Satellite Dynamics Toolbox - Principle, User Guide and tutorials

Daniel Alazard, Christelle Cumer

June 2014

EMPTY PAGE

Contents

Nomenclature

$\mathcal{R}_b = (O, \vec{x}_b, \vec{y}_b, \vec{z}_b)$	:	Main body (hub $\mathcal{B}$) reference frame. O is a reference point on the main body. $\vec{x}_b$, $\vec{y}_b$, $\vec{z}_b$ are unit vectors.
$\mathcal{R}_a = (P, \vec{x}_a, \vec{y}_a, \vec{z}_a)$	:	Appendage $\mathcal{A}$ reference frame. P is the appendage's anchoring point on the hub. $\vec{x}_a$, $\vec{y}_a$, $\vec{z}_a$ are unit vectors.
B	:	Hub's ($\mathcal{B}$) centre of mass.
A	:	Appendage's ($\mathcal{A}$) centre of mass.
G	:	Overall assembly's centre of mass $\mathcal{B} + \mathcal{A}$.
T_{ba}	:	Direction cosine matrix of the rotation from frame $\mathcal{R}_b$ to frame $\mathcal{R}_a$, that contains the coordinates of vectors $\vec{x}_a$, $\vec{y}_a$, $\vec{z}_a$ expressed in frame $\mathcal{R}_b$.
$\overrightarrow{a}_B$	:	Inertial acceleration (vector) of body $\mathcal{B}$ at point B.
$\overrightarrow{a}_P$	:	Inertial acceleration (vector) of body $\mathcal{B}$ at point P.
$\overrightarrow{\omega}$	:	Angular speed (vector) of $\mathcal{R}_b$ with respect to the inertial frame.
$\overrightarrow{F}_{ext}$	:	External forces (vector) applied to $\mathcal{B}$.
$\overrightarrow{T}_{ext,B}$	:	External torques (vector) applied to $\mathcal{B}$ at point B.
$\overrightarrow{F}_{\mathcal{B}/\mathcal{A}}$	:	Force (vector) applied by $\mathcal{B}$ on $\mathcal{A}$.
$\overrightarrow{T}_{\mathcal{B}/\mathcal{A},P}$	:	Torque (vector) applied by $\mathcal{B}$ on $\mathcal{A}$ at point P.
$D_B^{\mathcal{B}}$	:	Hub's static dynamics model expressed at point B.
$m^{\mathcal{B}}$	:	Hub's $\mathcal{B}$ mass.
$\mathbb{I}_B^{\mathcal{B}}$	:	Inertia tensor 3×3 of $\mathcal{B}$ at point B.
$M_P^{\mathcal{A}}(\mathbf{s})$	:	Appendage's dynamics model at point P.
$m^{\mathcal{A}}$	:	Appendage's mass.
$\mathbb{I}_A^{\mathcal{A}}$	:	Inertia tensor 3×3 of $\mathcal{A}$ at point A.
τ_{PB}	:	Kinematic model between points P and B:
	:	$\tau_{PB} = \begin{bmatrix} I_3 & (^*\overrightarrow{PB}) \\ 0_{3\times3} & I_3 \end{bmatrix}$.

$(^*\overrightarrow{PB})$: Skew symmetric matrix associated with vector $\overrightarrow{PB}$: if $\overrightarrow{PB} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\mathcal{R}_i}$

then $\left[(^*\overrightarrow{PB})\right]_{\mathcal{R}_i} = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}_{\mathcal{R}_i}$.

N_a : Number of appendages.

If there is a flexible appendage, we add the following definitions:

N : Number of flexible modes.
η : Modal coordinates' vector.
ω_i : i^{th} flexible mode's angular frequency.
ξ_i : i^{th} flexible mode's damping ratio.
$l_{i,P}$: i^{th} flexible mode's vector of modal participations (1×6), expressed at point P.
L_P : Matrix $N \times 6$ of the modal participation factors expressed at point P: $(L_P = [l_{1,P}^T,\ l_{2,P}^T, \cdots,\ l_{N,P}^T]^T)$.

If there is an embedded angular momentum, we add the following definitions:

Ω : Angular speed (rad/s) of the spinning top.
$\vec{z}_w$: Unit vector along the embedded angular momentum.
I_w : Spinning top's principal inertia.
I_r : Spinning top's radial inertia.

If there is a revolute joint between the appendage and the hub, we add the following definitions:

$\vec{r}_a$: Unit vector along the revolute joint axis: $\vec{r}_a = [x_{r_a}\ y_{r_a}\ z_{r_a}]^T_{\mathcal{R}_a}$.
$\ddot{\theta}$: Revolute joint's angular acceleration.
C_m : Revolute joint's torque.

General definitions:

$[X]_{\mathcal{R}_i}$: X (model, vector or tensor) expressed in frame $\mathcal{R}_i$.
$\frac{d\vec{X}}{dt}|_{\mathcal{R}_i} = 0$: Derivative of vector $\overrightarrow{X}$ with respect to frame $\mathcal{R}_i$.
$\vec{u} \wedge \vec{v}$: Cross product of vector $\vec{u}$ with vector $\vec{v}$ $(\vec{u} \wedge \vec{v} = (^*\vec{u})\vec{v})$
$\vec{u}.\vec{v}$: Scalar product of vector $\vec{u}$ with vector $\vec{v}$ $(\vec{u}.\vec{v} = [\vec{u}]^T_{\mathcal{R}_i}\,[\vec{v}]_{\mathcal{R}_i},\ \forall \mathcal{R}_i)$
$\mathbf{s}$: LAPLACE's variable.
I_n : Identity matrix $n \times n$.
$0_{n\times m}$: Zero matrix $n \times m$.
A^T : Transpose of A.
$\text{diag}(\omega_i)$: Diagonal matrix $N \times N$: $\text{diag}(\omega_i)(i,i) = \omega_i$, $i = 1, \cdots, N$.
$P(\mathbf{s})(i:j, l:m)$: Sub-system of $P(\mathbf{s})$ from inputs l to m to outputs i to j

Introduction

This document presents the principles and the MATLAB® implementation of the Satellite Dynamics Toolbox (SDT version 1.3) available for download following this link:
http://personnel.isae.fr/daniel-alazard/matlab-packages/satellite-dynamics-toolbox.html.
This toolbox allows the user to compute the **linear** dynamics model of a satellite fitted with one or more flexible appendage (solar generators, antennas, ...) possibly taking into account the joints and actuators (solar generator driving mechanism, Control Moment Gyro (CMG), external manipulator, ...). More generally, it can model an open kinematic chain of bodies (sub-structures) in a space environment (gravity efforts are not taken into account). Such a chain or structure can be described by a tree (see Figure 1) where:

- each node $\mathcal{A}_i$ corresponds to a body (or link or substructure),
- each edge corresponds to a joint between two bodies. 2 types of joint are considered:
  - clamped joint,
  - revolute joint (around a given axis).

Classical tree terminology will be used. In the example of Figure 1: $\mathcal{A}_0$ is the root, $\mathcal{A}_3$ to $\mathcal{A}_7$ are the leaves, $\mathcal{A}_2$ is the parent of $\mathcal{A}_5$ and the child of $\mathcal{A}_0$.
Each body can be either flexible or not (with the restriction that the flexible bodies must be the last elements or the leaves of the chain). The **linear** dynamics model is the 6 by 6 transfer between the external forces and torques applied on the main body structure at a reference point and the translational and rotational accelerations (at the same reference point). When the structure includes joints (in this version of the toolbox, only revolute joints between the sub-structures are considered), the model is augmented with the transfer between internal torque and acceleration for each joint.
Chapter 1 is a reminder on NEWTON-EULER equations applied to a single rigid body.

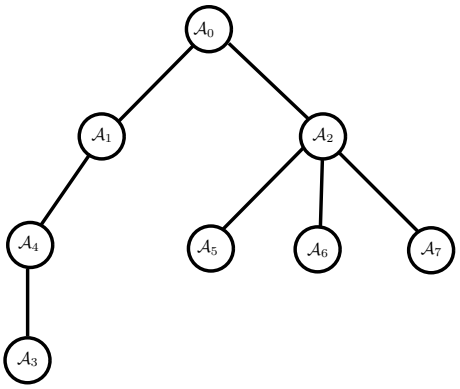

Figure 1: Example of a tree (open-chain) structure.

Chapter 2 explains the modelling principles used in the Satellite Dynamics Toolbox. Chapter 3 presents the MATLAB® implementation of the toolbox, and more particularly:

- the setup of the data files describing the structure to be modeled,
- structure examples:
  - `Spacecraft1.m` is a data file describing a spacecraft composed of a main body and two cantilevered flexible appendages (see section 3.4),
  - `Spacecraft2.m` is the same as `Spacecraft1.m` but the first appendage is connected to the main body through a revolute joint and tilted with a 10 degrees angle (see section 3.5),
  - `Spacecraft5.m` is the same as `Spacecraft1.m` but includes 3 additionnal appendages corresponding to angular momentums along to X, Y and Z axes taken into account to represent spinned RWAs (see section 3.6),
  - `FOUR_CMGs.m` is a data file describing a platform fitted with four identical CMGs (see section 3.7). The CMG is described by the data file `dataCMG.m`. This data correspond roughly to the experimental CMG platform TETRAGYRE developed by ONERA/DCSD and presented at: http://www.onera.fr/dcsd/gyrodynes/,
  - `SpaceRoboticArm.m` is a data file describing a platform fitted with a 6 d.o.f. rigid space robotic manipulator (see section 3.8): the 6 links of the manipulator are described in the files `Segment1.m`, `Segment2.m`, ..., `Segment6.m`.

  All these files and a demo script file (`demoSTD.m`) are included in the SDT package.

Remark: this new version of the toolbox can also manage **uncertain parameters** to perform sensitivity analyses. In this case, the outputs of the toolbox main functions are uncertain elements, matrices or models, compatible with the MATLAB® Robust Control Toolbox (see file `Spacecraft1u.m` as an example, section 3.9)).
The development of this version of the toolbox was done with MATLAB® R2013a.

EMPTY PAGE

Chapter 1

Rigid body dynamics based on Newton-Euler equations

This chapter is a reminder on rigid body dynamics based on NEWTON-EULER equations (see also [6]). The interest of NEWTON-EULER equations is to consider in the same model the 6 degrees-of-fredom (3 translations and 3 rotations) of the rigid body. This chapter will also present the **linearity assmuptions** which will be adopted in the following chapters.

1.1 Newton-Euler equations at the center of mass

Let us consider a body $\mathcal{B}$ with a center of mass B (see Figure 1.1).

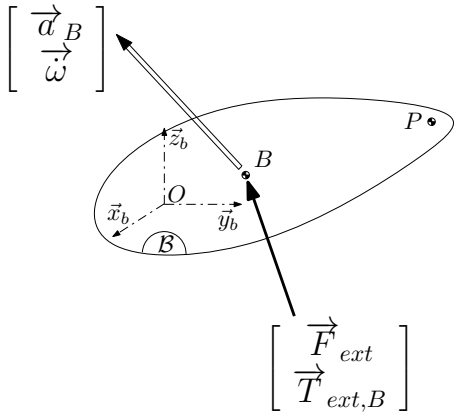

Figure 1.1: A rigid body.

The NEWTON-EULER equations reads:

$$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix} = \underbrace{\begin{bmatrix} m^{\mathcal{B}} I_3 & 0_{3\times 3} \\ 0_{3\times 3} & \mathbb{I}_B^{\mathcal{B}} \end{bmatrix}}_{D_B^{\mathcal{B}}} \begin{bmatrix} \overrightarrow{a}_B \\ \overrightarrow{\dot{\omega}} \end{bmatrix} + \begin{bmatrix} 0_{3\times 1} \\ \overrightarrow{\omega} \wedge \mathbb{I}_B^{\mathcal{B}} \overrightarrow{\omega} \end{bmatrix} \tag{1.1}$$

where:

- $\overrightarrow{F}_{ext}$: total force acting on the body,
- $\overrightarrow{T}_{ext,B}$: total torque acting about the centre of mass,
- $m^{\mathcal{B}}$: mass of the body,
- $\mathbb{I}_B^{\mathcal{B}}$: Inertia tensor of the body about the centre of mass,
- $\overrightarrow{a}_B$: acceleration of the centre of mass,
- $\overrightarrow{\omega}$: angular velocity of the body.

Eq. (1.1) is intrinsic and can be projected in any frame. Using the skew matrix $({}^*\overrightarrow{\omega})$ associated with vector $\overrightarrow{\omega}$ (see nomenclature), Eq. (1.1) can be written:

$$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix} = \underbrace{\begin{bmatrix} m^{\mathcal{B}} I_3 & 0_{3\times 3} \\ 0_{3\times 3} & \mathbb{I}_B^{\mathcal{B}} \end{bmatrix}}_{D_B^{\mathcal{B}}} \begin{bmatrix} \overrightarrow{a}_B \\ \overrightarrow{\dot{\omega}} \end{bmatrix} + \begin{bmatrix} 0_{3\times 1} \\ ({}^*\overrightarrow{\omega})\, \mathbb{I}_B^{\mathcal{B}} \overrightarrow{\omega} \end{bmatrix} \tag{1.2}$$

1.2 Newton-Euler equations at any reference point

Let us consider another point P on the body $\mathcal{B}$. Since the body is rigid (**assumption [H1]**),the velocity of P can be expressed as a function of the velocity of B and the angular velocity $\overrightarrow{\omega}$:

$$\overrightarrow{v}_P = \overrightarrow{v}_B + \overrightarrow{\omega} \wedge \overrightarrow{BP} = \overrightarrow{v}_B + \overrightarrow{PB} \wedge \overrightarrow{\omega} \ . \tag{1.3}$$

Using the skew matrix $({}^*\overrightarrow{PB})$ associated with vector $\overrightarrow{BP}$ (see nomenclature), one can write:

$$\overrightarrow{v}_P = \overrightarrow{v}_B + ({}^*\overrightarrow{PB})\overrightarrow{\omega} \ . \tag{1.4}$$

Considering the dual velovity vectors (6 components) at point B and P, one can write:

$$\begin{bmatrix} \overrightarrow{v}_P \\ \overrightarrow{\omega} \end{bmatrix} = \underbrace{\begin{bmatrix} I_3 & ({}^*\overrightarrow{PB}) \\ 0_{3\times 3} & I_3 \end{bmatrix}}_{\tau_{PB}} \begin{bmatrix} \overrightarrow{v}_B \\ \overrightarrow{\omega} \end{bmatrix} \ . \tag{1.5}$$

τ_{PB} is called the kinematic model (or jacobian) between points P and B.
Property: $\tau_{PB}^{-1} = \tau_{BP}$.
Considering the dual force vectors (6 components) at point B and P, one can also write:

$$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix} = \underbrace{\begin{bmatrix} I_3 & 0_{3\times 3} \\ -(^*\overrightarrow{PB}) & I_3 \end{bmatrix}}_{\tau_{PB}^T} \begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,P} \end{bmatrix} . \tag{1.6}$$

Indeed: $\overrightarrow{T}_{ext,B} = \overrightarrow{T}_{ext,P} + \underbrace{\overrightarrow{BP} \wedge \overrightarrow{F}_{ext}}_{\text{lever arm} \times \text{force}}$.
The time-derivation of (1.3) in the inertial frame gives:

$$\overrightarrow{a}_P = \overrightarrow{a}_B + \overrightarrow{PB} \wedge \dot{\overrightarrow{\omega}} + \underbrace{\left.\frac{d\overrightarrow{PB}}{dt}\right|_{\mathcal{B}}}_{=0\ ([H1])} \wedge \overrightarrow{\omega} + \overrightarrow{\omega} \wedge (\overrightarrow{PB} \wedge \overrightarrow{\omega}) , \tag{1.7}$$

or equivalently:

$$\begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix} = \underbrace{\begin{bmatrix} I_3 & (^*\overrightarrow{PB}) \\ 0_{3\times 3} & I_3 \end{bmatrix}}_{\tau_{PB}} \begin{bmatrix} \overrightarrow{a}_B \\ \dot{\overrightarrow{\omega}} \end{bmatrix} + \begin{bmatrix} (^*\overrightarrow{\omega})(^*\overrightarrow{PB})\overrightarrow{\omega} \\ 0_{3\times 1} \end{bmatrix} . \tag{1.8}$$

Using (1.2), (1.6) and (1.8), one can derive the Newton-Euler equations at point P:

$$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,P} \end{bmatrix} = \underbrace{\begin{bmatrix} m^{\mathcal{B}} I_3 & m^{\mathcal{B}}\,(^*\overrightarrow{BP}) \\ -m^{\mathcal{B}}\,(^*\overrightarrow{BP}) & \mathbb{I}_B^{\mathcal{B}} - m^{\mathcal{B}}\,(^*\overrightarrow{BP})^2 \end{bmatrix}}_{\tau_{BP}^T D_B^{\mathcal{B}} \tau_{BP}} \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix} + \begin{bmatrix} m^{\mathcal{B}}(^*\overrightarrow{\omega})(^*\overrightarrow{BP})\overrightarrow{\omega} \\ (^*\overrightarrow{\omega})\left(\mathbb{I}_B^{\mathcal{B}} - m^{\mathcal{B}}\,(^*\overrightarrow{BP})^2\right)\overrightarrow{\omega} \end{bmatrix} \tag{1.9}$$

1.3 Linearized Newton-Euler equations

Assuming that $\overrightarrow{\omega}$ is small (**linearity assumption [H2]**), quadratic terms in $(^*\overrightarrow{\omega})X_{3\times 3}\overrightarrow{\omega}$ can be neglected. Then equations (1.2) and (1.9) becomes:

- $$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix} = \underbrace{\begin{bmatrix} m^{\mathcal{B}} I_3 & 0_{3\times 3} \\ 0_{3\times 3} & \mathbb{I}_B^{\mathcal{B}} \end{bmatrix}}_{D_B^{\mathcal{B}}} \begin{bmatrix} \overrightarrow{a}_B \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$$

  $D_B^{\mathcal{B}}$ is the direct dynamics model of the body $\mathcal{B}$ about the centre of mass B.

- $$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,P} \end{bmatrix} = \underbrace{\begin{bmatrix} m^{\mathcal{B}} I_3 & m^{\mathcal{B}} (^*\overrightarrow{BP}) \\ -m^{\mathcal{B}} (^*\overrightarrow{BP}) & \mathbb{I}_B^{\mathcal{B}} - m^{\mathcal{B}} (^*\overrightarrow{BP})^2 \end{bmatrix}}_{D_P^{\mathcal{B}} = \tau_{BP}^T D_B^{\mathcal{B}} \tau_{BP}} \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$$

$$\boxed{D_P^{\mathcal{B}} = \tau_{BP}^T \, D_B^{\mathcal{B}} \, \tau_{BP}} \tag{1.10}$$

is the direct dynamics model of the body $\mathcal{B}$ about the point P.

Equations (1.10) is very convenient to transport the direct dynamics model from one point to another: considering a third point R, one can also write:

$$D_R^{\mathcal{B}} = \tau_{BR}^T \, D_B^{\mathcal{B}} \, \tau_{BR} = \tau_{BR}^T \tau_{PB}^T \, D_P^{\mathcal{B}} \, \tau_{PB} \, \tau_{BR} \tag{1.11}$$

$$D_R^{\mathcal{B}} = \tau_{PR}^T \, D_P^{\mathcal{B}} \, \tau_{PR} \, . \tag{1.12}$$

Without loss of generality, one can write the linear dynamics model of a mechanical system at its center of mass B and then transport it to any other point R (where the external force is applied for instance).
The development of (1.10) is:

$$D_P^{\mathcal{B}} = \tau_{BP}^T \, D_B^{\mathcal{B}} \, \tau_{BP} = \begin{bmatrix} m^{\mathcal{B}} I_3 & m^{\mathcal{B}} (^*\overrightarrow{BP}) \\ -m^{\mathcal{B}} (^*\overrightarrow{BP}) & \underbrace{\mathbb{I}_B^{\mathcal{B}} - m^{\mathcal{B}} (^*\overrightarrow{BP})^2}_{\mathbb{I}_P^{\mathcal{B}}} \end{bmatrix}$$

One can recognize in the bottom-right-hand term, the Huygens theorem between the inertia tensor $\mathbb{I}_B^{\mathcal{B}}$ of $\mathcal{B}$ at point B and the inertia tensor $\mathbb{I}_P^{\mathcal{B}}$ of $\mathcal{B}$ at point P:

$$\mathbb{I}_P^{\mathcal{B}} = \mathbb{I}_B^{\mathcal{B}} - m^{\mathcal{B}} (^*\overrightarrow{BP})^2 \tag{1.13}$$

or in projection in the frame $\mathcal{R}_b = (0, \vec{x}_b, \vec{y}_b, \vec{z}_b)$ attached to the body $\mathcal{B}$, with $\overrightarrow{BP} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\mathcal{R}_b}$:

$$\left[\mathbb{I}_P^{\mathcal{B}}\right]_{\mathcal{R}_b} = \left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b} + m^{\mathcal{B}} \begin{bmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{bmatrix}_{\mathcal{R}_b} \tag{1.14}$$

Note that the equation (1.12) works on the 6×6 direct dynamics model and is valid for any points (P and R) on the body $\mathcal{B}$ while Huygens theorem (equation (1.13)) works on the 3×3 tensor of inertia and is valid only if B coincides with the centre of mass of the body $\mathcal{B}$. In the general case:

$$\mathbb{I}_R^{\mathcal{B}} \neq \mathbb{I}_P^{\mathcal{B}} - m^{\mathcal{B}} (^*\overrightarrow{PR})^2 \, .$$

Nevertheless Huygens theorem (equation (1.13)) can be used to find $\mathbb{I}_B^{\mathcal{B}}$ from $\mathbb{I}_O^{\mathcal{B}}$, in the case where the data of the tensor of inertia are provided w.r.t. the body reference frame $\mathcal{R}_b$:

$$\mathbb{I}_B^{\mathcal{B}} = \mathbb{I}_O^{\mathcal{B}} + m^{\mathcal{B}} ({}^*\overrightarrow{OB})^2 \ . \tag{1.15}$$

Indeed, in the Matlab® implementation of the SDT (detailled in chapter 3), the tensor of inertia of each body must be defined w.r.t. to its centre of mass by the user. In some applications, data are provided at a reference point O different from the centre of mass B, then equation (1.15) can then be used and can be easily implemented in the SDT (see section 3.2.2 as a tutorial).

EMPTY PAGE

Chapter 2

Satellite Dynamics Toolbox: principles

We consider a satellite as a main body $\mathcal{B}$ (the hub) and one appendage $\mathcal{A}$ cantilevered to the hub at point P, as shown on Figure 2.1. The appendage is considered as a dynamic sub-structure either because of its flexibility or because of an embedded angular momentum (reaction wheels or CMG). The objective is two-fold: first, computing the **direct** dynamics model $M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$, i.e. the 6 by 6 transfer between the dual vector of the translational and rotational accelerations $\begin{bmatrix} \overrightarrow{a}_B \\ \overrightarrow{\dot{\omega}} \end{bmatrix}$ of the hub seen from the hub centre of mass B and the dual vector of the external forces and torques $\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix}$ applied to the hub at point B :

$$\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,B} \end{bmatrix} = M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) \begin{bmatrix} \overrightarrow{a}_B \\ \overrightarrow{\dot{\omega}} \end{bmatrix} .$$

Second, computing the **inverse** dynamics model $[M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})]^{-1}$.
Considering the result of the previous chapter (equation (1.10)), one can then compute the direct and inverse dynamics models about any arbitrary reference point R, that is: the models between dual vectors of accelerations $\begin{bmatrix} \overrightarrow{a}_R \\ \overrightarrow{\dot{\omega}} \end{bmatrix}$ and forces $\begin{bmatrix} \overrightarrow{F}_{ext} \\ \overrightarrow{T}_{ext,R} \end{bmatrix}$ seen from the reference point R:

$$M_R^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) = \tau_{BR}^T \, M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) \, \tau_{BR} \tag{2.1}$$

$$\left[M_R^{\mathcal{A}+\mathcal{B}}(\mathbf{s})\right]^{-1} = \tau_{RB} \, \left[M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})\right]^{-1} \, \tau_{RB}^T \, . \tag{2.2}$$

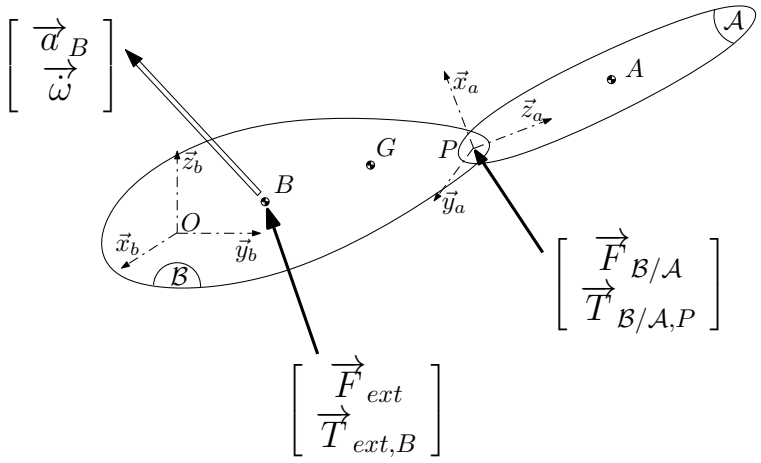

Figure 2.1: The system hub $\mathcal{B}$ + appendage $\mathcal{A}$.

2.1 Assumptions

[**H1**] The hub is rigid: $\frac{d\vec{BP}}{dt}|_{\mathcal{R}_b} = 0$.

[**H2**] Non-linear terms (in $\vec{\omega} \wedge X_{3\times3}\vec{\omega}$) of second or higher order are disregarded.

[**H3**] The only external force (resp. torque) applied to the appendage $\mathcal{A}$ is the force $\vec{F}_{\mathcal{B}/\mathcal{A}}$ (resp. torque $\vec{T}_{\mathcal{B}/\mathcal{A},P}$) applied by the hub $\mathcal{B}$ at the appendage's anchorage point P.

2.2 Direct dynamics model

The direct dynamics model of the assembly $M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$ at the hub B centre of mass is the sum ([2], [4]):

- of the hub's direct dynamics model at point B: $D_B^{\mathcal{B}}$:
  $D_B^{\mathcal{B}} = \begin{bmatrix} m^{\mathcal{B}} I_3 & 0_{3\times3} \\ 0_{3\times3} & \mathbb{I}_B^{\mathcal{B}} \end{bmatrix}$ which can be projected in frame $\mathcal{R}_b$ as follows:
  $\left[D_B^{\mathcal{B}}\right]_{\mathcal{R}_b} = \begin{bmatrix} m^{\mathcal{B}} I_3 & 0_{3\times3} \\ 0_{3\times3} & \left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b} \end{bmatrix}_{\mathcal{R}_b}$,
- and of the appendage's direct dynamics model at point P: $M_P^{\mathcal{A}}(\mathbf{s})$ moved to point B thanks to the kinematic model τ_{PB} (see nomenclature) and the transport equation (1.10), that leads to:

$$M_B^{\mathcal{A}}(\mathbf{s}) = \tau_{PB}^T M_P^{\mathcal{A}}(\mathbf{s}) \tau_{PB} \ . \tag{2.3}$$

When summing, we can write that:

$$M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) = D_B^{\mathcal{B}} + \tau_{PB}^T M_P^{\mathcal{A}}(\mathbf{s}) \tau_{PB} \ . \tag{2.4}$$

In projection in the hub's frame $\mathcal{R}_b$, the assembly's direct dynamics model is the following:

$$\left[M_B^{\mathcal{A}+\mathcal{B}}\right]_{\mathcal{R}_b}(\mathbf{s}) = \left[D_B^{\mathcal{B}}\right]_{\mathcal{R}_b} + \left[\tau_{PB}^T\right]_{\mathcal{R}_b} \begin{bmatrix} T_{ba} & 0_{3\times3} \\ 0_{3\times3} & T_{ba} \end{bmatrix} \left[M_P^{\mathcal{A}}(\mathbf{s})\right]_{\mathcal{R}_a} \begin{bmatrix} T_{ba} & 0_{3\times3} \\ 0_{3\times3} & T_{ba} \end{bmatrix}^T \left[\tau_{PB}\right]_{\mathcal{R}_b}$$

where T_{ba} is the direction cosine matrix of the rotation from frame $\mathcal{R}_b$ to frame $\mathcal{R}_a$ (see nomenclature). Figure 2.2 represents this model's block diagram and matching physical signals.

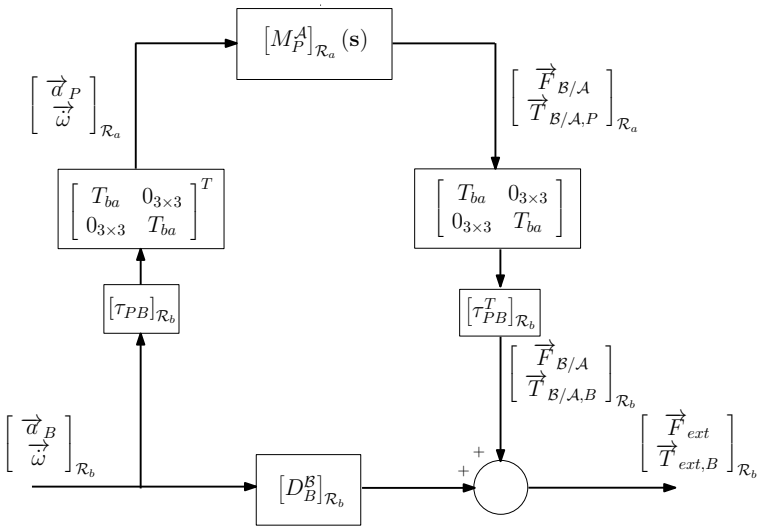

Figure 2.2: Direct dynamics model block diagram $M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$ expressed in frame $\mathcal{R}_b$.

The following operation allows for the transport of the direct dynamics model to point G, the global assembly's centre of mass:

$$M_G^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) = \tau_{BG}^T M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) \tau_{BG} \ .$$

2.3 Appendage dynamics model $M_P^{\mathcal{A}}(\mathbf{s})$

There are 3 different cases to consider in order to write the appendage's dynamics model $M_P^{\mathcal{A}}(\mathbf{s})$, at anchoring point P between $\mathcal{A}$ and $\mathcal{B}$ (see [2]).

2.3.1 Case of a rigid appendage

$$M_P^{\mathcal{A}}(\mathbf{s}) = D_P^{\mathcal{A}} = \tau_{AP}^T \begin{bmatrix} m^{\mathcal{A}} I_3 & 0_{3\times 3} \\ 0_{3\times 3} & \mathbb{I}_A^{\mathcal{A}} \end{bmatrix} \tau_{AP} \ . \tag{2.5}$$

This is a static model.

2.3.2 Case of a flexible appendage

$$M_P^{\mathcal{A}}(\mathbf{s}) = D_P^{\mathcal{A}} - \sum_{i=1}^{N} l_{i,P}^T l_{i,P} \frac{\mathbf{s}^2}{\mathbf{s}^2 + 2\xi_i\omega_i\mathbf{s} + \omega_i^2} \ , \tag{2.6}$$

in which $D_P^{\mathcal{A}}$ comes from (2.5) and the various parameters ω_i, ξ_i et $l_{i,P}$ (see nomenclature) can be provided by finite element software used to model the appendage (see [2] or [3] for more explanations). Other representations can be used for $M_P^{\mathcal{A}}(\mathbf{s})$:

- the hybrid-cantilevered model:

$$\begin{bmatrix} \overrightarrow{F}_{\mathcal{B}/\mathcal{A}} \\ \overrightarrow{T}_{\mathcal{B}/\mathcal{A},P} \end{bmatrix} = D_P^A \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix} + L_P^T \ddot{\eta}$$

$$\ddot{\eta} + diag(2\xi_i\omega_i)\dot{\eta} + diag(\omega_i^2)\eta = -L_P \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$$

- the second order generic model:

$$\begin{bmatrix} D_P^A & L_P^T \\ L_P & I_N \end{bmatrix} \begin{bmatrix} \ddot{q}_r \\ \ddot{\eta} \end{bmatrix} + \begin{bmatrix} 0_{6\times 6} & 0_{6\times N} \\ 0_{N\times 6} & \mathrm{diag}(2\xi_i\omega_i) \end{bmatrix} \begin{bmatrix} \dot{q}_r \\ \dot{\eta} \end{bmatrix} + \begin{bmatrix} 0_{6\times 6} & 0_{6\times N} \\ 0_{N\times 6} & \mathrm{diag}(\omega_i^2) \end{bmatrix} \begin{bmatrix} q_r \\ \eta \end{bmatrix} = \begin{bmatrix} F_{ext} \\ 0_{N\times 1} \end{bmatrix}$$

where $\ddot{q}_r = \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$ and $F_{ext} = \begin{bmatrix} \overrightarrow{F}_{\mathcal{B}/\mathcal{A}} \\ \overrightarrow{T}_{\mathcal{B}/\mathcal{A},P} \end{bmatrix}$.

- the state-space representation with a feedforward matrix $D_{P_0}^A$, which is the residual mass of the appendage $\mathcal{A}$ rigidly cantilevered to the hub $\mathcal{B}$ at point P:

$$\begin{bmatrix} \dot{\eta} \\ \ddot{\eta} \end{bmatrix} = \begin{bmatrix} 0_{N\times N} & I_N \\ -\mathrm{diag}(\omega_i^2) & -\mathrm{diag}(2\xi_i\omega_i) \end{bmatrix} \begin{bmatrix} \eta \\ \dot{\eta} \end{bmatrix} + \begin{bmatrix} 0_{N\times 6} \\ -L_P \end{bmatrix} \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$$

$$\begin{bmatrix} \overrightarrow{F}_{\mathcal{B}/\mathcal{A}} \\ \overrightarrow{T}_{\mathcal{B}/\mathcal{A},P} \end{bmatrix} = \begin{bmatrix} -L_P^T \mathrm{diag}(\omega_i^2) & -L_P^T \mathrm{diag}(2\xi_i\omega_i) \end{bmatrix} \begin{bmatrix} \eta \\ \dot{\eta} \end{bmatrix} + \underbrace{(D_P^A - L_P^T L_P)}_{D_{P_0}^A} \begin{bmatrix} \overrightarrow{a}_P \\ \dot{\overrightarrow{\omega}} \end{bmatrix}$$

- or the block diagram on Figure 2.3. This diagram easily allows for the introduction of parametric uncertainties on ω_i, ξ_i, L_p and $D_{P_0}^A$ because they appear with a minimum number of occurences.

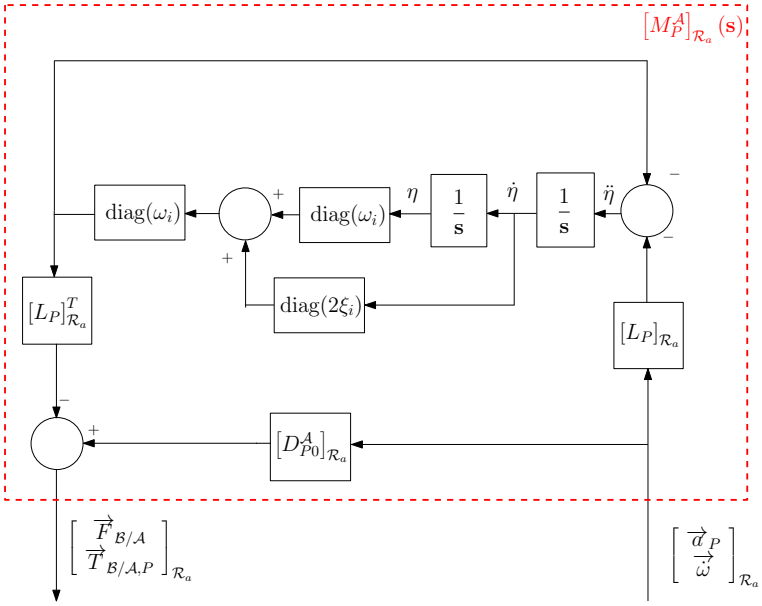

Figure 2.3: Direct dynamics model block diagram of the appendage $M_P^{\mathcal{A}}(\mathbf{s})$ expressed in frame $\mathcal{R}_a$.

2.3.3 Case of an embedded angular momentum

$$M_P^{\mathcal{A}}(\mathbf{s}) = D_P^{\mathcal{A}} + \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -\frac{1}{\mathbf{s}} I_w \Omega \, ({}^*\vec{z}_w) \end{bmatrix}, \tag{2.7}$$

where $D_P^{\mathcal{A}}$ comes from (2.5), where I_w $(Kg\,m^2)$ is the appendage's inertia about its spin axis (spinning top) defined by unit vector $\vec{z}_w$ and where Ω is the spinning top's speed (rad/s).

Complementary assumptions

When the appendage is an embedded angular momentum, we will further suppose that:

[**H4**] Ω and $\vec{z}_w$ (the angular speed and the spin axis of the spinning top) are constants in frame $\mathcal{R}_a$,

[**H5**] the spinning top is balanced,

[**H6**] (non-restrictive) the spin axis is along the z axis of the appendage body frame $\mathcal{R}_a$ (i.e. $\vec{z}_w = \vec{z}_a$).

Under assumptions [H5] and [H6], one can write:

$$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a} = \begin{bmatrix} I_r & 0 & 0 \\ 0 & I_r & 0 \\ 0 & 0 & I_w \end{bmatrix}_{\mathcal{R}_a}$$

where I_w is the spinning top's principal inertia (about $\vec{z}_w$) and I_r its radial inertia. When expressed in frame $\mathcal{R}_a$, equation (2.7) becomes:

$$\left[M_P^{\mathcal{A}}\right]_{\mathcal{R}_a}(\mathbf{s}) = \left[\tau_{AP}^T\right]_{\mathcal{R}_a} \begin{bmatrix} m^{\mathcal{A}} I_3 & 0_{3\times3} \\ 0_{3\times3} & \left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a} \end{bmatrix}_{\mathcal{R}_a} [\tau_{AP}]_{\mathcal{R}_a} + \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_w\Omega \begin{bmatrix} 0 & 1/\mathbf{s} & 0 \\ -1/\mathbf{s} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix}_{\mathcal{R}_a} . \tag{2.8}$$

Proof of the model (2.7)

At point A, the appendage's centre of mass, one can write:

$$\overrightarrow{F}_{\mathcal{B}/\mathcal{A}} = m^{\mathcal{A}} \overrightarrow{a}_A \tag{2.9}$$

$$\overrightarrow{T}_{\mathcal{B}/\mathcal{A},A} = \frac{d\overrightarrow{H}^{\mathcal{A}}}{dt}|_{\mathcal{R}_b} + \overrightarrow{\omega} \wedge \overrightarrow{H}^{\mathcal{A}} = \frac{d\overrightarrow{H}^{\mathcal{A}}}{dt}|_{\mathcal{R}_a} + \overrightarrow{\omega} \wedge \overrightarrow{H}^{\mathcal{A}} \quad \text{(from [H1])} \tag{2.10}$$

where $\overrightarrow{H}^{\mathcal{A}} = \mathbb{I}_A^{\mathcal{A}}\overrightarrow{\omega} + I_w\Omega\vec{z}_w$ is the total angular momentum of the appendage.

$$\begin{aligned} \overrightarrow{T}_{\mathcal{B}/\mathcal{A},A} &= \mathbb{I}_A^{\mathcal{A}}\dot{\overrightarrow{\omega}} + I_w \underbrace{\frac{d(\Omega\vec{z}_w)}{dt}|_{\mathcal{R}_a}}_{=0\ ([\text{H4}])} + \underbrace{\overrightarrow{\omega} \wedge \mathbb{I}_A^{\mathcal{A}}\overrightarrow{\omega}}_{=0\ ([\text{H2}])} + \overrightarrow{\omega} \wedge I_w\Omega\vec{z}_w \\ &= \mathbb{I}_A^{\mathcal{A}}\dot{\overrightarrow{\omega}} - I_w\Omega({}^*\vec{z}_w)\overrightarrow{\omega} \\ &= \left(\mathbb{I}_A^{\mathcal{A}} - \frac{1}{s}I_w\Omega({}^*\vec{z}_w)\right)\dot{\overrightarrow{\omega}} \end{aligned} \tag{2.11}$$

From (2.9) and (2.11), the direct dynamics model of the appendage at point A is:

$$M_A^{\mathcal{A}}(\mathbf{s}) = \begin{bmatrix} m^{\mathcal{A}} I_3 & 0_{3\times3} \\ 0_{3\times3} & \mathbb{I}_A^{\mathcal{A}} \end{bmatrix} + \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -\frac{1}{\mathbf{s}}I_w\Omega\,({}^*\vec{z}_w) \end{bmatrix} .$$

Transporting this model to point P, we get $M_P^{\mathcal{A}}(\mathbf{s}) = \tau_{AP}^T M_A^{\mathcal{A}}(\mathbf{s})\tau_{AP}$, that leads to equation (2.7) when taking the first of the following remarks into account.

Remarks (independent of the projection frame used):

- $\tau_{AP}^T \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -\frac{1}{\mathbf{s}} I_w \Omega \, (^*\vec{z}_w) \end{bmatrix} \tau_{AP} = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & -\frac{1}{\mathbf{s}} I_w \Omega \, (^*\vec{z}_w) \end{bmatrix}, \ \forall \ \tau_{AP}.$

- It clearly appears in (2.8) that the model of an embedded angular momentum appendage is a second order 6×6 model containing 2 integrators. These 2 integrators were used to model the hub's rotation vector $\vec{\omega}$ from the angular acceleration $\dot{\vec{\omega}}$ (equation (2.11)), that is the last 3 inputs of direct dynamics model.

- When computing a minimal realization of the inverse dynamics model $[M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})]^{-1}$ augmented with the 6 integrators needed to compute the speeds from the accelerations, these previous integrators disappear.

- When considering a hub fitted with N_a embedded angular momentum appendages $\mathcal{A}_i$, $i = 1, \cdots, N_a$ (for instance, reaction wheels), the dynamics model of the assembly $M_B^{\sum_{i=1}^{N_a} \mathcal{A}_i + \mathcal{B}}(\mathbf{s})$, which is a $2N$ order model, should reduce to a second-order model because the matrix $(^*\vec{h})$ is rank 2 for all vectors $\vec{h}$ and:

$$\sum_{i=1}^{N_a} \frac{1}{\mathbf{s}} I_{w_i} \Omega_i \, (^*\vec{z}_{w_i}) = \frac{1}{\mathbf{s}} \sum_{i=1}^{N_a} I_{w_i} \Omega_i \, (^*\vec{z}_{w_i}) = \frac{1}{\mathbf{s}} (^*\vec{h})$$

  where $\vec{h} = \sum_{i=1}^{N_a} I_{w_i} \Omega_i \vec{z}_{w_i}$ is the total angular momentum of the global assembly, containing the N_a spinning tops.

2.4 Inverse dynamics model

Figure 2.4 shows the block diagram of the inverse dynamics model $[M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})]^{-1}$, expressed in frame $\mathcal{R}_b$. It also shows that the appendage's direct dynamics model is a feedback for the hub's inverse dynamics model :

$$\left(D_B^{\mathcal{B}} + M_B^{\mathcal{A}}(\mathbf{s})\right)^{-1} = [D_B^{\mathcal{B}}]^{-1} \left(I_6 + M_B^{\mathcal{A}}(\mathbf{s})[D_B^{\mathcal{B}}]^{-1}\right)^{-1} .$$

The usefulness of this representation instead of the direct inversion turns up when conducting sensitivity or robustness analyses with respect to parametric variations of the appendage's model. Indeed, there is no need to invert the direct dynamics model of the appendage $M_P^{\mathcal{A}}(\mathbf{s})$. Moreover, one can assure that each characteristic parameter of the model $M_P^{\mathcal{A}}(\mathbf{s})$ ($m^{\mathcal{A}}$, $\mathbb{I}_P^{\mathcal{A}}$, ω_i, ξ_i, $l_{i,P}$, Ω, $\cdots$) appear a minimum number of occurences.

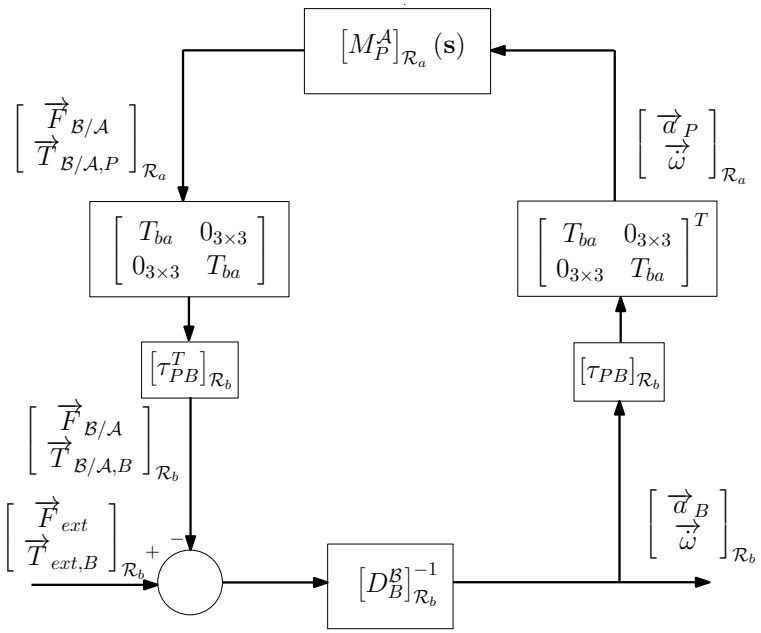

Figure 2.4: Inverse dynamics model block diagram of the appendage $[M_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})]^{-1}$ expressed in frame $\mathcal{R}_b$.

2.5 Revolute joint between the appendage and the hub

Let us consider Figure 2.5 in which the joint between the hub $\mathcal{B}$ and the appendage $\mathcal{A}$ at point P is a revolute joint along $\vec{r}_a$ axis. We use the following definitions:

- $\ddot{\theta}$ is the angular acceleration inside the revolute joint:

$$\dot{\vec{\omega}}_{\mathcal{A}/\mathcal{B}} = \ddot{\theta}\,\vec{r}_a \text{ or } \left[\dot{\vec{\omega}}_{\mathcal{A}/\mathcal{B}}\right]_{\mathcal{R}_a} = \ddot{\theta} \begin{bmatrix} x_{r_a} \\ y_{r_a} \\ z_{r_a} \end{bmatrix}_{\mathcal{R}_a},$$

- C_m is the torque (if present) along $(P, \vec{z}_a)$ applied by an actuator located inside the revolute joint.

The objective is to compute the augmented direct model $P_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$ (7×7) of the assembly $\mathcal{A}+\mathcal{B}$ such that:

$$\begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext,B} \\ C_m \end{bmatrix} = P_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s}) \begin{bmatrix} \vec{a}_B \\ \dot{\vec{\omega}} \\ \ddot{\theta} \end{bmatrix}$$

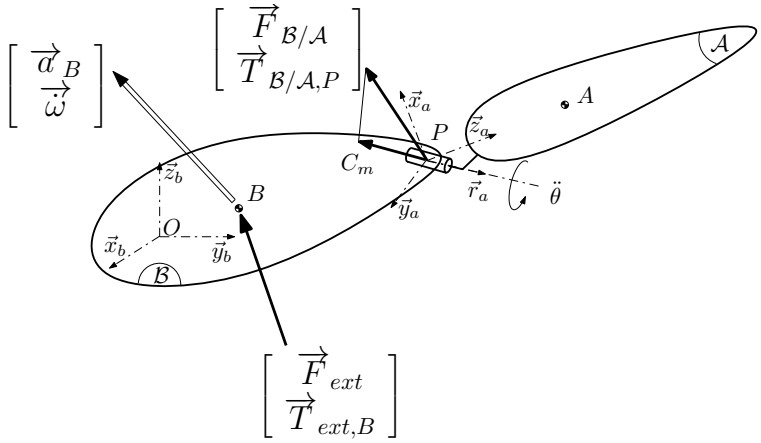

Figure 2.5: Assembly of the base $\mathcal{B}$ and the appendage $\mathcal{A}$ linked with a revolute joint along $\vec{z}_a$

Because of the revolute joint, the projection of the torque $\vec{T}_{\mathcal{B}/\mathcal{A},P}$, exerted by the base on the appendage at point P, along $\vec{r}_a$ axis is either: null in case of a free revolute joint or equal to C_m in case of an actuated joint.

$$C_m = \vec{T}_{\mathcal{B}/\mathcal{A},P}.\vec{r}_a\,. \tag{2.12}$$

Expressing the direct dynamics model $M_P^{\mathcal{A}}(\mathbf{s})$ of the appendage at point P in frame $\mathcal{R}_a$ enables us to write that:

$$\begin{bmatrix} \vec{F}_{\mathcal{B}/\mathcal{A}} \\ \vec{T}_{\mathcal{B}/\mathcal{A},P} \end{bmatrix} = M_P^{\mathcal{A}}(\mathbf{s}) \begin{bmatrix} \vec{a}_P \\ \vec{\dot{\omega}} + \ddot{\theta}\vec{r}_a \end{bmatrix} \tag{2.13}$$

From (2.12) and (2.13), one can write the augmented direct model (7×7) of the appendage $\left[P_P^{\mathcal{A}}\right]_{\mathcal{R}_a}(\mathbf{s})$ at point P and expressed in frame $\mathcal{R}_a$:

$$\begin{bmatrix} \begin{bmatrix} \vec{F}_{\mathcal{B}/\mathcal{A}} \\ \vec{T}_{\mathcal{B}/\mathcal{A},P} \end{bmatrix}_{\mathcal{R}_a} \\ C_m \end{bmatrix} = \underbrace{\begin{bmatrix} I_6 \\ 0\;0\;0\;x_{r_a}\;y_{r_a}\;z_{r_a} \end{bmatrix} \left[M_P^{\mathcal{A}}\right]_{\mathcal{R}_a}(\mathbf{s}) \begin{bmatrix} 0 \\ 0 \\ 0 \\ I_6 \quad x_{r_a} \\ y_{r_a} \\ z_{r_a} \end{bmatrix}}_{\left[P_P^{\mathcal{A}}\right]_{\mathcal{R}_a}(\mathbf{s})} \begin{bmatrix} \begin{bmatrix} \vec{a}_P \\ \vec{\dot{\omega}} \end{bmatrix}_{\mathcal{R}_a} \\ \ddot{\theta} \end{bmatrix}. \tag{2.14}$$

The block diagram of Figure 2.6 represents this operation. It also shows the connection of the first six inputs and outputs between $\left[P_P^{\mathcal{A}}\right]_{\mathcal{R}_a}(\mathbf{s})$ and the hub's direct model $D_B^{\mathcal{B}}$ in order to get the assembly model $P_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$, expressed in frame $\mathcal{R}_b$.

Taking into account a revolute joint between the hub and an appendage with an embedded angular momentum then allows for the modelling of CMGs (Control Moment Gyros), the axis of the joint being the precession axis of the CMG.

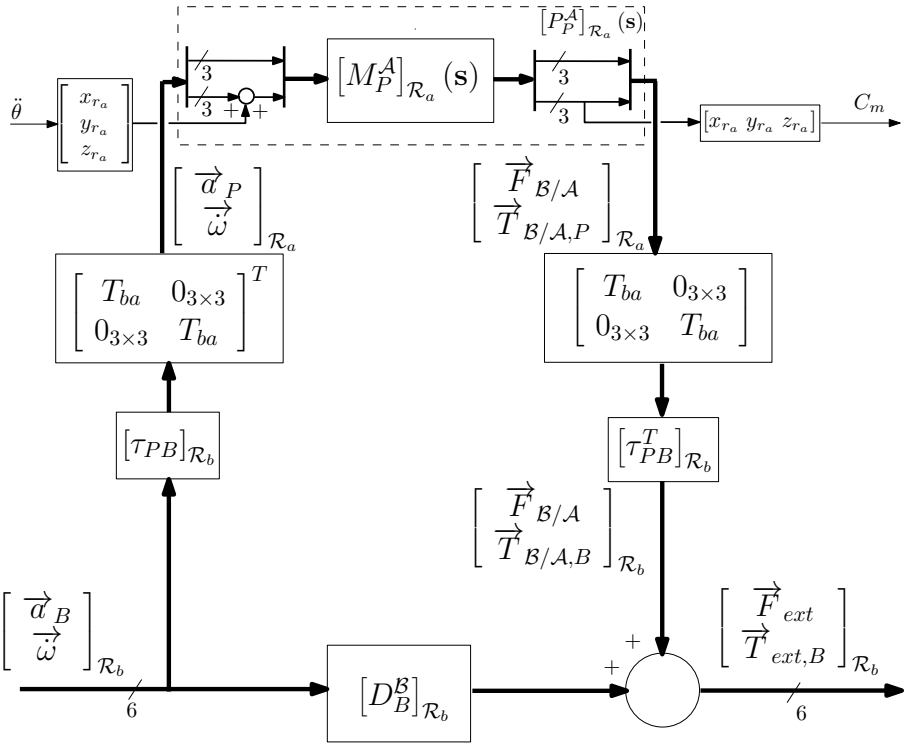

Figure 2.6: Direct dynamics model (7×7) block diagram of the assembly hub + appendage $P_B^{\mathcal{A}+\mathcal{B}}(\mathbf{s})$ with revolute joint, expressed in frame $\mathcal{R}_b$.

For control laws synthesis, one can use the following inverse model:

$$\begin{bmatrix} \begin{bmatrix} \vec{a}_B \\ \vec{\dot{\omega}} \end{bmatrix}_{\mathcal{R}_b} \\ \ddot{\theta} \end{bmatrix} = \left[P_B^{\mathcal{A}+\mathcal{B}}\right]_{\mathcal{R}_b}^{-1}(\mathbf{s}) \begin{bmatrix} \begin{bmatrix} \vec{F}_{ext} \\ \vec{T}_{ext,B} \end{bmatrix}_{\mathcal{R}_b} \\ C_m \end{bmatrix} \tag{2.15}$$

that allows the user to introduce, between the seventh input to the seventh output, a local model of the joint mechanism or controller $K(s)$ according to Figure 2.7.

2.6 Generalisation

One can generalise the previous approach:

- to take N_a appendages ($\mathcal{A}_i$, $i = 1, \cdots, N_a$) linked with the hub at points P_i into account,

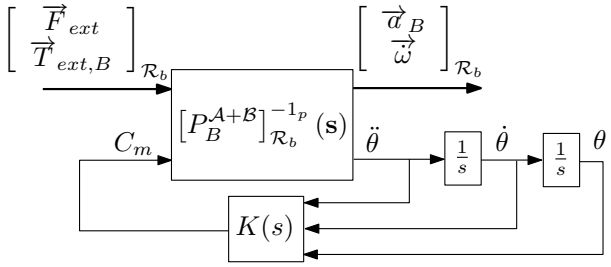

Figure 2.7: Inverse dynamics model of the assembly hub + appendage with a revolute joint and a local mechanism or controller model $K(s)$, expressed in frame $\mathcal{R}_b$.

- to take into account that an appendage $\mathcal{A}_i$ can itself be the support (the parent) of another appendage $\mathcal{A}_{ij}$ at point P_{ij}. Thus, one can model any open kinematic chain of bodies that can be described through a connection tree. To do this, one has to build the global model, starting from the tree leaves and using the previous approach where the body $\mathcal{A}_i$ is considered as a base (hub) or the parent of appendage $\mathcal{A}_{ij}$. However, because of assumption [H1], each intermediate body must be rigid. Flexible bodies are allowed only at the end of the kinematic chain.

The intrinsic formulae (they do not depend on a projection frame) that enable transport (2.3) or addition (2.4) of direct dynamics models allow to build the global model and to use it recursively. The SDT toolbox was developed on this principle. For example, one can write the global direct dynamics model of the system shown on Figure 2.8 as follows:

$$\begin{aligned} M_B^{\sum \mathcal{A}_i+\mathcal{B}}(\mathbf{s}) &= D_B^{\mathcal{B}} \\ &+ \tau_{P_1B}^T \left(D_{P_1}^{\mathcal{A}_1} + \tau_{P_{11}P_1}^T M_{P_{11}}^{\mathcal{A}_{11}}(\mathbf{s}) \tau_{P_{11}P_1} \right) \tau_{P_1B} \\ &+ \tau_{P_2B}^T \left(M_{P_2}^{\mathcal{A}_2}(\mathbf{s}) \right) \tau_{P_2B} \ . \end{aligned}$$

One can also use this generalization to model revolute joints between elements, as detailed in the previous section. Thus, it allows for the modelling of a gyroscopic actuator (CMG) fitted on a hub using three bodies as shown on Figure 2.9:

- the hub $\mathcal{B}$,
- the fork $\mathcal{A}_i$ connected through a revolute joint (precession axis $\vec{r}_{a_i} = \vec{z}_{a_i}$) with $\mathcal{B}$ at point P_i,
- the spinning top $\mathcal{A}_{ii} \equiv \mathcal{W}_i$ linked with $\mathcal{A}_i$ at point $P_{ii} = G_i$ and with its angular momentum along $\vec{z}_{w_i}$ (according to the assumptions [H6]).

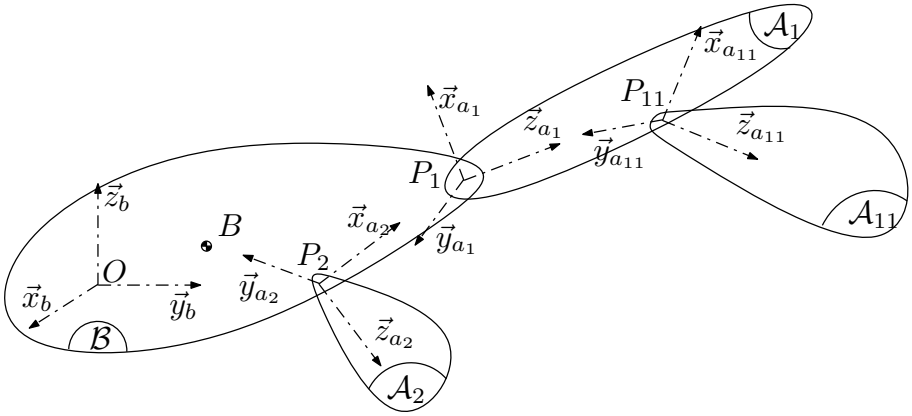

Figure 2.8: Example of an open kinematic chain

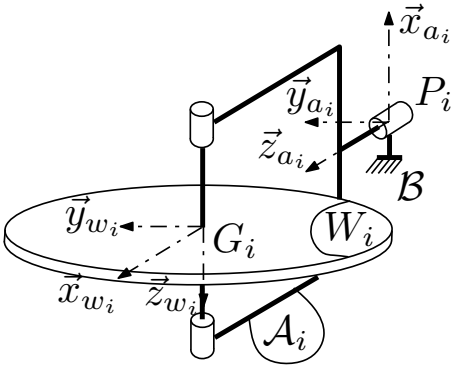

Figure 2.9: Illustration of a CMG.

Chapter 3

Implementation with Matlab®

The MATLAB® package `SDT_V1.3.zip` to implement the Satellite Dynamics Toolbox can be downloaded from:
http://personnel.isae.fr/daniel-alazard/matlab-packages/satellite-dynamics-toolbox.html.
This toolbox requires a user defined data file describing the structure of the spacecraft. The variables to be declared by such a script file are described in section 3.2. The MATLAB® package contains also several examples of such a script file detailled in sections 3.4 to 3.9 and the associated demo file `demoSTD.m`. It is advised to the user to run step by step this demo file and to use one of the data file examples as a guide to create its own data file. The next section describes the various functions included in the toolbox in relation with the general notations used in the document and summarized in the nomenclature. The inline `help` of these functions are given in appendix A.

3.1 Satellite Dynamics Toolbox functions

Once the user defined data script file is created, 5 functions can be used: `main`, `translate_dynamic_model`, `rotate_dynamic_model`, `kinematic_model` and `antisym`. Let us called `nom_fich.m` the user defined data script file:

- `main.m`: using the syntax `mod=main('nom_fich')`, the function outputs a structure variable `mod` with 5 fields:
  - `mod.TotalMass`: the total mass of the satellite ($\sum_{i=1}^{N_a} m^{\mathcal{A}_i} + m^{\mathcal{B}}$, in kg),
  - `mod.TotalCg`: the coordinates of the global centre of mass expressed in the reference frame $\mathcal{R}_b$: ($\left[\overrightarrow{OG}\right]_{\mathcal{R}_b}$, in m),
  - `mod.DynamicModel`: the direct dynamics model expressed at the global

centre of mass (G) in the reference frame $\mathcal{R}_b$: $\left(\left[M_G^{\sum_{i=1}^{N_a}\mathcal{A}_i+\mathcal{B}}\right]_{\mathcal{R}_b}(\mathbf{s})\right.$ in SI units, meaning that the accelerations in input of the model are in m/s^2 and in rad/s^2 and that the forces and torques outputted by the model are in N and Nm),

- `mod.InverseTotalModel`: the above inverse model (inputs are forces, outputs are accelerations)
- `mod.liste_IOs`: a string of chars that describes the different channels of the (square) model `mod.DynamicModel` and built from the names of the bodies written in `nom_fich.m`. The first 6 channels are always relative to the 3 translations and the 3 rotations of the hub (base). The optional 7th and beyond channels are the transfers between the torques C_m and accelerations $\ddot{\theta}$ local to each revolute joint linking two bodies, if activated.

- `translate_dynamic_model.m`: the syntax `MB=translate_dynamic_model(A,B,MA)` translates (or transports) the direct dynamics model `MA` projected in a frame $\mathcal{R}_i$ from point `A` (vector of the 3 components of the point A in the frame $\mathcal{R}_i$) to point `B` (vector of the 3 components of the point B in the frame $\mathcal{R}_i$). Inputs and output arguments are expressed in the same frame $\mathcal{R}_i$:

$$[M_B]_{\mathcal{R}_i}(\mathbf{s}) = \begin{bmatrix} [\tau_{AB}]^T_{\mathcal{R}_i} & 0_{6\times N_r} \\ 0_{N_r\times 6} & I_{N_r} \end{bmatrix} [M_A]_{\mathcal{R}_i}(\mathbf{s}) \begin{bmatrix} [\tau_{AB}]_{\mathcal{R}_i} & 0_{6\times N_r} \\ 0_{N_r\times 6} & I_{N_r} \end{bmatrix}.$$

  where N_r is the number of revolute joints in the structures.

- `rotate_dynamic_model.m`: the syntax `MB=rotate_dynamic_model(MA,ANG,AXIS)` computes the direct or the inverse dynamics model (input argument `MA`) projected in a frame $\mathcal{R}_i$ after a rotation of `ANG` (deg) around the axis `AXIS` (unit vector of the 3 components of the rotation axis $\vec{r}_a$ in the frame $\mathcal{R}_i$). Inputs and output arguments are expressed in the same frame $\mathcal{R}_i$. This function computes the direction cosine matrix T associated to the rotation and then:

$$[M_B]_{\mathcal{R}_i}(\mathbf{s}) = \begin{bmatrix} T & 0_{3\times 3} & 0_{3\times N_r} \\ 0_{3\times N_r} & T & 0_{3\times N_r} \\ 0_{N_r\times 3} & 0_{N_r\times 3} & I_{N_r} \end{bmatrix} [M_A]_{\mathcal{R}_i}(\mathbf{s}) \begin{bmatrix} T^T & 0_{3\times 3} & 0_{3\times N_r} \\ 0_{3\times N_r} & T^T & 0_{3\times N_r} \\ 0_{N_r\times 3} & 0_{N_r\times 3} & I_{N_r} \end{bmatrix}.$$

  where N_r is the number of revolute joints in the structures.

- `kinematic_model.m`: the syntax `TAU=kinematic_model(A,B)` computes the kinematic model `TAU` betwen points `A` and `B` (vectors of the 3 components of points A and B in a frame $\mathcal{R}_i$):

$$[\tau_{AB}]_{\mathcal{R}_i} = \begin{bmatrix} I_3 & \left[(^*\overrightarrow{AB})\right]_{\mathcal{R}_i} \\ 0_{3\times 3} & I_3 \end{bmatrix}.$$

- `antisym.m`: the syntax `M=antisym(V)` computes the skew symmetric matrix $\mathtt{M} = \left[(^*\overrightarrow{V})\right]_{\mathcal{R}_i}$ associated to the vector $\mathtt{V} = \overrightarrow{V}$ (3 components) projected in the frame $\mathcal{R}_i$ (see nomenclature).

3.2 User defined data file `nom_fich.m`

From a reference frame $\mathcal{R}_b$, the user-defined data file `nom_fich.m` describes the geometry, the dynamics parameters of the main body (hub) and of the different appendages $\mathcal{A}_i$. Running this file through:

```
>> nom_fich
```

must creates 3 variables:

- `MB`: structure describing the main body $\mathcal{B}$. The different required fields of this structure are detailed in table 3.1,
- `nappend`: the number of appendages N_a (integer),
- `SA`: vector of N_a cells. Each cell `SA{i}`, `i=1:nappend`, describes the appendage $\mathcal{A}_i$. The different required fields of the structure `SA{i}` are detailed in table 3.2.

3.2.1 Remarks

One can also define appendage `SA{i}` in an independent data file `nom_fich_i.m` (see table 3.2, line 6), that creates the same variables `MB`, `nappend` and `SA` as `nom_fich.m`. It allows the user:

- to use the recursivity of `main.m` in order to model chains of rigid bodies (see example data files `SpaceRoboticArm.m` and files `Segment1.m` to `Segment6.m` provided with the toolbox. They model a robotic manipulator on a space platform).
- to use only one data file when the platform is fitted with several identical appendages. We can meet this case when modelling a platform fitted with a cluster of gyroscopic actuators (see example data file presented in section 3.7).

Variable name	Notation	Type	Unit
MB.Name		char	
MB.cg	$\left[\overrightarrow{OB}\right]_{\mathcal{R}_b}$	double 3×1	m
MB.m	$m^{\mathcal{B}}$	double	Kg
MB.Ixx	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(1,1)$	double	$Kg\,m^2$
MB.Iyy	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(2,2)$	double	$Kg\,m^2$
MB.Izz	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(3,3)$	double	$Kg\,m^2$
MB.Ixy	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(1,2)$	double	$Kg\,m^2$
MB.Ixz	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(1,3)$	double	$Kg\,m^2$
MB.Iyz	$\left[\mathbb{I}_B^{\mathcal{B}}\right]_{\mathcal{R}_b}(2,3)$	double	$Kg\,m^2$
nappend	N_a	integer	

Table 3.1: Description of the fields of the structure MB.

3.2.2 A first tutorial (on inertia tensor specified at any reference point)

In the user defined data file nom_fich.m, the tensor of inertia of each body (fields MB.I** and SA{i}.I**) must be defined w.r.t. to its centre of mass. In some applications, data are provided at a reference point O different from the centre of mass B, then Huygens thereom (1.15) can be used in the script file nom_fich.m by using the following Matlab® sequence based on the function antisym:

```
MB.Name='Platform';                    % Name of the body
MB.cg = [ 0.0063;-0.0200;1.7744];      % position of gravity center
                                       % in (O,X,Y,Z) (m)
MB.m =  905.6290;                      % mass (kg)

MB.IO=[3.8991*1.0e+03 0.0086*1.0e+03 -0.0265*1.0e+03;... % Main inertia in
       0.0086*1.0e+03 4.0907*1.0e+03 0.0114*1.0e+03;...  % (O,X1,Y1,Z1)
      -0.0265*1.0e+03 0.0114*1.0e+03 0.4606*1.0e+03];    % (Kg m^2)

MB.I=MB.IO + MB.m*antisym(MB.cg)^2; % Huygens theorem
MB.Ixx=MB.I(1,1);       % Main Inertia in
MB.Iyy=MB.I(2,2);       % (CG1,X1,Y1,Z1) (kgm^2)
MB.Izz=MB.I(3,3);
MB.Ixy=MB.I(1,2);       % Cross Inertia in
MB.Ixz=MB.I(1,3);       % (CG1,X1,Y1,Z1) (kgm^2)
MB.Iyz=MB.I(2,3);
```

In this example the HUYGENS thereom is applied to the main body `MB` but the same procedure can be applied to any appendage `SA{i}`.

3.3 FAQ

- *The field* `liste_IOs` *returned by the function* `main` *is wrong* (?):
  the function `main.m` is called recursively and uses persistent variables. In a nominal use, the persistent variables are managed inside the function `main.m` but if the function returned on an error message (due to wrong data in `nom_fich.m` for instance), it is then required to clear persistent variables for the next call by the command:

  ```
  clear main
  ```

- *The fields* `DynamicModel` *or* `InverseTotalModel` *returned by the function* `main` *contains some* `NaN` (?):
  the mass or inertia seen from one of the d.o.fs is null. There is a mistake in the file `nom_fich.m`.

- *The* `ss` *model returned by the function* `main` *in the field* `DynamicModel` *or* `InverseTotalModel` *contains unstable dynamics* (?):
  the mass or inertia of one or more bodies is not definite positive. There is a mistake in the file `nom_fich.m`.

	Variable name	Notation	Type	Unit
1	`SA{i}.Name`		char	
2	`SA{i}.P`	$\left[\overrightarrow{OP}\right]_{\mathcal{R}_b}$	double 3×1	m
3	`SA{i}.TM`	T_{ba}	double 3×3	
4(*)	`SA{i}.pivot`		boolean	
5(*)	`SA{i}.pivotdir`	$[\vec{r}_a]_{\mathcal{R}_a} = [x_{r_a}\ y_{r_a}\ z_{r_a}]^T$	double 3×1	
6(**)	`SA{i}.angle`	θ	double	deg
7(***)	`SA{i}.filename`		char	
8	`SA{i}.cg`	$\left[\overrightarrow{PA}\right]_{\mathcal{R}_a}$	double 3×1	m
9(#)	`SA{i}.omega`	Ω	double	rad/s
10	`SA{i}.m`	$m^{\mathcal{A}}$	double	Kg
11	`SA{i}.Ixx`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(1,1)$	double	$Kg\,m^2$
12	`SA{i}.Iyy`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(2,2)$	double	$Kg\,m^2$
13	`SA{i}.Izz`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(3,3)$	double	$Kg\,m^2$
14	`SA{i}.Ixy`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(1,2)$	double	$Kg\,m^2$
15	`SA{i}.Ixz`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(1,3)$	double	$Kg\,m^2$
16	`SA{i}.Iyz`	$\left[\mathbb{I}_A^{\mathcal{A}}\right]_{\mathcal{R}_a}(2,3)$	double	$Kg\,m^2$
17(##)	`SA{i}.flex`		integer	
18	`SA{i}.L`	$[L_P]_{\mathcal{R}_a}$ ou $[L_A]_{\mathcal{R}_a}$	double $N \times 6$	$\sqrt{Kg}$, $\sqrt{Kg}m$
19	`SA{i}.wi`	$[\omega_1\ \omega_2\ \cdots\ \omega_N]$	double $1 \times N$	rad/s
20	`SA{i}.z`	$[\xi_1\ \xi_2\ \cdots\ \xi_N]$	double $1 \times N$	

Table 3.2: Description of the fields of the structure in cell `SA{i}`.

(*): if `SA{i}.pivot=1` then there is a revolute joint between the appendage and the hub at point P. The direction $\vec{r}_a$) of the revolute joint axis can be specified in the field # 5 `SA{i}.pivotdir`, otherwize the defaut value is `SA{i}.pivotdir=[0;0;1]`. In case of a revolute joint, the direct dynamics model will be augmented on the outputs, by the joint torque C_m and on the inputs, by the joint acceleration $\ddot{\theta}$. If `SA{i}.pivot=0` then the appendage is cantilevered to the hub $\mathcal{B}$ at point P and the fields # 5 and 6 are optional or disregarded.

(**): if `SA{i}.pivot=1` then is possible to take into account a tilt of the appendage by `SA{i}.angle` degrees along the revolute joint axis.

(***): if the appendage is described by a data file, then the fields from line 8 to 20 are optional or disregarded.

(#) if the field `SA{i}.omega` exists, then the appendage is considered as an onboard angular momentum along $\vec{z}_{a_i}$. Then, `SA{i}.omega` and `SA{i}.Izz` are the angular rate and the main inertia of the spinning top, `SA{i}.Ixx` is its radial inertia, fields # 12, 14, 15 and 16 are optional or disregarded and the appendage must be rigid: `SA{i}.flex=0`.

(##): if `SA{i}.flex=0` then the appendage is supposed rigid and the fields from line 18 to 20 are optional or disregarded. If `SA{i}.flex=1` then the appendage is flexible and `SA{i}.L` is the list of the modal participation factors L_A expressed at centre of mass A of the appendage. If `SA{i}.flex=2` then the appendage is flexible and `SA{i}.L` is the list of the modal participation factors L_P at anchoring point P. We remind that $L_P = L_A \tau_{AP}$.

3.4 Example 1: Spacecraft1.m

This file describes a platform (`MB.Name='Platform'`) fitted with 2 (`nappend=2`) flexible appendages (`SA{1}` and `SA{2}`) cantilevered in 2 different points of the platform:

- a solar array (`SA{1}.Name = 'Solar Array'`) described by 3 flexible modes,
- an antenna (`SA{2}.Name = 'Antenna'`) described by 3 flexible modes.

The frequencies $\omega_i,\ i = 1, 2, 3$ of the 3 modes are the same for both appendages. Because of the symmetry of the two modal participation factors along the z axis of each appendage, one flexible mode is uncontrollable. It is reduced when computing a minimal realization of the model.
The following MATLAB® session (see also the script file `demoSTD.m`) shows how to use this file with the SDT and perform complementary analyses.

```
>> mod=main('Spacecraft1');
2 states removed.
>> % 1 flexible mode removed (due to symmetry of flexible appendage)
>> mod.TotalCg            % the position of the total centre of mass G.

ans =

    0.1909
    0.3909
         0

>> mod.liste_IOs          % list of channels

ans =

Trans. X Platform
Trans. Y Platform
Trans. Z Platform
Rot. X Platform
Rot. Y Platform
Rot. Z Platform

>> Md=mod.DynamicModel;         % direct dynamic model.

>> % Translation of the direct dynamic model from the global centre of mass G
>> % to the hub's centre of mass B:
>> Spacecraft1            % to get all the data in the workspace.
>> MD_B=translate_dynamic_model(mod.TotalCg,MB.cg,Md);
```

```
>> % Inverse Dynamic Model:
>> Mi=mod.InverseTotalModel;
>> damp(Mi)  % The 5=6-1 flexible modes.

      Eigenvalue          Damping     Frequency

 -4.03e-02 + 8.03e+00i     5.02e-03     8.03e+00
 -4.03e-02 - 8.03e+00i     5.02e-03     8.03e+00
 -5.07e-02 + 9.03e+00i     5.61e-03     9.03e+00
 -5.07e-02 - 9.03e+00i     5.61e-03     9.03e+00
 -9.64e-02 + 1.70e+01i     5.67e-03     1.70e+01
 -9.64e-02 - 1.70e+01i     5.67e-03     1.70e+01
 -1.09e-01 + 1.80e+01i     6.02e-03     1.80e+01
 -1.09e-01 - 1.80e+01i     6.02e-03     1.80e+01
 -1.11e-01 + 2.09e+01i     5.29e-03     2.09e+01
 -1.11e-01 - 2.09e+01i     5.29e-03     2.09e+01

(Frequencies expressed in rad/seconds)

>> figure
>> sigma(Mi) % Frequency domain response.
```

The frequency response is presented on Figure 3.1.

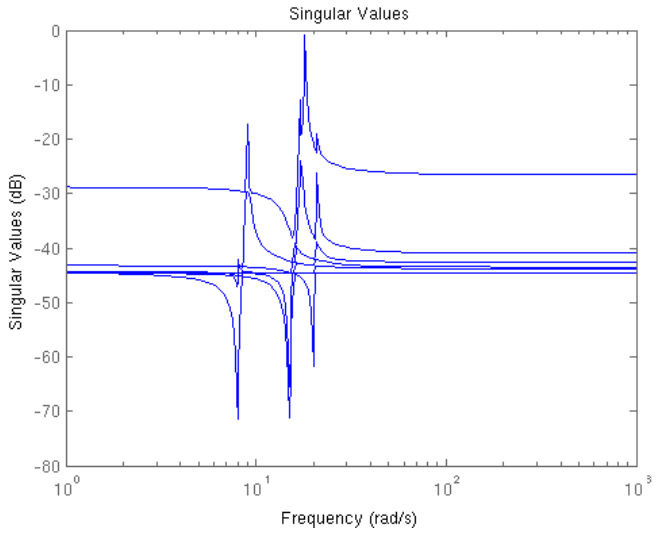

Figure 3.1: Frequency response (Singular Values) of the inverse dynamics model.

3.5 Example 2: Spacecraft2.m

This file describes the same platform as example 1 but now, the link between the solar array and the hub is a revolute joint along $\vec{e}_{a_1} = \vec{z}_{a_1}$. The following MATLAB® session (see also the script file `demoSTD.m`) shows how to use this file with the SDT and perform complementary analyses, more particularly study the response of the revolute joint's angle to a force impulse applied at the global centre of mass. One will notice that thanks to the revolute joint, the 6 flexible modes are now both controllable and observable: computing a minimal realization of the model will not reduce the number of modes.

```
>> clear all,close all,
>> mod=main('Spacecraft2');
>> mod.liste_IOs % Now the dynamic model is 7x7

ans =

Trans. X Platform
Trans. Y Platform
Trans. Z Platform
Rot. X Platform
Rot. Y Platform
Rot. Z Platform
Joint: Solar Array/Platform

>> Md=mod.DynamicModel;          % direct dynamic model.
>> Mi=mod.InverseTotalModel;     % inverse dynamic model
>> Mi71=Mi(7,1);  % Transfer between a force applied on the
>> %               platform along X axis and the revolute joint's
>> %               relative acceleration.
>> ddint=tf(1,[1 0 0]); % double integrations
>> % impulse response of Mi71/s/s :
>> figure
>> impulse(Mi71*ddint,3) % response of the revolute joint angle to
>> %                       an impulse on the X thruster.
```

The impulse response is shown on Figure 3.2.

Remark: The file `Spacecraft3.m` describes the same assembly but uses file `APPENDAGE1.m` to define the solar array. File `Spacecraft4.m` shows the usefulness of the recursivity used in function `main.m`: it describes the same assembly as `Spacecraft3.m`, with a third (more complex) appendage which is the assembly described in `Spacecraft3.m`...

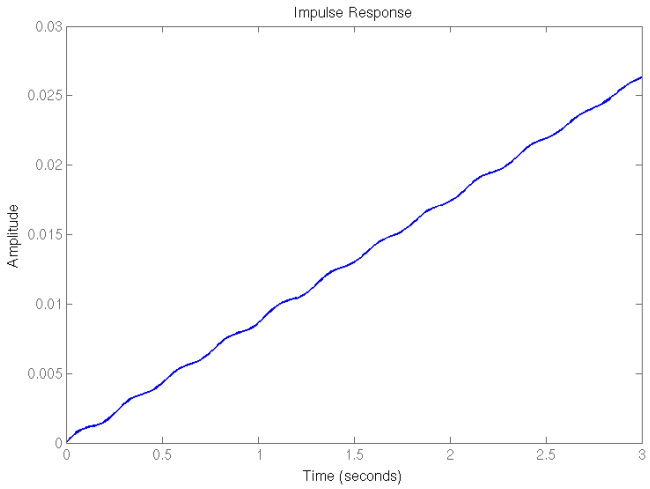

Figure 3.2: Impulse response of the transfer between θ, the angle of the revolute joint, and the force applied at the global centre of mass along X

3.6 Example 3: `Spacecraft5.m`

This file describes the same platform as example 1 but now, it is fitted with 3 more appendages (`nappend=5`) modelling embedded angular momentums along the 3 directions ($\vec{x}_b$, $\vec{y}_b$, $\vec{z}_b$) in the hub's frame. Such devices can model reaction wheels (RWA), that have a very high speed (close to their saturation).
The following Matlab® session (see also the script file `demoSTD.m`) shows how to use this file with the SDT and perform complementary analyses. One will notice the following facts:

- The reduction of one uncontrollable mode (same as example 1)
- The reduction of a double pair of integrators. Indeed, to model the 3 embedded angular momentum (see the remarks in section 2.3.3), we introduced three pairs of integrators in the direct dynamics model. However, only 2 integrators are required to model the gyroscopic couplings due to the total angular momentum (i.e. the sum of the 3 embedded angular momentums).
- The gyroscopic mode in the inverse dynamics model which is represented by a pair of poles on the imaginary axis, at low frequency (around $0.1\,rd/s$)
- The reduction of a pair of integrators when introducing the 3 integrators between the 3 angular accelerations and speeds of the hub. All the integrators added into the dynamics model to take into account gyroscopic couplings disappeared in the reduction, as expected.

```
>> clear all,close all,
>> mod=main('Spacecraft5');
2 states removed.
2 states removed.
2 states removed.
>> % 1 flexible mode removed (due to symmetry of flexible appendages)
>> % + 2 double integrators removed.
>> Md=mod.DynamicModel;          % direct dynamic model.
>> damp(Md) % only 2 integrators to represent the 3 angular momentums.

      Eigenvalue          Damping     Frequency

  1.36e-15                -1.00e+00     1.36e-15
 -5.68e-15                 1.00e+00     5.68e-15
 -4.00e-02 + 8.00e+00i     5.00e-03     8.00e+00
 -4.00e-02 - 8.00e+00i     5.00e-03     8.00e+00
 -4.00e-02 + 8.00e+00i     5.00e-03     8.00e+00
 -4.00e-02 - 8.00e+00i     5.00e-03     8.00e+00
 -7.50e-02 + 1.50e+01i     5.00e-03     1.50e+01
 -7.50e-02 - 1.50e+01i     5.00e-03     1.50e+01
 -7.50e-02 + 1.50e+01i     5.00e-03     1.50e+01
 -7.50e-02 - 1.50e+01i     5.00e-03     1.50e+01
 -1.00e-01 + 2.00e+01i     5.00e-03     2.00e+01
 -1.00e-01 - 2.00e+01i     5.00e-03     2.00e+01

(Frequencies expressed in rad/seconds)

>> Mi=mod.InverseTotalModel;
>> damp(Mi)

      Eigenvalue          Damping     Frequency

 -2.14e-11 + 9.21e-02i     2.32e-10     9.21e-02
 -2.14e-11 - 9.21e-02i     2.32e-10     9.21e-02
 -4.03e-02 + 8.03e+00i     5.02e-03     8.03e+00
 -4.03e-02 - 8.03e+00i     5.02e-03     8.03e+00
 -5.06e-02 + 9.02e+00i     5.61e-03     9.02e+00
 -5.06e-02 - 9.02e+00i     5.61e-03     9.02e+00
 -9.62e-02 + 1.70e+01i     5.67e-03     1.70e+01
 -9.62e-02 - 1.70e+01i     5.67e-03     1.70e+01
 -1.09e-01 + 1.80e+01i     6.02e-03     1.80e+01
 -1.09e-01 - 1.80e+01i     6.02e-03     1.80e+01
 -1.11e-01 + 2.09e+01i     5.29e-03     2.09e+01
 -1.11e-01 - 2.09e+01i     5.29e-03     2.09e+01
```

```
(Frequencies expressed in rad/seconds)

>> % ==> in addition to flexible modes, one can see a very low frequency
>> %     (around 0.1 rd/s) mode corresponding to the gyroscopic mode
>> figure
>> sigma(Mi) % ==> a very low fequency resonance.
>> % Angular model:
>> Mi=Mi(4:6,4:6);
>> % Integrations between angular acceleration and rates:
>> int=tf(1,[1 0]);
>> Mi_int=minreal(Mi*(int*eye(3)));
2 states removed.
>> % ==> 2 states removed: extra integrators to model gyroscopic couplings
>> %     are removed, as wanted.
```

The frequency-domain response is shown on Figure 3.3.

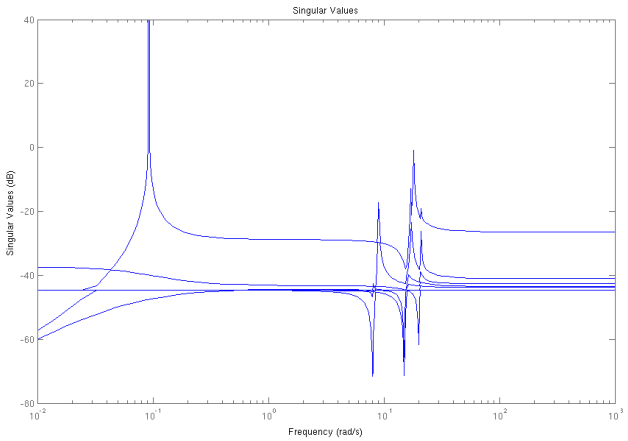

Figure 3.3: Frequency-domain response (Singular Values) of the inverse dynamics model.

3.7 Example 4: `FOUR_CMGs.m`

This file describes the example of a platform fitted with a cluster of 4 identical gyroscopic actuators, in a pyramidal configuration. The geometries, weights and inertias of the different bodies are very close to those of the experimental platform TETRAGYRE developed by ONERA/DCSD (see http://www.onera.fr/dcsd/gyrodynes/

and Figure 3.4). The default configuration is obtained when the spinning tops axes are horizontal and when the sum of the angular momentums is zero.

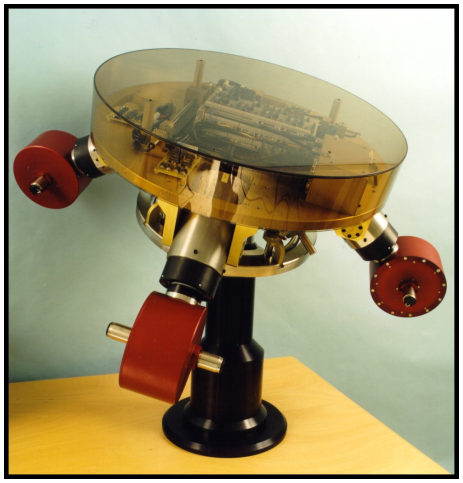

Figure 3.4: The bench TETRAGYRE.

The main data file is `FOUR_CMGS.m`. It describes the platform (hub) $\mathcal{B}$ and the nominal configuration of the 4 CMGs. One difference with the example files detailed above is that we add a global variable `angles` (4×1) that represents the current configuration of the precession axes. It also enables the computation of the dynamics model in any configuration without modifying the data file. This variable is initialized with the nominal angular configuration (`angles=[0;0;0;0];`). The file `dataCMG.m` describes one CMG (see Figure 2.9) and is called 4 times by the file `FOUR_CMGS.m`. The CMG is modelled as a fork $\mathcal{A}_i$ for the main part of the appendage and a spinning top $\mathcal{W}_i$ as the appendage's appendage.
The following MATLAB® session (see also the script file `demoSTD.m`) shows how to use this file with the SDT, performs complementary analyses and proposes a 3-axes attitude control law. This control law has the following structure, also depicted in Figure 3.5:

- a inner control loop feedbacking the precession speeds $\dot{\sigma}_i$. It is a pure proportional control and it is the same on the 4 precession axes:

$$C_{m_i} = 0.06(\dot{\sigma}_{i,ref} - \dot{\sigma}_i), \forall\, i = 1, \cdots, 4\ ,$$

- an outer control loop feedbacking the 3 hub attitudes. It is a pure proportional control and it is the same on the hub's 3 rotation axes:

$$\left[\overrightarrow{T}_{ref}\right]_{\mathcal{R}_b} = 3[(\phi_{ref} - \phi) \quad (\theta_{ref} - \theta) \quad (\psi_{ref} - \psi)]^T\ ,$$

- a guidance law for the cluster of gyroscopic actuators which is the pseudo-inverse of the Jacobian $J_0 = J(\sigma_i)|_{\sigma_i=0}$ (3×4) computed from the 4 precession speeds to the hub's 3 rotation speeds when in nominal configuration:

$$[\dot{\sigma}_{1,ref} \quad \dot{\sigma}_{2,ref} \quad \cdots \quad \dot{\sigma}_{4,ref}]^T = J_0^T (J_0 J_0^T)^{-1} \left[\overrightarrow{T}_{ref} \right]_{\mathcal{R}_b} .$$

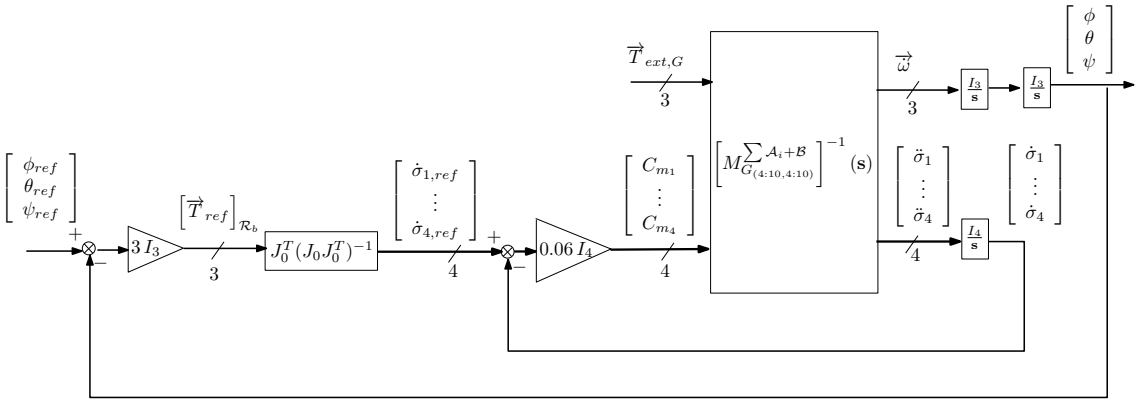

Figure 3.5: Attitude control using 4 CMGs.

```
>> clear all,close all,
>> mod=main('FOUR_CMGs');
Angular configuration: 0  0  0  0
2 states removed.
>> mod.liste_IOs          % Definition of the various channels

ans =

Trans. X Platform
Trans. Y Platform
Trans. Z Platform
Rot. X Platform
Rot. Y Platform
Rot. Z Platform
Joint: CMG #1/Platform
Joint: CMG #2/Platform
Joint: CMG #3/Platform
Joint: CMG #4/Platform

>> Md=mod.DynamicModel;  % direct dynamic model
>> % Model restricted to 3 axes angular motion + CMGs channels:
>> Md=Md(4:10,4:10);
```

```
>> % One can check that the total angular momentum is null:
>> M3x3=minreal(Md(1:3,1:3))
6 states removed.

M3x3 =

  d =
              u1          u2          u3
   y1      5.315           0  -0.005015
   y2          0       5.315  -0.005015
   y3  -0.005015  -0.005015      10.61

Static gain.

>> %  ==> The 3x3 model is static: no more gyroscopic couplings between platform
>> %  and inertial frame.

>> Mi=inv(Md);  % Inverse dynamic model.
>> damp(Mi)

       Eigenvalue          Damping     Frequency

 -3.55e-15 + 4.85e+01i     7.33e-17     4.85e+01
 -3.55e-15 - 4.85e+01i     7.33e-17     4.85e+01
 -5.33e-15 + 4.85e+01i     1.10e-16     4.85e+01
 -5.33e-15 - 4.85e+01i     1.10e-16     4.85e+01
 -7.25e-17 + 8.40e+01i     8.63e-19     8.40e+01
 -7.25e-17 - 8.40e+01i     8.63e-19     8.40e+01

(Frequencies expressed in rad/seconds)

>> % ==> the pulsations of the 3 gyroscopic modes.

>> % Dynamic model for a new angular configuration:
>> global angles
>> angles=[10;20;-5;-25];
>> mod_bis=main('FOUR_CMGs');
Angular configuration: 10  20  -5 -25
2 states removed.
>> damp(mod_bis.InverseTotalModel)
       Eigenvalue          Damping     Frequency

 -1.24e-16 + 3.13e+01i     3.95e-18     3.13e+01
 -1.24e-16 - 3.13e+01i     3.95e-18     3.13e+01
```

```
  7.11e-15 + 6.32e+01i    -1.12e-16     6.32e+01
  7.11e-15 - 6.32e+01i    -1.12e-16     6.32e+01
  8.29e-17 + 8.59e+01i    -9.65e-19     8.59e+01
  8.29e-17 - 8.59e+01i    -9.65e-19     8.59e+01
(Frequencies expressed in rad/seconds)
>> % ==> angular frequecies of gyroscopic modes have changed.

>> % First integrators (angular rates on outputs on nominal model Mi):
>> int=tf(1,[1 0]);
>> Gv=minreal((int*eye(7))*Mi);
3 states removed.

>> % Precession rate servo-loop (inner loop):
>> bf2=feedback(Gv*diag([1 1 1 0.06 0.06 0.06 0.06]),eye(4),4:7,4:7);
>> bf2=minreal(bf2);  % 3 states removed: OK !!
3 states removed.
>> damp(bf2)

       Eigenvalue          Damping     Frequency

 -3.00e+01 + 3.81e+01i     6.18e-01     4.85e+01
 -3.00e+01 - 3.81e+01i     6.18e-01     4.85e+01
 -3.00e+01 + 3.81e+01i     6.18e-01     4.85e+01
 -3.00e+01 - 3.81e+01i     6.18e-01     4.85e+01
 -5.99e+01                 1.00e+00     5.99e+01
 -3.00e+01 + 7.85e+01i     3.57e-01     8.40e+01
 -3.00e+01 - 7.85e+01i     3.57e-01     8.40e+01

(Frequencies expressed in rad/seconds)

>> % ==> gyroscopic modes are correctly damped.

>> % Platform attitude servo-loop:
>> % The jacobian is defined as the DCgain of the transfer between the 4
>> % precession rates reference inputs and the 3 platform angular rates:
>> Jacob=dcgain(bf2(1:3,[4:7]))

Jacob =

    0.0120   -0.0120    0.0000    0.0000
    0.0000    0.0000    0.0120   -0.0120
    0.0035    0.0035    0.0035    0.0035

>> % Second integrators: platform angular positions and rates on outputs:
```

```
>> sint=ss(0,1,[1;0],[0;1]);
>> Gpv=minreal(append(sint,sint,sint,1,1,1,1)*bf2);

>> % CMGs guidance:
>> Gpv=Gpv*[eye(3),zeros(3);zeros(4,3) pinv(Jacob)];

>> % Attitude servo-loop bandwitdh (outer loop):
>> w=3;
>> bf=feedback(Gpv*diag([1,1,1,w,w,w]),eye(3),[4 5 6],[1 3 5]);
>> damp(bf)

       Eigenvalue          Damping     Frequency

 -3.08e+00                 1.00e+00     3.08e+00
 -3.25e+00                 1.00e+00     3.25e+00
 -3.25e+00                 1.00e+00     3.25e+00
 -2.84e+01 + 3.69e+01i     6.09e-01     4.66e+01
 -2.84e+01 - 3.69e+01i     6.09e-01     4.66e+01
 -2.84e+01 + 3.69e+01i     6.09e-01     4.66e+01
 -2.84e+01 - 3.69e+01i     6.09e-01     4.66e+01
 -5.99e+01                 1.00e+00     5.99e+01
 -2.84e+01 + 7.80e+01i     3.43e-01     8.30e+01
 -2.84e+01 - 7.80e+01i     3.43e-01     8.30e+01
(Frequencies expressed in rad/seconds)

>> %  ==> all the modes are correctly damped.
>> figure
>> step(bf([1 3 5],[4 5 6]),2);
```

The step response is shown on Figure 3.6.

3.8 Example 5: SpaceRoboticArm.m

This file describes a spatial vehicle fitted with a manipulator (arm) with 6 degrees of freedom and a high aspect ratio, as shown on Figure 3.7. We suppose that the segments are all rigid. The obtained model has 12 channels: 6 for the degrees of freedom of the hub and 6 for the manipulator. The files `Segment1.m`, `Segment2.m`, ..., `Segment6.m` describe the different segments. Each segment is composed of a main body (hub) and one appendage which is the next segment. The global variable `config` (6×1) can be used to set the angular configuration of the manipulator (same as in the previous example).

The following MATLAB® session (see also the script file `demoSTD.m` in the `STD_V1.3` sub-directory) shows how to use this file with the SDT.

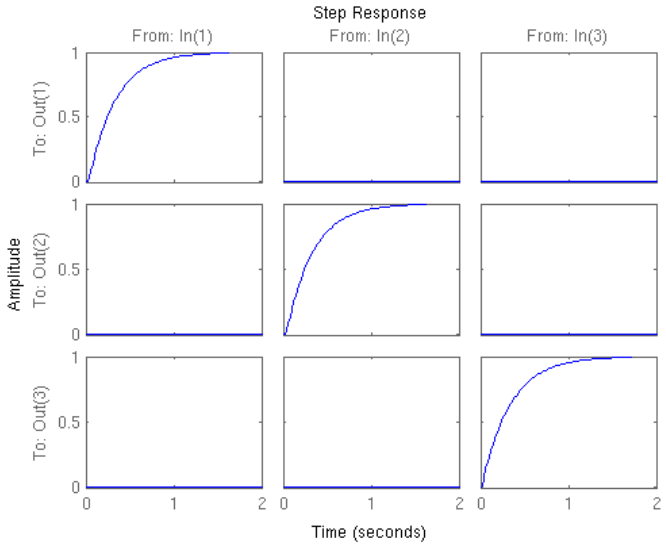

Figure 3.6: Step response of the CMG platform with attitude control, on the 3 axes

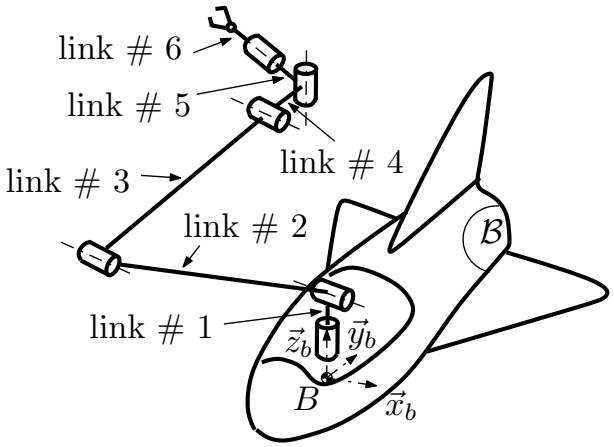

Figure 3.7: A manipulator arm fitted on a spatial vehicle

```
>> clear all,close all,
>> global config
>> mod=main('SpaceRoboticArm');
Angular configuration: 0   30  120    0    0    0
>> mod.liste_IOs  % The 12 channels (d.o.f. order).

ans =

Trans. X Main_body
```

```
Trans. Y Main_body
Trans. Z Main_body
Rot. X Main_body
Rot. Y Main_body
Rot. Z Main_body
Joint: Link#6/Link#5
Joint: Link#5/Link#4
Joint: Link#4/Link#3
Joint: Link#3/Link#2
Joint: Link#2/Link#1
Joint: Link#1/Main_body

>> mod.TotalCg    % Position of the global centre of mass.

ans =

    0.0270
   -0.3741
    1.4893


>> mod.DynamicModel

ans =

  d =
                u1           u2           u3           u4
   y1         1400  -6.311e-30   6.311e-30  -7.272e-31
   y2   -6.311e-30         1400   3.864e-46   4.547e-13
   y3    6.311e-30   3.864e-46         1400  -3.411e-13
   y4   -7.272e-31   4.547e-13  -3.411e-13   1.361e+04
   y5            0   3.088e-30  -2.132e-14      -57.02
   y6    3.411e-13   2.132e-14  -2.361e-30      -266.3
   y7            0           0           0      0.3375
   y8    8.472e-16      -32.74       -18.9       202.1
   y9   -4.216e-15        25.2      -43.65      -241.7
   y10  -1.002e-13       598.8       -1037       -4377
   y11  -5.978e-14        1500       523.7   -1.07e+04
   y12        523.7        37.8           0      -266.3

                u5           u6           u7           u8
   y1            0   3.411e-13           0   8.472e-16
   y2    3.088e-30   2.132e-14           0      -32.74
   y3   -2.132e-14  -2.361e-30           0       -18.9
```

```
      y4       -57.02      -266.3      0.3375       202.1
      y5     1.141e+04        1269   4.133e-17       15.53
      y6         1269        2934  -5.646e-17      -26.91
      y7     4.133e-17  -5.646e-17      0.3375   2.067e-17
      y8        15.53      -26.91   2.067e-17       32.09
      y9        19.45       11.23     -0.3375   2.562e-15
      y10       223.7       129.2     -0.3375   2.562e-15
      y11       57.02       225.4     -0.3375      -208.9
      y12        1269        2267  -5.646e-17      -27.79

                   u9         u10         u11         u12
      y1    -4.216e-15  -1.002e-13  -5.978e-14       523.7
      y2          25.2       598.8        1500        37.8
      y3        -43.65       -1037       523.7           0
      y4        -241.7       -4377   -1.07e+04      -266.3
      y5         19.45       223.7       57.02        1269
      y6         11.23       129.2       225.4        2267
      y7       -0.3375     -0.3375     -0.3375  -5.646e-17
      y8     2.562e-15   2.562e-15      -208.9      -27.79
      y9         30.01       385.8       225.1       11.91
      y10        385.8        7804        3983       145.3
      y11        225.1        3983    1.088e+04       265.9
      y12        11.91       145.3       265.9        2464

Static gain.
>> % This model is static since all links are rigid
```

3.9 Example 6: `Spacecraft1u.m`

This version of the toolbox (SDT Version 1.3) also allows to take parametric variations of parameters into account, and more particularly the main structural parameters such as the mass, the inertias, the geometry (position of the centre of mass and position of the anchoring points), the angular frequencies, dampings and modal participation factors. To sum up, all the parameters defined by the fields of type `double` of the structures `MB` and `SA{i}` (see table 3.1 and 3.2) except the angular parameters (`SA{i}.TM`, `SA{i}.pivotdir` and `SA{i}.angle`) can be defined with parametric uncertainties. The varying parameters can be defined by the function `ureal` from the Robust Control Toolbox (RCT). Then, the output variables of the SDT are uncertain matrices or state-space representations and are fully compatible with the analysis tools of the RCT.

The file `Spacecraft1u.m` is based on example 2 and adds parametric variations on dynamics and geometric parameters of the hub and of the appendage. The following

MATLAB® session shows how to perform a sensitivity analysis of the frequency response of the inverse dynamics model (see Figure 3.8).

```
>> clear all,close all,
>> mod=main('Spacecraft1u');
>> mod.TotalCg                % the position of the global centre of mass.

ans =

  Uncertain matrix with 3 rows and 1 columns.
  The uncertainty consists of the following blocks:
    MB_m: Uncertain real, nominal = 100, variability = [-10,10]%, 1 occurrences
    SA1_CGx: Uncertain real, nominal = 1, variability = [-10,10]%, 1 occurrences
    SA1_Px: Uncertain real, nominal = 0, range = [-0.01,0.01], 1 occurrences
    SA1_Py: Uncertain real, nominal = 1, variability = [-10,10]%, 1 occurrences
    SA1_m: Uncertain real, nominal = 50, variability = [-10,10]%, 1 occurrences
    SA2_CGy: Uncertain real, nominal = 1, variability = [-10,10]%, 1 occurrences
    SA2_Py: Uncertain real, nominal = -1, variability = [-10,10]%, 1 occurrences
    SA2_Pz: Uncertain real, nominal = 0, range = [-0.01,0.01], 1 occurrences
    SA2_m: Uncertain real, nominal = 20, variability = [-10,10]%, 1 occurrences

Type "ans.NominalValue" to see the nominal value, "get(ans)" to see all properties,
 and "ans.Uncertainty" to interact with the uncertain elements.

>> Mi=mod.InverseTotalModel;  % inverse dynamic model
>> figure
>> sigma(Mi)
>> % sensitivity of the frequency domain response to parametric variations.
```

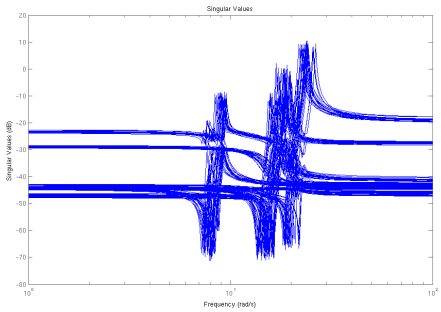

Figure 3.8: Frequency responses (Singular Values) of the inverse dynamics model for several parameter configurations

3.10 Exercise

Statement

Using the data file `SpaceRoboticArm.m` (see section 3.8) describing a chaser spacecraft with a robotic arm and the data file `Spacecraft1.m` (see section 3.4) describing a spacecraft with two symmetrical flexible appendages, the objective of this exercise is two build the data file to analyse the dynamic behavior of the chaser holding the spacecraft considered as a debris (see Figure 3.9). The interface point P_7 between the arm end effector and the debris is characterized by:

- its coordinate vector in the frame $\mathcal{R}_{a_6}$ attached to the end effector (`Segment6.m`): $\left[\overrightarrow{O_6P_7}\right]_{\mathcal{R}_{a_6}} = [0 \quad 0 \quad 0.55]^T$,
- its coordinates vector in the frame $\mathcal{R}_{a_7}$ attached to the debris spacecraft (`Spacecraft1.m`): $\left[\overrightarrow{O_7P_7}\right]_{\mathcal{R}_{a_7}} = [-0.3 \quad 0 \quad -1.2]^T$.

The direction cosine matrix between frames $\mathcal{R}_{a_6}$ and $\mathcal{R}_{a_7}$ (see nomenclature) is:

$$T_{a_6a_7} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad \text{with: } \theta = 80\,deg\ .$$

Solution

Firstly, the data file `Segment6.m` must be completed by the following instructions to take into account that `Segment6` (the end-effector) hold an appendage described in the data file `Spacecraft1.m`

```
nappend = 1;
%=========================================================================
% APPENDAGE 1 : SA{1}
% DEBRIS SPACECRAFT
%-------------------------------------------------------------------------
SA{1}.Name = 'DEBRIS';      % Name of next appendage: the debris
SA{1}.P = [0;0;0.55];       % position of connection point P7 in (O6,X,Y,Z) (m)

SA{1}.TM = [ cos(80*pi/180) 0  sin(80*pi/180)    % T_A6A7
              0             1      0
             -sin(80*pi/180) 0  cos(80*pi/180)];

SA{1}.pivot = 0;            % cantilevered joint

SA{1}.filename = 'Spacecraft1';
```

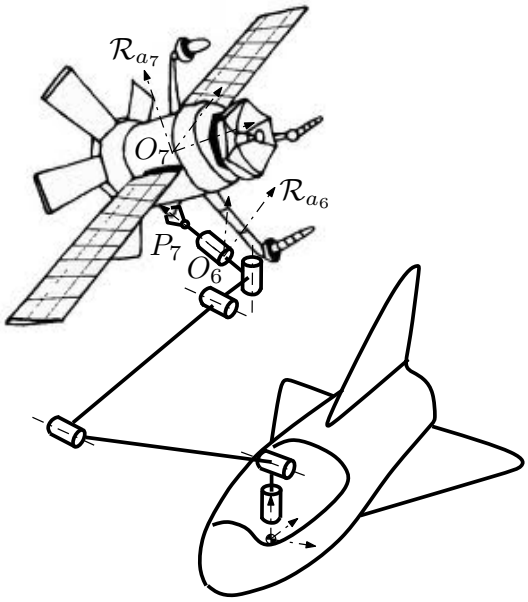

Figure 3.9: A manipulator arm fitted on a spatial vehicle holding a debris spacecraft.

Secondly, the data file `Spacecraft1.m` must be completed by the following instructions to take into account the new reference point P_7 (instead of the previous one: O_7):

```
% DEBRIS case: taking into account P7_O7:
MB.cg = MB.cg + [0.3;0;1.2];
SA{1}.P = SA{1}.P + [0.3;0;1.2];
SA{2}.P = SA{2}.P + [0.3;0;1.2];
```

Then the model of the assembly: chaser + robotic arm + debris can be computed using:

```
>> clear all,close all,
>> global config
>> mod=main('SpaceRoboticArm');
```

EMPTY PAGE

References

[1] M. Rognant, Ch. Cumer "Configurations à 6 actionneurs gyroscopiques - Etude bibliographique", ONERA, RT 1/22821 DCSD, Janvier 2016

[2] D. Alazard, Ch. Cumer and K. Tantawi "Linear dynamic modeling of spacecraft with various flexible appendages and on-board angular momentums", 7th International ESA Conference on Guidance, Navigation & Control Systems , 01-05 Jun 2008, Tralee, Ireland

[3] G, Nicolas, D. Alazard, Ch. Cumer and C. Charbonnel Journal of Dynamic Systems Measurement and Control, vol. 136 (num. 2), ISSN 0022-0434, 2014

[4] Ch. Cumer, D. Alazard "Itérations design mécanique/commande - Tâche 1 : Modélisation mécanique classique", RAv 1/21116 DCSD - Avril 2013

[5] D. Alazard "Reverse engineering in control design", Wiley, 2013.

[6] Newton–Euler equations: http://en.wikipedia.org/wiki/Newton-Euler_equations

EMPTY PAGE

Appendix A

Inline help

A.1 `help` of function `main.m`

```
MODEL = main(FILENAME) compute the model MODEL of the
  spacecraft described in the file FILENAME.m

MODEL is a structure:
  MODEL.TotalMass: total mass of the spacecraft,
  MODEL.TotalCg: 3x1 coordinate vector of the global centre of mass
                 (CGtot) in the reference frame attached to the main body
                 (0, X, Y, Z),
  MODEL.InverseTotalModel: [(6+NBPIVOTS)x(6+NBPIVOTS) ss] inverse dynamic
                  model between inputs:
                   * 6 dof external force-torque applied on the main body,
                   * NBPIVOTS torques applied inside the NBPIVOTS
                     pivot joints between main body and appendages,
                  and outputs:
                   * 6 dof linear-angular accelerations of main body,
                   * NBPIVOTS relative angular accelerations of
                   appendages w.r.t main body around pivots axis.
                   This model is written at point CGtot in frame (CGtot,
                   X, Y, Z).
  MODEL.DynamicModel: [(6+NBPIVOTS)x(6+NBPIVOTS) ss] direct dynamic model of
                  the spacecraft (the inverse of the previous one).
  MODEL.liste_IOs: Description and ordering of the (6+NBPIVOTS) inputs
                  (outputs) used in the dynamic model.

  Reference:
      Alazard, D., Cumer, C., and Tantawi, K.,
      \Linear dynamic modeling of spacecraft with various flexible
       appendages and on-board angular momentums".
```

```
In Proceedings of the 7th International ESA Conference on Guidance,
Navigation & Control Systems
Tralee, Ireland, 1-5 June 2008
```

A.2 help of function translate_dynamic_model.m

```
TDM = translate_dynamic_model(vec_A,vec_B,DM) translate the direct
dynamic model from point A to point B in the same reference frame.
  Inputs:
   * vec_A: 3x1 coordinate vector of point A,
   * vec_B: 3x1 coordinate vector of point B,
   * DM:  dynamic model at point A.
  Output:
   * TDM: dynamic model at point B.

It is assumed that the first 6x6 block of DM represent the dynamic model
between the 6 dof acceleration vector and the 6 dof external force
vector.
```

A.3 help of function rotate_dynamic_model.m

```
DM_OUT = rotate_dynamic_model (DM_IN , ANGLE, AXIS)
computes the dynamic model after a rotation of ANGLE (deg)
around the axis AXIS :
  * DN_IN:  dynamic model of a body in a frame R,
  * ANGLE:  rotation angle (deg),
  * AXIS:   3 components in the frame R of the unitary vector
    along the rotation axis,
  * DN_OUT: dynamic model of the rotated body in the frame R.

It is assumed that the first 6x6 block of DM_IN represent the dynamic model
between the 6 dof acceleration vector and the 6 dof external force
vector.
```

A.4 help of function kinematic_model.m

```
 JACOB=kinematic_model(vec_A, vec_B) calculates the kinematic model JACOB
of a body between two points A and B
  Inputs:
   * vec_A: 3x1 coordinate vector of point A in a given frame,
   * vec_B: 3x1 coordinate vector of point B (in the same frame).
```

```
Output:
 * JACOB: 6x6 kinematic model (projected in the same frame).

                | eye(3)      (*AB) |
        JACOB = |                   |
                |zeros(3)     eye(3)|
```

A.5 help of function antisym.m

```
MAT=antisym(VEC) computes the antisymmetric matrix MAT associated
     with a vector VEC.
     if VEC=[x y z]' then MAT=[0 -z y;z 0 -x;-y x 0].
```

EMPTY PAGE