in a synthetic way resuming the main features for the ease of comparison. It can be seen that the most employed configuration is the table-top air-bearing with sliding masses used for balancing. While in some cases balancing is carried out manually, some platforms have automatic balancing systems which includes the use of sensors, computers, and balancing algorithms, which will be described in the following sections.

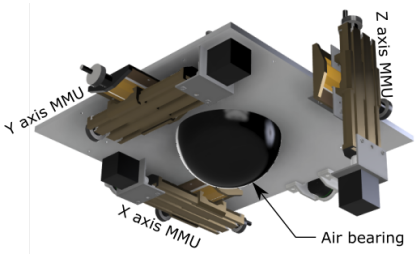
3. Parameters identification

A reasonable starting point to assess the performance of an automatic balancing method is to compare its performance to that obtained by manually balancing the testbed. Some works in the literature demonstrate that manual balancing allows to obtain levels of gravitational torque disturbances low enough to test the attitude determination and control algorithms of a given satellite. In [19, 26], manually balancing an air bearing testbed resulted in a gravitational torque as low as 0.01 Nm.

Although these results proved to be good enough to test most ADCS techniques, this balancing method has the disadvantage of requiring too much time - even hours - and expertise from the operator. Considering that the balancing procedure should be repeated very often, *e.g.* every time a change is made in the satellite mock-up or in the simulator as a whole, manual balancing techniques can result impractical in many cases.



(a) Upper view



(b) Bottom view

Figure 1: Spacecraft simulator elements.

Borrowing from [27] the categorization of the balancing techniques and adapting it to cover all methods in a general manner, it is possible to divide the automatic balancing techniques in the following categories: I) Batch estimation techniques, which include various formulations of the Least Squares Method (LSM), such as the torque method, the momentum integral method, the energy balance method and the equilibrium points method [28, 11, 8, 19, 29, 30, 21, 31, 15, 20, 32]; II) Filtering techniques, which include the various versions of the Kalman filter for linear and nonlinear systems[20, 28, 33, 3, 34, 35]; and III) Active control balancing techniques, which range from linear to nonlinear control methods [11, 27, 7, 36, 37].

3.1. Batch estimation techniques

Batch techniques rely on determining the testbed offset $\mathbf{r}$ and inertia J in order to compensate the CM offset in a single step or iteratively. As one may realize from the execution of these methods, in order to achieve reasonable results, it is desirable to repeat the estimation-correction cycle of the CM position iteratively. Although these methods may be implemented in an automatic balancing system, they might result in a time consuming solution to the problem.

3.1.1. Least Squares Methods

Since Eq. (6) directly relates the desired CM offset with other known variables, it is possible to determine

Table 1: Existing testbeds and balancing methods.

Institution	**Platform configuration & actuators**[1]	**Hardware specs**[2]	**Balancing method & results**[3]
Georgia Tech [3]	1) Table-top; 2) Control Moment Gyros	OBC (general) w/ CPU @750MHz and 128MB(V)/128MB(non-V) memories; rate gyros res. @0.029/0.073 $^\circ/s$	Recursive LSM (mass properties identification); CM compensation method N/A
Naval Postgraduate School [7, 17]	1) Spherical rotor; 2) Sliding masses, reaction wheels	OBC TS-7200-32-16F (general) w/ CPU @200MHz and 32MB(V)/up to 16MB(non-V) memories; ADIS16400 rate gyros res. @0.05 $^\circ/s$	1) two-step (under actuated control and parameter estimation) 2) N/A (4.2 kg).
University of Brasília [8]	1) Table-top 2) Sliding masses	COTS components: Arduino Uno (specific) w/ CPU @16MHz and 2KB(V)/32KB(non-V) memories; L3GD20H rate gyros res. @0.05 $^\circ/s$	1) LSM (mass prop. identif.); iterative CM compensation. 2) achievable 3.5 $10^{-5} N.m$ (14 kg)
University of Bologna [9]	1) Table-top 2) Sliding masses	COTS components: Arduino Due (specific) w/ CPU @84MHz and 96KB(V)/512KB(non-V) memories; BNO055 rate gyros res. @0.004 $^\circ/s$ (min.)	1) two-step (under act. control and param. est.); iter. CM comp. 2) $< 5\ 10^{-5} N.m$ or $< 1\mu m$ (11.5 kg)
University of Florida [10]	1) Table-top; 2) Static weights	OBC (general) ADLS15PC w/ CPU @1.1-1.60GHz and up to 2GB(V)/up to 8GB(non-V) memories; attitude from PhaseSpace System (res. N/A)	1) Manual balancing. 2) N/A
Harbin Inst. of Technology [16]	1) Table-top; 2) Pyramidal sliding masses	OBC (general) ARK5260 w/ CPU @up to 1.60GHz and up to 2GB(V)/SSD or HDD capacity (non-V) memories; CS-ARS-03 rate gyros res. @$< 0.04^\circ/s$	1) LSM (mass prop. identif.); iterative CM compensation; 2) 5 μm final offset (platform mass N/A).
UNAM [18]	1) Table-top 2) Sliding masses	OBC 16 bit SAB80C166 (general) w/16KB ROM and 1.28MB RAM; rate gyros res. @0.05 $^\circ/s$	1) LSM 2) 0.002 Nm
Honeywell [19]	1) Umbrella 2) Prismatic actuators	OBC DS1005 (general) w/ CPU@1 GHz; rate gyros res. $0.01\sqrt{hr}$ drift	1) LSM 2) 0.06096 mm
KNT Univ. of Technology [20]	1) Dumbbell 2) Sliding masses	N/A. Rate gyros res. @0.25 $^\circ/s$	Kalman filter
Virginia Tech [21]	1) Table-top 2) Linear actuators	(general) 32-bit 133MHz Tri-M MZ104+ ZFx86 processor with 64MB of RAM.	LSM
Vietnam NSC [22, 23]	1) Table-top 2) 2 sets of sliding masses (coarse and fine)	N/A	1) Adaptive estimation and L1 adaptive control 2) 5E-4 mm
University of Sidney [24]	1) Table-top 2) Counterweights	COTS components: Arduino Mega 2560 (specific) w/ CPU @16MHz and 8KB(V)/256KB(non-V) memories; L3GD20H rate gyros res. @0.05 $^\circ/s$	1) Manual (xy plane only) 2) N/A
Aalto University [25]	1) Spherical rotor 2) Counterweights	N/A	1) Manual 2) N/A

Notes: [1]: for air bearing based platforms, the platform configuration borrows the classification from [1]. The **Platform configuration & actuators** column is divided in two main info: 1) the platform configuration; 2) the available balancing devices (e.g. sliding masses.
[2]: The **Hardware specs** column focuses on: the capabilities of the processing unit, which may be responsible for balancing the platform only or simulating the spacecraft ADCS as well; and the rate gyros resolution, since this spec is closely related to the mass properties estimation precision. Resolution may be calculated from range (^{o}/s) and resolution (bits) characteristics. **General** and **Specific** in parenthesis refers to the OBC being application specific (balancing) or not. V/Non-V refers to the memory being volatile or not.
[3]: The **Balancing method & results** column is divided in two info: 1) the balancing method; and 2) the results achieved (minimum gravitational torque in Nm, or imbalance length in mm) and the corresponding platform mass (in kg).

it directly. However, some of these variables carry with themselves an inherent noise (e.g. the angular velocities measured from the gyroscopes, the attitude angles measured from computer vision or sensor combination). To reduce the error caused from sensor noise and other inherent model uncertainties, one may use the Least Squares Method (LSM), which oversamples the r equation aiming to reach an estimation that optimally minimizes the mean squared error (MSE) between the samples and the true unbalance vector. To use this method, Eq. (6) is firstly discretized, making it possible to separate the unbalance vector as a 3×1 vector of unknowns, giving

$$\Delta\boldsymbol{\Omega} = \boldsymbol{\phi}\mathbf{r}$$
$$\underbrace{\begin{bmatrix} \Delta\omega_x^k \\ \Delta\omega_y^k \\ \Delta\omega_z^k \end{bmatrix}}_{\Delta\boldsymbol{\Omega}} = \underbrace{\begin{bmatrix} 0 & \phi_{12} & \phi_{13} \\ \phi_{21} & 0 & \phi_{23} \\ \phi_{31} & \phi_{32} & 0 \end{bmatrix}}_{\boldsymbol{\phi}} \underbrace{\begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}}_{\mathbf{r}}, \quad (8)$$

in which the superscript indicates the moment at which the measurement is taken. Then, after oversampling Eq.(8), the unbalance vector may be obtained as

$$\mathbf{r} = [\boldsymbol{\phi}_{aug}^T \boldsymbol{\phi}_{aug}]^{-1} \boldsymbol{\phi}_{aug}^T \Delta\boldsymbol{\Omega}_{aug}\ , \quad (9)$$

in which the *aug* subscript denotes row-augmentation of the vectors in Eq. (8).

A batch of 500 measurements is sufficient to estimate the CM displacement, according to the repeated experiments conducted in [8]. After having determined $\mathbf{r}$, it can be compensated in the testbed by moving the masses to provide the possibility of translating the CM offset in the 3D space. These masses are usually placed parallel to the testbed axes, orthogonally, although they might be also disposed in other configurations, such as the pyramid formation shown in [16]. In any case, it is important to guarantee that the chosen configuration is capable of moving the CM offset tridimensionally. The required mass displacement on each axis to compensate $\mathbf{r}$ is equal to $-\mathbf{d}_i,\ i \in (x, y, z)$.

3.1.2. Torque method

Batch estimation methods focus only on determining the CM offset components. However, in order to obtain

unbiased estimations, the principal components of the inertia tensor of the testbed must be precisely determined too. This can be accomplished by numerically calculating the inertia tensor components by approximating the mass distribution of the testbed [38], or by having a reliable description of the system in a 3D Computer Aided Design (CAD) model, which provides a proper estimation of the inertia tensor. The inertia tensor components may also be estimated in an experiment by analyzing the testbed oscillation in orthogonal axes [38]. One disadvantage of determining the inertia components before performing the estimation shown in Eq. (9) is that, besides adding another step in the balancing procedure, it may lead to bias in the estimations, since the inertia tensor components may change considerably with any change in the distribution of components mounted on the testbed.

One way to avoid the need of estimating the inertia tensor components is to arrange the terms of the testbed dynamic model in such a manner that both the CM offset and the inertia tensor components may be estimated simultaneosly [20, 27]. First, it is assumed that momentum exchange devices, *e.g.* reaction wheels, are present in the system. The model in Eq. (4) may be modified in order to take into account the torque generated by these devices, $\boldsymbol{\Gamma}_*$, leading to

$$\frac{d\boldsymbol{\Gamma}}{dt} = \frac{d\mathbf{J}\boldsymbol{\omega}}{dt} + \frac{d\boldsymbol{\Gamma}_*}{dt} = \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times\mathbf{J}\boldsymbol{\omega} + \dot{\boldsymbol{\Gamma}}_* + \boldsymbol{\omega}\times\boldsymbol{\Gamma}_* \ , \quad (10)$$

in which the Basic Kinematic Equation (BKE) is used. By using the matrix form of the cross product and rearranging the terms, the equation

$$\mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times\mathbf{J}\boldsymbol{\omega} = -\dot{\boldsymbol{\Gamma}}_* - \boldsymbol{\omega}\times\boldsymbol{\Gamma}_* - [\mathbf{g}\times]m\mathbf{r} \ , \quad (11)$$

may be written as

$$\left[\ \dot{\boldsymbol{\Omega}} + \boldsymbol{\omega}\times\boldsymbol{\Omega} \quad [\mathbf{g}\times]\ \right] \underbrace{\begin{bmatrix} \tilde{\mathbf{J}} \\ m\mathbf{r} \end{bmatrix}}_{\mathbf{x}} = -\dot{\boldsymbol{\Gamma}}_* - \boldsymbol{\omega}\times\boldsymbol{\Gamma}_* \quad (12)$$

where the terms $\boldsymbol{\Omega}$, $\tilde{\mathbf{J}}$ and $[\mathbf{g}\times]$ (in the body frame) are given by

$$\boldsymbol{\Omega} = \begin{bmatrix} \omega_1 & 0 & 0 & -\omega_2 & -\omega_3 & 0 \\ 0 & \omega_2 & 0 & -\omega_1 & 0 & -\omega_3 \\ 0 & 0 & \omega_3 & 0 & -\omega_1 & -\omega_2 \end{bmatrix} \quad (13)$$

$$\tilde{\mathbf{J}} = \begin{bmatrix} J_x & J_y & J_z & J_{xy} & J_{xz} & J_{yz} \end{bmatrix}^T \quad (14)$$

$$[\mathbf{g}\times] = g\begin{bmatrix} 0 & -\cos\phi\cos\theta & \sin\phi\cos\theta \\ \cos\phi\cos\theta & 0 & \sin\theta \\ -\sin\phi\cos\theta & -\sin\theta & 0 \end{bmatrix} \quad (15)$$

providing one way of isolating the vector $\mathbf{x}$ with the desired parameters to be estimated. In order to reduce the effects of deriving the angular rates, which may present considerable amount of noise, Eq. (12) may be integrated over time, yielding

$$\underbrace{\begin{bmatrix} \boldsymbol{\Omega} + \int_{t_0}^{t} \boldsymbol{\omega}\times\boldsymbol{\Omega}\ dt & \int_{t_0}^{t}[\mathbf{g}\times]\ dt \end{bmatrix}}_{\boldsymbol{\Phi}} \underbrace{\begin{bmatrix} \tilde{\mathbf{J}} \\ m\mathbf{r} \end{bmatrix}}_{\mathbf{x}} = \underbrace{-\boldsymbol{\Gamma}_* - \int_{t_0}^{t} \boldsymbol{\omega}\times\boldsymbol{\Gamma}_*\ dt}_{\mathbf{S}} \ , \quad (16)$$

which provides a least squares estimation method in a similar manner to Eq. (9),

$$\mathbf{x} = (\boldsymbol{\Phi}_{aug}^T\boldsymbol{\Phi}_{aug})^{-1}\boldsymbol{\Phi}_{aug}^T\mathbf{S}_{aug} \ , \quad (17)$$

where the $\boldsymbol{\Phi}_{aug}$ and $\mathbf{S}_{aug}$ terms are given by

$$\boldsymbol{\Phi}_{aug} = \begin{bmatrix} \boldsymbol{\Phi}_0 & \boldsymbol{\Phi}_1 & \dots & \boldsymbol{\Phi}_k \end{bmatrix}^T_{3n\times 9} \ , \quad (18)$$

$$\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{\Omega} + \int_{t_0}^{t_k} \boldsymbol{\omega}\times\boldsymbol{\Omega}\ dt & \int_{t_0}^{t_k}[\mathbf{g}\times]\ dt \end{bmatrix}_{3\times 9} \ , \quad (19)$$

$$\mathbf{S}_{aug} = \begin{bmatrix} \mathbf{S}_0 & \mathbf{S}_1 & \dots & \mathbf{S}_k \end{bmatrix}^T_{3n\times 1} \ , \quad (20)$$

$$\mathbf{S}_k = \left[-\boldsymbol{\Gamma}_* - \int_{t_0}^{t_k} \boldsymbol{\omega}\times\boldsymbol{\Gamma}_*\ dt\right]_{3\times 1} \ . \quad (21)$$

3.1.3. Recursive Least Squares Method (RLS)

One way to recursively estimating the $\mathbf{x}$ vector in Eq. (17) by implementing the RLS method in [39] is through the following four steps [20],

$$\mathbf{K}_k = \frac{\lambda^{-1}\mathbf{P}_{k-1}\boldsymbol{\Phi}_k}{1+\lambda^{-1}\boldsymbol{\Phi}_k^T\mathbf{P}_{k-1}\boldsymbol{\Phi}_k} \ , \quad (22)$$

$$\boldsymbol{\alpha}_k = \mathbf{S}_k - \boldsymbol{\Phi}_k^T\mathbf{x}_{k-1} \ , \quad (23)$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{K}_k\boldsymbol{\alpha}_k \ , \quad (24)$$

$$\mathbf{P}_k = \lambda^{-1}\mathbf{P}_{k-1} - \lambda^{-1}\mathbf{K}_k\boldsymbol{\Phi}_k^T\mathbf{P}_{k-1} \ , \quad (25)$$

where $\boldsymbol{\alpha}_k$ is the residual estimation error at time k and λ is the forgetting factor, a scalar which controls the influence of past observations in the estimation process.

Several implementations of the RLS method are presented in [33], with each one using a different way to avoid amplifying the noise from sensors through differentiation: either by using integrated forms in the model equation or by adopting tracking differentiators (TD) and extended tracking differentiators (ETD) in the estimation process.

3.1.4. Classical Levenberg-Marquadt (CLM) estimation

CLM is another method used to estimate the $\mathbf{x}$ vector in Eq. (17) [20]. The CLM algorithm is a combination of the Gauss-Newton Algorithm (GNA) and the Gradient Descent (GD) algorithm, and it decides between these two algorithms by means of a λ parameter in the result update equation (for large values of λ, the GD algorithm takes place, whereas for small values GNA takes place).

3.2. Filtering methods

As stated in Subsec. 3.1.3, recursive estimation is particularly useful when process noise is present in the system. The Kalman filter and its extensions are still the most common solutions currently used, being widely applied in on-board attitude estimators.

As previously mentioned, the main motivation here is to determine the unbalance vector. One of the most important characteristics of determining the unbalance vector through a Kalman filter is related to its ability to model nonstationary dynamic systems driven by random process noise. Moreover, high dimension matrix inversion may be avoided and there is no need to store measurements (recursive versus batch processing), making possible to implement the balancing method in an embedded system with less computational effort.

Consider the following general system

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}), \quad \mathbf{w}_k \sim (0, \mathbf{Q}_k) \tag{26}$$

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k, \mathbf{v}_k), \quad \mathbf{v}_k \sim (0, \mathbf{R}_k) \tag{27}$$

where $\mathbf{x}_k$ is the state vector, $\mathbf{u}_k$ the input signal, $\mathbf{w}_k$ the system additive noise, $\mathbf{v}_k$ the measurement additive noise, and $\mathbf{f}_k$ and $\mathbf{h}_k$ are functions of the previous variables.

In the linear case $\mathbf{f}_k$ and $\mathbf{h}_k$ reduce to linear state matrices. For this kind of systems, the Kalman filter is the main option for the recursive estimation. Specifically, as shown in [28], Eq. (8) can be rewritten as Eq.(28), leading to the linear Eqs. (29) and (30),

$$\begin{bmatrix} \omega_x^{k+1} \\ \omega_y^{k+1} \\ \omega_z^{k+1} \end{bmatrix} = \begin{bmatrix} \omega_x^{k} \\ \omega_y^{k} \\ \omega_z^{k} \end{bmatrix} + \begin{bmatrix} 0 & \phi_{12}^k & \phi_{13}^k \\ \phi_{21}^k & 0 & \phi_{23}^k \\ \phi_{31}^k & \phi_{32}^k & 0 \end{bmatrix} \begin{bmatrix} r_x^{k} \\ r_y^{k} \\ r_z^{k} \end{bmatrix} \tag{28}$$

$$\underbrace{\begin{bmatrix} \omega_x^{k+1} \\ \omega_y^{k+1} \\ \omega_z^{k+1} \\ r_x^{k+1} \\ r_y^{k+1} \\ r_z^{k+1} \end{bmatrix}}_{\mathbf{x}_{k+1}} = \underbrace{\left[\begin{array}{c|ccc} & 0 & \phi_{12} & \phi_{13} \\ \mathbf{I}_{3\times3} & \phi_{21} & 0 & \phi_{23} \\ & \phi_{31} & \phi_{32} & 0 \\ \hline \mathbf{0}_{3\times3} & \multicolumn{3}{c}{\mathbf{I}_{3\times3}} \end{array}\right]}_{\mathbf{A}} \underbrace{\begin{bmatrix} \omega_x^{k} \\ \omega_y^{k} \\ \omega_z^{k} \\ r_x^{k} \\ r_y^{k} \\ r_z^{k} \end{bmatrix}}_{\mathbf{x}_k} + \mathbf{w}_{6\times1} \tag{29}$$

$$\mathbf{y}_k = \mathbf{H} \cdot \mathbf{x}_k + \mathbf{v}_k, \tag{30}$$

where $\mathbf{H} = [\mathbf{I}_{3\times3}\ \mathbf{0}_{3\times3}]$ and $\mathbf{v}_k$ is the 3×1 vector of measurement noise. The observability of the state vector may be analyzed by checking the rank of the observability matrix

$$\mathcal{O} = \begin{bmatrix} \mathbf{H}_{3\times6} \\ (\mathbf{HA})_{3\times6} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3\times3}\ \mathbf{0}_{3\times3} \\ \mathbf{I}_{3\times3}\ \Phi_{3\times3} \end{bmatrix} . \tag{31}$$

On the other hand, for nonlinear systems in which the dynamic and measurement models can be expanded in first order Taylor series about the current estimate, the extended Kalman filter (EKF) can be used to estimate the CM offset components [28]. A realization for the linear approximation is given by the Jacobian matrices of the vector functions $\mathbf{f}$ and $\mathbf{h}$ as follows

$$\left\{ \underbrace{\frac{\partial \mathbf{f}_k}{\partial \mathbf{x}}}_{\mathbf{F}}\bigg|_{\mathbf{x}=\bar{x}}, \quad \underbrace{\frac{\partial \mathbf{f}_k}{\partial \mathbf{w}}}_{\mathbf{L}}\bigg|_{\mathbf{w}=\bar{w}}, \quad \underbrace{\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}}}_{\mathbf{H}}\bigg|_{\mathbf{x}=\bar{x}}, \quad \underbrace{\frac{\partial \mathbf{h}_k}{\partial \mathbf{v}}}_{\mathbf{M}}\bigg|_{\mathbf{v}=\bar{v}} \right\} \tag{32}$$

In order to develop the EKF using the nonlinear dynamic model equations in Eq. (4), consider the Euler equation of rotation given in Eq. (33),

$$\dot{\boldsymbol{\omega}} = \mathbf{J}^{-1}[\mathbf{J}\boldsymbol{\omega}\times]\boldsymbol{\omega} + \mathbf{J}^{-1}[-m\mathbf{g}_b\times]\mathbf{r} \ . \tag{33}$$

It is important to mention that the products of inertia (off-diagonal terms of the inertia matrix) are negligible. In other words, the inertia matrix J becomes diagonal. The state vector includes the angular velocities and the CM displacement of the simulator, *i.e.* $\mathbf{x} = [\omega_i r_i]^T,\ i \in \{x, y, z\}$, and, according to the notation of [40], the Jacobian $\mathbf{F}$ is given by [28]

$$\mathbf{F}_{k-1} = \begin{bmatrix} \frac{\partial f_1^{k-1}}{\partial x_1^{k-1}} & \cdots & \frac{\partial f_1^{k-1}}{\partial x_6^{k-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_6^{k-1}}{\partial x_1^{k-1}} & \cdots & \frac{\partial f_6^{k-1}}{\partial x_6^{k-1}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{f}_{k-1}^{1:3}}{\partial \mathbf{x}_{k-1}} \\ \frac{\partial \mathbf{f}_{k-1}^{4:6}}{\partial \mathbf{x}_{k-1}} \end{bmatrix} = \begin{bmatrix} \left[\frac{\partial \mathbf{f}_{1:3}^{k-1}}{\partial \mathbf{x}_{k-1}}\right]_{3\times6} \\ \hline \mathbf{0}_{3\times3}\ \ \mathbf{I}_{3\times3} \end{bmatrix} , \tag{34}$$

where the derivation of the partial derivatives $\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}}$ was made analytically and took into account the time propagation of the angular velocities, *i.e.* $\mathbf{f}_i^{k-1} = \boldsymbol{\omega}_{j,k} = \boldsymbol{\omega}_j + \dot{\boldsymbol{\omega}}_j T, i \in \{1, 2, 3\},\ j \in \{x, y, z\}$, T being the sampling time. Furthermore, the time propagation of the CM offset components, *i.e.* $\mathbf{f}_i^{k-1} = \mathbf{r}_j^k = \mathbf{r}_j^{k-1},\ i \in \{4, 5, 6\},\ j \in \{x, y, z\}$, follows from the assumption that the offset components do not change in time, except for the additive noises of the filter.

The remaining Jacobians, $\mathbf{M}$ and $\mathbf{L}$, are identity matrices since it is assumed there are only additive noises in the filtering process, and the output equation is linear, given $\mathbf{H} = [\mathbf{I}_{3\times3}\ \mathbf{0}_{3\times3}]$. The discrete EKF may be used following the standard procedure shown in [40].

For the case in which momentum exchange devices (MED) are available in the system, the model from Eq. (11) is employed, whose state vector $\mathbf{x}_1 = [\boldsymbol{\omega}_i]^T,\ i \in \{x, y, z\}$, can be written as $\mathbf{x}_1 = f(\mathbf{x}_1, \mathbf{J}, m\mathbf{r}, \mathbf{u})$, where $\mathbf{u} = -\dot{\boldsymbol{\Gamma}}_* + \boldsymbol{\Gamma}_* \times \boldsymbol{\omega}$ is the torque generated by the MEDs. In this case, the mass properties of a spacecraft simulator can be estimated by augmenting the state vector with the desired unknown parameters [33]. Then, a vector of unknown parameters $\mathbf{x}_2 = [J_i, J_{ij}, m\mathbf{r}_i]^T,\ i, j \in \{x, y, z\}$ containing the 9 unknown mass property parameters of the simulator is augmented in the model, leading to the

state-space representation

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} = \begin{bmatrix} f(\mathbf{x}_1, \mathbf{x}_2, \mathbf{u}) \\ 0 \end{bmatrix}, \tag{35}$$

where $\dot{\mathbf{x}}_2 = 0$ due to the assumption that the mass properties of the simulator are constant.

In this case, the Jacobians $\mathbf{L}$ and $\mathbf{M}$ are 12×12 and 3×3 identity matrices, respectively, and the output equation is also linear, with $\mathbf{H} = [\mathbf{I}_{3\times3}\ \mathbf{0}_{3\times9}]^T$. On the other hand, the Jacobian $\mathbf{F}$ has a more complex structure:

$$\mathbf{F}_{k-1} = \left[\begin{array}{c:c:c} \left[\dfrac{\partial \mathbf{f}_{1:3}^{k-1}}{\partial \boldsymbol{\omega}_{k-1}}\right]_{3\times3} & \left[\dfrac{\partial \mathbf{f}_{1:3}^{k-1}}{\partial \mathbf{J}_{k-1}}\right]_{3\times6} & \left[\dfrac{\partial \mathbf{f}_{1:3}^{k-1}}{\partial m\mathbf{r}_{k-1}}\right]_{3\times6} \\ \hdashline \mathbf{0}_{9\times3} & \multicolumn{2}{c}{\mathbf{I}_{9\times9}} \end{array}\right]. \tag{36}$$

In this case, the $\mathbf{f}_i$ terms are given by $\mathbf{f}_i = \boldsymbol{\omega}_j^{k-1} + \dot{\boldsymbol{\omega}}_j^{k-1}T + \mathbf{w}_i^{k-1}$, $i \in \{1,2,3\}$, $j \in \{x,y,z\}$ and $\mathbf{f}_i^{k-1} = \boldsymbol{\gamma}^{k-1} + \mathbf{w}_i^{k-1}$, $i \in \{1,2,3\}$, where $\boldsymbol{\gamma}$ represents the augmented parameters, $\boldsymbol{\gamma} \in \{J_i, J_{ij}, mr_i\}$, $i,j \in \{x,y,z\}$. Considering the complexity of analitically determining the Jacobians in (36), [28] proposes the use of Complex Step Differentiation (CSD), which is exemplified in the following equations:

$$\left[\frac{\partial \mathbf{f}_{1:3}^{k-1}}{\partial \boldsymbol{\omega}_{k-1}}\right]_{3\times3} = \mathbf{I}_{3\times3} + T\left[\frac{\partial \dot{\boldsymbol{\omega}}^{k-1}}{\partial \boldsymbol{\omega}_{k-1}}\right]_{CSD}, \tag{37}$$

$$\left[\frac{\partial \dot{\boldsymbol{\omega}}^{k-1}}{\partial \boldsymbol{\omega}_{k-1}}\right]_{CSD} = \left[\begin{array}{cc} \dfrac{\mathrm{Im}\left[\dot{\boldsymbol{\omega}}^{k-1}\begin{pmatrix} \omega_x^{k-1} + h\cdot i \\ \omega_y^{k-1} \\ \omega_z^{k-1} \end{pmatrix}\right]}{h} & \cdots \end{array}\right]. \tag{38}$$

The concept of observability for nonlinear systems [41] can be used to prove that the torque generated by the momentum exchange devices is sufficient to guarantee the observability of the 9 augmented unknown parameters [33].

It must be emphasized that all the nonlinear filtering methods presented for the EKF can also be implemented with an Unscented Kalman Filter (UKF). The UKF does not linearize the model around the current estimate, but it approximates the model with a finite number of points from the state space (σ-points). The σ-points are then propagated with the nonlinear model to update the state estimate and covariance. The motivation besides the use of UKF is based on the fact that the approximation of the uncertainty through the σ-points is more accurate than the linearization performed by the EKF.

Another result is that with the UKF there is no need to calculate the Jacobians as with the EKF. An analysis of the behaviour of each filter in conditions where its development assumptions do not hold anymore is made in [28], showing comparatively the robustness of using an UKF.

4. Active Control techniques

Active control techniques are those in which the balancing process is accomplished with the use of actuators providing torque in a given control system design. The imbalance can be approximated, during a sampling interval, by the net impulse generated by antiparallel torquers placed in each rotating axis and can be compensated by feedbacking this information in an analog circuit control system [36]. This section describes recent implementations of control systems for balancing spacecraft simulators, focusing on PID, nonlinear and adaptive control.

PID control is used as a prior method for automatic balancing a spacecraft 3-axes simulator. The PID control gains can be determined with the Ziegler-Nichols method followed by a fine tuning adjustment [22]. Following the theory introduced in [42], the work in [22] also proposes an L1 adaptive control scheme and shows that it presents better automatic balancing performance than that attainable through PID control, while still having the advantage of providing more robustness when the system is subject to external disturbances or uncertainties.

4.1. Underactuated non-linear control balancing

In many scenarios, the available actuators do not allow generating a torque in the vertical direction, because the torque generated by the MMUs motion is always constrained to a plane orthogonal to the gravity vector. This means that the control problem is underactuated. To handle this situation, a two-step nonlinear control method is proposed in [7]. The control torque is obtained solely by the gravitational torque variation provided by the MMUs motion. With this method, the first step is dedicated to compensating only the x,y-components of the CM offset, while the z-component compensation is left to a further step that involves an estimation step.

This balancing technique is based on the conservation of the angular momentum, which is conserved if its derivative becomes null, *i.e.* when the external torques are null. In other words, by assuming the only external torque acting on the system is the gravitational torque and considering that the MMUs may be a source of control torque, the goal is to obtain a control torque rule for the MMUs positions such that the derivative of the angular momentum becomes null. To develop this problem, a control torque $\boldsymbol{\tau}_r$ is added to the EOM in Eq. (4), leading to

$$\dot{\boldsymbol{\Gamma}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} = \boldsymbol{\tau}_g + \boldsymbol{\tau}_r = \mathbf{r} \times M\mathbf{g} + \boldsymbol{\tau}_r \tag{39}$$

with $\boldsymbol{\tau}_r = m_p \sum_i \mathbf{r}_i \times \mathbf{g}^b$ being the designed control torque generated by the MMUs positions. The efficacy of this control strategy can be studied selecting the Lyapunov function V

$$V(\mathbf{q}, \boldsymbol{\omega}, \tilde{\boldsymbol{\Phi}}) = \frac{1}{2}\boldsymbol{\omega}^T\mathbf{J}\boldsymbol{\omega} + \frac{1}{2}\tilde{\boldsymbol{\Phi}}^{\boldsymbol{T}}\tilde{\boldsymbol{\Phi}} + \frac{1}{2}\mathbf{q}^T\mathbf{q}\,, \tag{40}$$

with $\tilde{\boldsymbol{\Theta}} = \mathbf{r} - \hat{\mathbf{r}}$ being the difference between the CM offset $\mathbf{r}$ and its estimation $\hat{\mathbf{r}}$. The derivative of this Lyapunov

function is obtained as $\dot{V} = -k_p||\boldsymbol{\omega_p}||^2$, which is negative semidefinite, proving the stability of the system in the Lyapunov sense. Also, by using the La Salle Invariance Principle [43], it may be proved that the system will converge to a state such that $\boldsymbol{\omega}_p \to 0$, *i.e.* the transverse angular velocities will diminish to zero [7]. This control strategy may be implemented by mapping the designed control torque in Eq. (39) to the MMUs positions as

$$\mathbf{r} = \frac{\mathbf{g} \times \boldsymbol{\tau}_r}{||\mathbf{g}||m_p} \ . \tag{41}$$

Finally, assuming the transverse components of the imbalance were compensated in the first step, the second step of the balancing procedure consists in estimating the vertical component of the CM offset with an UKF. In [7], a reduced state-space representation with respect to those presented in [33] for the EKF or in [28] for the UKF is proposed, whose state vector is solely defined by the angular velocities and the final imbalance components to be estimated $\mathbf{x} = [\omega_x \ \omega_y \ \omega_z \ r_z]^T$. In this case, the non-linear model used in the filter is based on Eq. (33) adapted under the assumption that $\mathbf{r} = [0 \ 0 \ r_z]^T$ and the output equation is linear, with $\mathbf{H} = [\mathbf{I}_{3\times3} \ \mathbf{0}_{3\times1}]$. In order to maintain the system observable, the platform must be placed in a tumbling motion during the estimation, avoiding the condition $[g_x, g_y] = [0, 0]$.

4.2. Fully actuated control technique

Full three-axis actuation can be provided through three Control Moment Gyros (CMG) [27]. This method assumes the presence of a control torque $\boldsymbol{\tau}$ in the EOM, in the same manner as that presented in Eq.(39), with the difference that the designed control torque $\boldsymbol{\tau}_r$ gives place to $\boldsymbol{\tau}$, the torque generated by the CMGs. This torque is given by $\boldsymbol{\tau} = -\dot{\boldsymbol{\Gamma}}_* - [\boldsymbol{\omega}\times]\boldsymbol{\Gamma}_*$.

From this model equation, starting with the proposal of a feedback control law on the actuation torque as $\boldsymbol{\tau} = (-K + [\boldsymbol{\omega}\times])\boldsymbol{\Gamma}$, K being a symmetric positive definite matrix, it is demonstrated that the closed loop equation of this control system may be described as $\dot{\boldsymbol{\Gamma}} + K\boldsymbol{\Gamma} = -m[\boldsymbol{\omega}\times]\mathbf{r}$. From this equation and studying the Lyapunov candidate function $V(\boldsymbol{\Gamma}, \mathbf{r}) = \frac{1}{2}\boldsymbol{\Gamma}^{\mathbf{T}}\boldsymbol{\Gamma} + \frac{1}{2}\mathbf{r}\Psi^{-1}\mathbf{r}$ and its derivatives, it is possible to prove the desired convergence and stability which leads the system to a state in which the total angular momentum is conserved [27]. In this case, the null vector solution is presented, showing that the proposed Lyapunov function cannot be used, since the system can eventually reach a state where the angular momentum is conserved and the testbed is not balanced (see Fig. ??). This problem is addressed by selecting a desired testbed momentum trajectory $\boldsymbol{\Gamma}_\mathbf{d}$, leading to changes in the prior Lyapunov candidate function [27], which becomes

$$V(\boldsymbol{\Gamma}, \mathbf{r}) = \frac{1}{2}(\boldsymbol{\Gamma} - \boldsymbol{\Gamma}_\mathbf{d})^T(\boldsymbol{\Gamma} - \boldsymbol{\Gamma}_\mathbf{d}) + \frac{1}{2}\mathbf{r}\Psi^{-1}\mathbf{r} \ , \tag{42}$$

providing the same required convergence and stability.

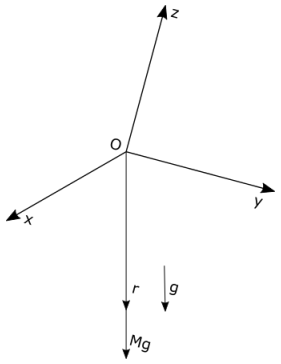


Figure 2: Null state vector solution.

5. Performance Assessment

At this point, the advantages and disadvantages of adopting one or another balancing method can be presented. Although the goals of parameter estimation methods and balancing methods are different, some common criteria for evaluation can be identified. Computational effort is one criteria that can be quantitatively assessed in both cases. Accuracy and robustness can be qualitatively assessed in both cases, too. Tables 2 and 3 present a synthetic picture of the pros and cons of the methods reviewed in this work.

The parameter estimation methods are subject to erratic estimations when submitted to certain conditions, such as estimation under high angular velocities or if the products of inertia have considerable magnitudes when compared to the diagonal terms of the inertia tensor of the simulator. Besides the foregoing conclusions, these estimation techniques may still be overcome by control theory based methods, since, as the unbalance vector has little magnitude (order of μm), it is highly subject to the hardware limitations of the platform (*e.g.* sensors' noise and precision, torque excitation measurement). In the case of control methods, the balancing procedure rely on physical conditions observable when the CM offset is compensated.

In order to evaluate if a given method has achieved the desired performance or if it performed better than another method, one must have a way of evaluating how close the CM is to the CR by its predictable effects. In [8, 15], the balancing performance is determined indirectly from the reduction of the oscillation period of the testbed. This oscillation period may be obtained by constraining the system to move around a single axis, or can be obtained from the frequency spectrum of the roll and pitch angle signals, as shown in [8]. Other two ways of assessing the balancing performance are: i) indirectly, through an energy method, by measuring the amplitude of the kinetic energy signal of the system; and ii) directly, through estimation of the gravitational torque exerted on the platform.

The energy method is based on the principle that, if the CM is placed on the CR, the gravitational potential of the system will remain constant and, consequently, its kinetic energy will also be constant, assuming there are no

Table 2: Parameter estimation methods.

Method	Pros	Cons
LSM [15]	Fast convergence.	Computational effort (*e.g.* matrix inversion); high memory requirements (batch of measurements); mismodeling errors.
KF [28]	Recursive; require less memory.	Mismodeling errors.
EKF [33]	Recursive; more accurate than linear methods.	Need to calculate Jacobians; inertia tensor simplification may lead to mismodeling errors.
UKF [7]	Recursive; do not need to calculate Jacobians; robustness.	Require more computational effort than the other KF variants due to σ-points calculation[1].

Notes: [1]: Although there are techniques to optimize the computational effort of UKF, as can be seen in [44], the algorithmic complexity of the UKF is higher than that of the EKF considering the respective original publications of these filters, for example.

Table 3: Balancing methods.

Method	Pros	Cons
Manual [10, 19, 26]	Do not require specialized hardware/software. May reduce the residual gravitational torque in a level that makes it feasible to conduct ADCS experiments.	Require expertise and may take hours to reach reasonable results (the procedure must be repeated every time the masses distribution on the platform changes).
Under actuated [7]	Faster than manual methods. Allow to adopt one of the control techniques for balancing without the need of another set of actuators, besides the movable masses.	Two-step procedure: the vertical component of the CM offset needs to be estimated through a filter and the corresponding compensation is more susceptible to the mismodeling effect of the movable mass (e.g. misalignment, inaccurate mass estimates), since dynamics is not taken into account in this step.
Fully actuated [33, 27]	Faster than manual methods. With a similar hardware setup (e.g. sensor specs), they may overcome other methods. Balancing obtained in one step.	Require a set of actuators capable of generating any three-dimensional torque vector, which necessarily means another set of actuators other than the movable masses (e.g. CMGs, reaction wheels).

energy losses (*e.g.* friction in the air bearing or gimbals). It means that any kinetic energy added to the system (*e.g.* pushing the testbed) will determine a constant motion. In practice, balancing the system will make the amplitude of the oscillation of the kinetic energy graph decrease.

Gravitational torque estimation is a direct way of quantifying how close the CM is to the CR, since it is directly proportional to the $\mathbf{r}$ offsets. The external torque $\mathbf{M}$ can be estimated from Eq. (4) using the gyroscope signals ω and its derivative $\dot{\omega}$. Furthermore, the use of Savitzky–Golay filtering can be used in order to decrease the noise of the $\dot{\omega}$ signal, which is amplified by the ω signal through differentiation [11].

6. Concluding remarks

Simulators are widely used to improve and optimize the design of physical components, algorithms, and different subsystems of a satellite. Recently, small satellites have been widely employed in many different space missions, mainly motivated by the recent technology advances that made them an affordable tool to access space, and also a promising platform to develop a new variety of space applications. The diffusion of these small platforms has given new impulse to the construction of small satellite simulators in many institutions in the last years, given the associated relatively small costs.

One aspect of satellite missions that can be easily studied in a laboratory is the attitude motion, along with the algorithms for attitude control and determination. A problem related to the development of attitude motion simulators is the calibration of the testbed. This paper is dedicated to the balancing methods and solutions for this problem. A review representative of the current state of the art of attitude motion simulators is presented, along with their balancing methods and dedicated hardware components. The balancing methods available for calibration have been described, classified, and assessed, in order to present to the reader a compact and uniform tool to guide him through the best solution for calibrating similar platforms.

Acknowledgments

This work was supported by the University of Brasília (UnB), the Federal District Research Support Foundation (FAPDF), the Coordination for the Improvement of Higher Education Personnel (CAPES), and the National Council for Scientific and Technological Development (CNPq). The authors also thank Lukas Lorenz de Andrade and Matheus Ribeiro de Brito Vieira for the updated project CAD models.

References

[1] J. L. Schwartz, M. A. Peck, C. D. Hall, Historical review of air-bearing spacecraft simulators, Journal of Guidance, Control, and Dynamics 26 (4) (2003) 513–522.

[2] R. C. da Silva, I. S. K. Ishioka, C. Cappelletti, S. Battistini, R. A. Borges, Helmholtz cage design and validation for nanosatellites hwil testing, IEEE Transactions on Aerospace and Electronic Systems 55 (6) (2019) 3050–3061.

[3] D. Jung, P. Tsiotras, A 3-dof experimental test-bed for integrated attitude dynamics and control research, in: AIAA guidance, navigation, and control conference and exhibit, 2003, p. 5331.

[4] I. Ishioka, S. Battistini, C. Cappelletti, R. Borges, Design and development of an active magnetic actuator for attitude control system of nanosatellites, Advances in the Astronautical Sciences 163 (2018) 327–342.

[5] J. De Loiola, L. Da Silva, S. Battistini, R. Borges, C. Cappelletti, Development of a hardware-in-the-loop test platform for nanosatellites adcs integrated with an ukf, Advances in the Astronautical Sciences 163 (2018) 365–373.

[6] M. Ovchinnikov, D. Roldugin, R. Borges, C. Cappelletti, S. Battistini, Modeling a satellite mockup's angular motion on an air bearing with uniaxial magnetic attitude control, Mathematical Models and Computer Simulations 12 (4) (2020) 474–481.

[7] S. Chesi, Q. Gong, V. Pellegrini, R. Cristi, M. Romano, Automatic mass balancing of a spacecraft three-axis simulator: Analysis and experimentation, Journal of Guidance, Control, and Dynamics 37 (1) (2014) 197–206.

[8] R. C. da Silva, F. C. Guimarães, J. V. L. d. Loiola, R. A. Borges, S. Battistini, C. Cappelletti, Tabletop testbed for attitude determination and control of nanosatellites, Journal of Aerospace Engineering 32 (1) (2019) 04018122.

[9] D. Modenini, A. Bahu, G. Curzi, A. Togni, A dynamic testbed for nanosatellites attitude verification, Aerospace 7 (3) (2020) 31.

[10] K. Saulnier, D. Pérez, R. Huang, D. Gallardo, G. Tilton, R. Bevilacqua, A six-degree-of-freedom hardware-in-the-loop simulator for small spacecraft, Acta Astronautica 105 (2) (2014) 444–462.

[11] A. Bahu, D. Modenini, Automatic mass balancing system for a dynamic cubesat attitude simulator: development and experimental validation, CEAS Space Journal.

[12] A. Cortiella, D. Vidal, J. Jane, E. Juan, R. Olive, A. Amezaga, J. F. Munoz, P. V. H. Carreno-Luengo, A. Camps, 3cat-2: Attitude determination and control system for a gnss-r earth observation 6u cubesat mission, European journal of remote sensing 49 (1) (2016) 759–776.

[13] R. Sutherland, I. Kolmanovsky, A. R. Girard, Attitude control of a 2u cubesat by magnetic and air drag torques, IEEE Transactions on Control Systems Technology 27 (3) (2018) 1047–1059.

[14] M. D. Shuster, A survey of attitude representations, The Journal of the Astronautical Sciences 41 (4) (1993) 439–517.

[15] J. S. Young, Development of an automatic balancing system for a small satellite attitude control simulator, Master's thesis, Utah State University (1998).

[16] Y. Liu, L. Li, Z. Fu, J. Tan, K. Li, Automatic mass balancing of a spacecraft simulator based on non-orthogonal structure, in: 2016 UKACC 11th International Conference on Control, IEEE, 2016, pp. 1–6.

[17] S. Chesi, Attitude control of nanosatellites using shifting masses, Ph.D. thesis, UC Santa Cruz (2015).

[18] J. Prado, G. Bisiacchi, L. Reyes, E. Vicente, F. Contreras, M. Mesinas, A. Juárez, Three-axis air-bearing based platform for small satellite attitude determination and control simulation, Journal of Applied Research and Technology 3 (3) (2005) 222–237.

[19] M. A. Peck, L. Miller, A. R. Cavender, M. Gonzalez, T. Hintz, An airbearing-based testbed for momentum control systems and spacecraft line of sight, Advances in the Astronautical Sciences 114 (2003) 427–446.

[20] G. Sharifi, M. Mirshams, H. Shahmohamadi Ousaloo, Mass properties identification and automatic mass balancing system for satellite attitude dynamics simulator, Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering 233 (3) (2019) 896–907.

[21] J. L. Schwartz, C. D. Hall, System identification of a spherical air-bearing spacecraft simulator, in: in Proceedings of the AAS/AIAA Space Flight Mechanics Conference, no. AAS 04-122, (Maui), Vol. 122, 2004, pp. 8–12.

[22] H. T. Xuan, A. Chemori, T. P. Anh, H. Le Xuan, T. P. Hoai, P. V. Viet, From pid to l1 adaptive control for automatic balancing of a spacecraft three-axis simulator, International Journal of Emerging Technology and Advanced Engineering 6 (1).

[23] B. Hua, L. Chen, Y. Wu, Z. Chen, A study of pid and l1 adaptive control for automatic balancing of a spacecraft three-axis simulator, International Journal of Intelligent Computing and Cybernetics.

[24] T. H. Kwan, K. M. B. Lee, J. Yan, X. Wu, An air bearing table for satellite attitude control simulation, in: 2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA), 2015, pp. 1420–1425.

[25] N. Jovanovic, J. M. Pearce, J. Praks, Design and testing of a low-cost, open source, 3-d printed air-bearing-based attitude simulator for cubesat satellites, Journal of Small Satellites (JoSS) 8 (2) (2019) 859.

[26] M. Romano, B. N. Agrawal, Acquisition, tracking and pointing control of the bifocal relay mirror spacecraft, Acta Astronautica 53 (4) (2003) 509–519.

[27] J. J. Kim, B. N. Agrawal, Automatic mass balancing of air-bearing-based three-axis rotational spacecraft simulator, Journal of Guidance, Control, and Dynamics 32 (3) (2009) 1005–1017.

[28] R. C. da Silva, Filtering and adaptive control for balancing a nanosatellite testbed, Master's thesis, University of Brasília (2018).

[29] J. Prado, G. Bisiacchi, Dynamic balancing for a satellite attitude control simulator, Instrumentation and Development. Journal of the Mexican Society of Instrumentation. SOMI 4 (5) (2000) 76–81.

[30] B. Kim, E. Velenis, P. Kriengsiri, P. Tsiotras, A spacecraft simulator for research and education, in: Proceedings of the AIAA/AAS astrodynamics specialists conference, no. AAS 01-367, 2001, pp. 897–914.

[31] M. Peck, Estimation of inertia parameters for gyrostats subject to gravity-gradient torques, Advances in the Astronautical Sciences.

[32] S. Tanygin, T. Williams, Mass property estimation using coasting maneuvers, Journal of Guidance, Control, and Dynamics 20 (4) (1997) 625–632.

[33] Z. Xu, N. Qi, Y. Chen, Parameter estimation of a three-axis spacecraft simulator using recursive least-squares approach with tracking differentiator and extended kalman filter, Acta Astronautica 117 (2015) 254–262.

[34] S. Wright, Parameter estimation of a spacecraft simulator using parameter-adaptive control, MS Project, Aerospace and Ocean Engineering Department, Virginia Polytechnic Institute and State University, Blacksburg, VA.

[35] D. Kim, S. Yang, S. Lee, Rigid body inertia estimation using extended kalman and savitzky-golay filters, Mathematical Problems in Engineering 2016.

[36] N. M. Hatcher, R. N. Young, An automatic balancing system for use on frictionlessly supported attitude-controlled test platforms, Tech. rep., NASA Langley Research Center (1968).

[37] D. Small, F. Zajac, A linearized analysis and design of an automatic balancing system for the three-axis air bearing table, Tech. rep., NASA Goddard Space Flight Center (1963).

[38] T. A. Olsen, Design of an adaptive balancing scheme for the small satellite attitude control simulator (ssacs)., Master's thesis, Utah State University (1996).

[39] S. Haykin, Adaptive filter theory, Prentice-Hall, Inc., 1996.

[40] D. Simon, Optimal state estimation: Kalman, H infinity, and

nonlinear approaches, John Wiley & Sons, 2006.
[41] W. Kang, J.-P. Barbot, Discussions on observability and invertibility, IFAC Proceedings Volumes 40 (12) (2007) 426–431.
[42] N. Hovakimyan, C. Cao, L1 Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation, SIAM, 2010.
[43] R. I. Leine, N. Van de Wouw, Stability and convergence of mechanical systems with unilateral constraints, Vol. 36, Springer Science & Business Media, 2007.
[44] M. Raitoharju, R. Piché, On computational complexity reduction methods for kalman filter extensions, IEEE Aerospace and Electronic Systems Magazine 34 (10) (2019) 2–19.