SPACE PROPULSION 2016 - MARRIOTT PARK HOTEL, ROME, ITALY / 2–6 MAY 2016

# PROGRESS IN COUPLED SIMULATION PROPULSION SYSTEM AND VEHICLE WITH ESPSS SATELLITE LIBRARY

**Christophe R. Koppel[(1)], Francesco Di Matteo[(2)], Johan Steelant[(3)], José Moral[(4)], Javier Vila Vacas[(5)]**

*[(1)] KopooS Consulting Ind., 57 rue d'Amsterdam 75008 Paris, France, Christophe.Koppel@kopoos.com*
*[(2) (3)] ESTEC, Keplerlaan 1, P.O. Box 299, 2200 AG Noordwijk, The Netherlands, Francesco.Di.Matteo@esa.int, Johan.Steelant@esa.int*
*[(4) (5)] Empresarios Agrupados, c/Magallanes 3, 28015 Madrid, Spain, frj@empre.es, jvv@empre.es*

**KEYWORDS**: simulation propulsion vehicle system ESPSS EcosimPro…

**ABSTRACT**

The paper documents the progress realized during the 4th phase of the work performed on the ESPSS library (European Space Propulsion System Simulation) running on the existing platform EcosimPro® for the implementation and validation of the coupled simulation of the propulsion system and vehicle.

This includes a significant update of the Satellite platform propulsion system modelling and the coupling is performed by updating the Fluid flow, Tanks and Combustion chamber components and taking into account the Archimedes pressure coming from acceleration and rotations given by the vehicle or by any perturbation forces.

EcosimPro® is an object-oriented visual simulation tool capable of solving various kinds of dynamic systems represented by writing equations and discrete events. It can be used to study both transients and steady states [R1,2]. The object oriented tool, with the propulsion library ESPSS for example, allows the user to draw (and to design at the same time) the propulsion system with components of that specific library with tanks, lines, orifices, thrusters, tees. The user enhances the design with components from the thermal library (heaters, thermal conductance, radiators), from the control library (analogue/digital devices), from the electrical library, etc.

The paper presents the progress in several new features added to the Satellite library as well as some new components for performing full attitude control of a platform. A new powerful compact equation discovered in the course of the work for solving in an elegant manner the Archimedes pressure coming from combined acceleration and rotation in the most general case (non collinear) will be presented and discussed. The paper presents the progress in modelling of propulsion system performed in order to check the implementation of the new components especially the components dealing with the effects of the mission on the propulsion sub-system. In particular, the accelerations generated by thrusters, reaction wheels and other spacecraft components have an influence on the fluid-dynamic behaviour of the propulsion system. Moreover, the propellant tanks fill level influences the spacecraft's centre of gravity and moments of inertia, which can in turn have an effect on the pressure and flow conditions in fluid lines feeding the thrusters. The use of the ESPSS Satellite library for being able to model some interactions between the AOCS and the propulsion system is presented.

## 1. INTRODUCTION

In the frame of the 4th phase of the work performed on the version 3.2 of the ESPSS library (European Space Propulsion System Simulation) [R 5] within the existing tool EcosimPro® [R 9] the implementation and validation of the "coupled simulation of the propulsion system and vehicle" is presented hereafter. This work comes after a first step performed in the previous phase [R 3] regarding the "evolutionary behaviour of components", where a specific Satellite library was set up for the simulation of the Attitude and Orbit Control System's (AOCS) effects on the propulsion sub-system since it includes the flight dynamic (orbit and attitude) capabilities for an entire spacecraft with orbital and attitude perturbations. Thus the present work includes a significant update of this spacecraft platform propulsion system modelling and the coupling is performed by updating the Fluid flow, Tanks and Combustion chamber components and taking into account the Archimedes pressure coming from acceleration and angular rates given by the vehicle or by any perturbation forces.

The paper presents several new components for

performing full attitude control of a platform.
A new powerful compact equation discovered in the course of the work for solving in an elegant manner the Archimedes pressure coming from combined acceleration and angular rates in the most general case (non collinear) is presented in and discussed in Chapter 4 after some generalities and overview of the Satellite library next.

## 2. ESPSS background

EcosimPro® is a Physical Simulation Modelling tool developed for ESA by Empresarios Agrupados Internacional (Spain) since 1989. EcosimPro® was a precursor and now with its 27 years of careful growing, it belongs to the last generation of the common engineering tools after computer aided design (CAD) and integrated engineering analysis tools available on classical laptops. The kernel of EcosimPro® is an expert solver of all the equations set in the different components of a system. Thanks to such expert solver, the tool allows to manipulate components like objects that can be independently further developed with more sophisticated equations. EcosimPro® is based on a visual simulation tool for solving simple and complex physical processes that can be expressed in terms of equations (including ordinary differential equations and differential-algebraic equations) and discrete events.
Practically, the modelling of physical components is based on a basic “EcosimPro language” (EL), an object-oriented programming language which is very similar to other conventional programming languages (Basic, C++) but is very powerful to write any equations and differential equations for modelling continuous and discrete processes. EcosimPro employs a set of libraries containing various types of components (mechanical, electrical, pneumatic, hydraulic, etc…) which can be interconnected to model complex multi-domain dynamic systems. The ESA European Space Propulsion System Simulation (ESPSS) is a set of EcosimPro® libraries written to model all aspects of a functional propulsion system. As a tool ESPSS is relying on 1D flow equations, thermodynamic relationships and real fluid properties, there is no need for fudge factors, therefore the results of the simulation could be considered as general as long as the flow is one-dimensional and homogeneous (either as mono-phase, or two-phase state or as a mixture).

## 3. Satellite Library

The Satellite library contains the main components needed to build a spacecraft AOCS with the major interactions and perturbations that occurs in flight [R 6],[R 7]: perturbations from Moon and Sun (mainly corrected by the use of thrusters for the North-South station keeping of a GEO satellite), and perturbations from the planets also added with the general N-body gravitational force equations (allowing to perform interplanetary travels, flyby, etc). Earth flatness or so called "J2" perturbations (to be used for the heliosynchronous satellites), Sun pressure interaction with the solar arrays of the spacecraft, Drag forces from upper Earth atmosphere, Gravity gradient torques, Magnetic torques from the interaction of the spacecraft magnetic moment with the Earth magnetic field are altogether also described by equations added to the library.
Moreover, a control of the spacecraft attitude can be added (using the existing components of the existing CONTROL library) for performing an Earth Pointing with a set of actions on the reaction wheels or on a set of thrusters.
Eventually, Figure 1 shows the palette with the set of the components..

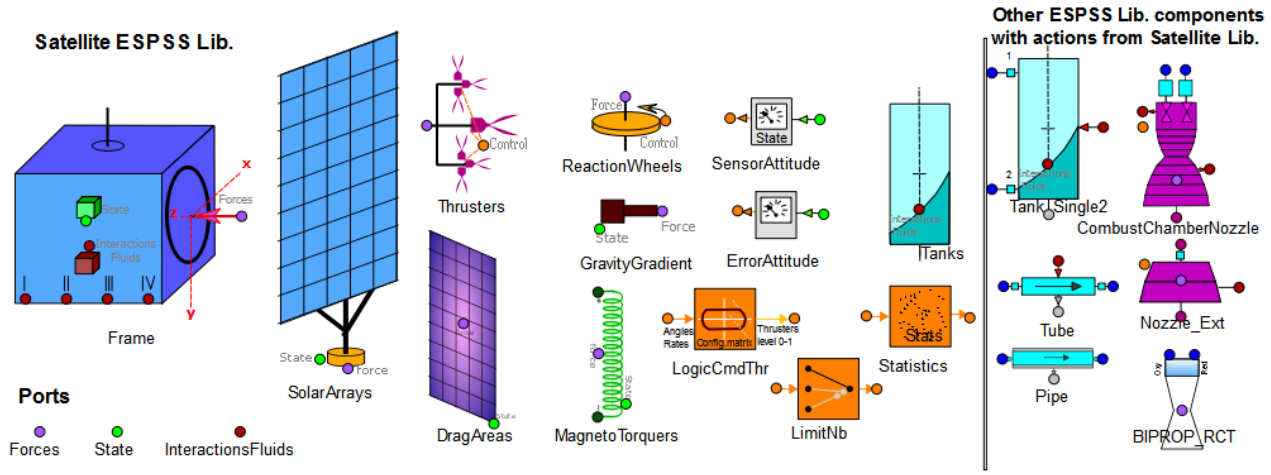



*Figure 1 Palette of components ESPSS's Satellite library of with coupled components in other ESPSS libs*

Three ports have been designed in the Satellite library for the transfer of information between the objects (i.e. components) of the library:

**Force port**: this is a multifunctional port for allowing inputs from a set of thrusters, Reaction Wheels (RW), Solar Arrays (SA), Drag Area (DA), MagnetoTorquers (MT) and Gravity Gradient (GG) with the port directions set at IN for the spacecraft frame, OUT for all other components. The variables of the port are of type SUM in order to automatically account for all mass flow rates, forces, moment, angular momentum and power coming from all connected object or components.

**State port**: this is a multipurpose port for allowing 3D visualization and the attitude and orbit data for further inputs for the independent objects like SA, DA, GG, Tanks and for the control. The port directions is set at OUT for the spacecraft frame, IN for all other objects or components.

**InteractionsFluids port**: this port allows to communicate interactively to the Frame component: the mass, the location of the Centre Of Mass (COM) and the inertia matrix of each liquid fluid in the tanks components.

### 3.1. Frame component

This major component of the Satellite library has been slightly improved in the 4th phase by adding the perturbation from the other 8 planets in order to get the possibility to make interplanetary flights, and the number of "InteractionsFluids ports" has been increased to 5 allowing to simulate many kind of propulsion systems (launcher stage, satellites propulsion platform, etc.).

The description of the Frame component is recalled here for completeness.

This component, representing a spacecraft itself, is in charge of solving the flight dynamic equations with input data of the initial orbit and with non-gravity (i.e. non-body) forces vectors coming from the thrusters (including RCT, combustion chamber nozzle or nozzle extension from other ESPSS lib), SA, DA. The equations for perturbation forces coming from Moon, Sun and other planets N-body gravity are directly set inside this component.

The first main equation of the component is based on the momentum equation (1) as follow:

$$\frac{d\vec{\pi}}{dt} = \vec{\Pi} \qquad (1)$$

with

$$\vec{\pi} = \begin{pmatrix} \vec{r} \\ \vec{v} \\ m \end{pmatrix} \quad \vec{\Pi} = \begin{pmatrix} \vec{v} \\ -GM_{focus}\,\vec{r}/r^3 + \sum\left(\vec{F}_{thrust} + \vec{P}_{perturb}\right)/m \\ \sum -thrusters\_mass\_flow \end{pmatrix}$$

where $\vec{v}$ is the velocity vector of the spacecraft frame with respect to the Earth Centred Inertial frame (ECI), GM the gravitational constant time the mass of the focus body (Earth), $\vec{r}$ the radius vector from the focus body centre, $\vec{F}$ the resultant thrust vector, applied on the spacecraft COM, in ECI, $\vec{P}$ the perturbation and interaction force vector in ECI and *m* the instantaneous mass of the spacecraft that is corrected dynamically by the thrusters consumption if any.

The second main equation of this component is based on the dynamic equation (2) for the attitude around the current COM as follow:

$$\frac{d\vec{H}}{dt} + \vec{\Omega}\wedge\vec{H} = \vec{M}_{Control} + \vec{M}_{Perturbation} \quad (2)$$

with

$$\vec{M}_{Control} = \left\{ M_{thruster} - \sum_i \frac{d\vec{H}_{RWi}}{dt} \right\}$$

$$\vec{M}_{Perturbation} = \left\{ M_{perturb} - \sum_i \vec{\Omega}\wedge\vec{H}_{RWi} - \sum_i \vec{\Omega}\wedge\vec{H}_i - \sum_i \frac{d\vec{H}_i}{dt} \right\}$$

with $\vec{\Omega}$ the instantaneous rotation with respect to the inertial orientation frame of the spacecraft.

EcosimPro® being an object oriented tool, the source terms come from the torques, angular momentum from all connected components at the Force port of the Frame component, i.e. any of : thrusters, MT, GG and RW, SA. In equation (2), $\vec{H}_{RWi}$ is the angular momentum with respect to the spacecraft of the reaction wheels and $\vec{H}_i$ the one of the other mobile parts but with all their coordinates and derivative written in the spacecraft axis. The angular momentum $\vec{H}$ takes into account the possible "evolutionary behaviour" of the fluid in the tanks through the inertia matrix update. The attitude angles of the spacecraft must be known in order to properly set the local data like the thrust with respect to the spacecraft frame into the ECI frame. Those can be given by the Cardan angles (yaw, pitch, roll) of the spacecraft axis with respect to the orbital frame. The attitude angles of the orbital frame can be given by the Euler angles (precession, nutation, spin) with respect to the inertial frame ECI. But in order to ensure robustness in the solutions, the equations dealing with the attitude angles are based on the quaternion theory. The general equation of the quaternion (3) is to be solved simultaneously with

the other previous equations.

$$\frac{dQ}{dt} = +\frac{1}{2}[Q].\Omega \qquad (3)$$

The integration produces at each time t the quaternion $Q(t)$ that enables to retrieve (with suitable conversion matrixes) the attitude angles without any risk of singularities with unit quaternion.
As the quaternion of the spacecraft with respect to the orbital frame is available (given by the state port) it can be used in the external control loop of the attitude of the spacecraft for an Earth pointing command by setting to zero its imaginary components.

### 3.2. Solar arrays component

For the current release of the library, either a fixed SA on the spacecraft body or an automatic orientation of the SA with respect to the Sun is considered. The component is linked to the Frame component (state port) in order to get the spacecraft to Sun vector and other information.
The classic equations of Sun pressure interaction on the solar array are used in this component with input coefficients of reflection, specular and absorption (the diffuse coefficients being complement to 1). During eclipses of Sun, those interactions are null.
The output of the solar arrays are the 3D forces due to the Sun pressure, their moments and the electrical power produced by the solar cells according to the input data of the efficiency of the solar cells.
As for other components, the solar array component is written in terms of vector to allow a number of SA (up to 12 for instance) that are all described in term of location, orientation of the axis of rotation and orientation of the canonical normal to the solar cells, size.

### 3.3. Thrusters component

When activated, this component, based on classic equations, outputs the thrust vector with respect to the spacecraft, the moment induced by the thrusters' forces and the mass flow rates (for mono propellant as well as for bi-propellant). In case of electric propulsion, the electric power consumption is provided as well.

### 3.4. Reaction wheels component

Once activated this component, based on classic equations, outputs the angular momentum vector with respect to the spacecraft as well as the electric power consumption of all reaction wheels considered. This component has been improved in the 4th phase by adding equations for the operational limits in terms of maximum torque and angular rate or rpm limits.

### 3.5. Drag array component

This component is a new component of the 4th phase. It outputs the drag force coming from the dynamic pressure in flight. The Earth atmosphere type is taken from "ESA/space agencies/industry space environment standard ECSS-E-ST-10-04C" with several cases depending on the Sun activity.

### 3.6. Gravity gradient component

This component is also a new component of the 4th phase. Based on the input from the state port coming from the spacecraft frame component and the inertia matrix $[Inertia]$ of the whole body (including fluids), this component equation (4) provides the moment vector with respect to the spacecraft without any forces (pure torque).

$$\begin{aligned} \vec{M} &\approx \frac{3GM_{earth}}{R^3}\int_{body}(\vec{r}\cdot\vec{e}_{earth})\vec{r}\times\vec{e}_{earth}dm \\ &\approx \frac{3GM_{earth}}{R^3}\vec{e}_{earth}\times([Inertia]\cdot\vec{e}_{earth}) \end{aligned} \qquad (4)$$

where *R* is the distance from COM to Earth centre, $\vec{e}_{earth}$ the unit vector Earth, and $\vec{r}$ the local vector from the COM.

## 4. Tanks component

The classic equation for Archimedes' pressure $dp = -\rho.\gamma.(z - z_0)$ was already taken into account in Flow1D tank model of the previous release of ESPSS [R 5], but this was too restrictive. In space, the spacecraft is rotating around its COM (and some torques are produced by thrusters and reaction wheels, or windmill effect by the solar pressure on the SA). Further particular forces e.g. originating from thrusters or solar pressure on the SA, produce acceleration. Therefore a better approach for the Archimedes' pressure needs to take into account the effect of the acceleration and the effect of the rotations even when not collinear.

### 4.1. A new equation for the Archimedes pressure

First, one notes that with $\vec{a}$ the acceleration of the COM in the inertial frame; $m.\vec{g}$ all the body forces

(which does not include the surface forces $\vec{F}$ like the reaction to the gravity, the thruster forces, the Sun pressure force, etc.); the acceleration due to surface forces is given by (5):

$$\vec{a} - \vec{g} = +\vec{F}/m \quad (5)$$

The Navier-Stokes equation is the Newton's Law of momentum " $\vec{a}_{abs} = \left(\vec{g} + \vec{F}/m\right)$ ", adapted for Newtonian fluids (with a control volume fixed in the non-inertial space considered) where $\vec{a}_{abs}$ is the absolute acceleration in an inertial frame. Hence this absolute acceleration can be developed as in equation (6):

$$\begin{aligned}&\vec{a} + 2\vec{\Omega}\times\vec{V}_{rel} + \vec{\Omega}\times\left(\vec{\Omega}\times\vec{r}\right) + \frac{d\vec{\Omega}}{dt}\times\vec{r} + \frac{\partial\vec{V}_{rel}}{\partial t} + \vec{\nabla}\,\tfrac{1}{2}\vec{V}_{rel}^{\,2} \\ &-\vec{V}_{rel}\times\left(\vec{\nabla}\times\vec{V}_{rel}\right) = \vec{g} - \frac{\vec{\nabla}p}{\rho} + \mu/\rho\left(\vec{\nabla}\cdot\vec{\nabla}\vec{V}_{rel} + \tfrac{1}{3}\vec{\nabla}(\vec{\nabla}\cdot\vec{V}_{rel})\right)\end{aligned} \quad (6)$$

with $\vec{\Omega}$ the instantaneous rotation vector around COM; $\vec{r}$ and $\vec{V}_{rel}$ the location and velocity of the fluid particle wrt the non-inertial frame; $\mu$ the viscosity of the fluid.
Considering a static behaviour in the non-inertial frame i.e. $\vec{V}_{rel}$ =0, it remains for the point $\vec{r}$ :

$\vec{a} + \vec{\Omega}\times\left(\vec{\Omega}\times\vec{r}\right) + \frac{d\vec{\Omega}}{dt}\times\vec{r} = \vec{g} - \frac{\vec{\nabla}p}{\rho}$ , thus for constant rotation ($\vec{\Omega}$=Cst), one gets in (7) a simple equation for the pressure gradient in the general case of non-collinear force and angular rotation:

$$\vec{\nabla}p = -\rho\left(\vec{a} - \vec{g} + \vec{\Omega}\times\left(\vec{\Omega}\times\vec{r}\right)\right) \quad (7)$$

*In the literature, it was not possible to find this kind of simple equation for the pressure gradient, so no references can be linked to equation (7).*

The pressure difference between two points is the integral of a scalar product $\vec{\nabla}p\cdot d\vec{s}$ :

$p_2 - p_1 = \int_{\vec{r}_1}^{\vec{r}_2} dp = \int_{\vec{r}_1}^{\vec{r}_2} \vec{\nabla}p\cdot d\vec{s}$ , and noting that the centripetal acceleration can be derived from a potential

$\vec{\Omega}\times\left(\vec{\Omega}\times\vec{r}\right) = -\vec{\nabla}\left(\tfrac{1}{2}\left(\vec{\Omega}\times\vec{r}\right)^2\right)$ see [R 8] ,

hence, if the fluid density $\rho$ is constant, one gets straightforward the integrated equation (8) for the Archimedes pressure difference in the general case of non-collinear force and angular rotation:

$$p_2 - p_1 = \left[-\rho\left(\vec{a} - \vec{g}\right)\cdot\vec{r} + \tfrac{1}{2}\rho\left(\vec{\Omega}\times\vec{r}\right)^2\right]_{\vec{r}_1}^{\vec{r}_2} \quad (8)$$

*Once again, it was not possible to find such kind of simple equation (8) for the Archimedes pressure difference in the literature; hence no references can be linked.*

### 4.2. Case of tanks and components of the Fluid_Flow_1D library

Inside the tanks, the free surface shall be considered for one of the points, hence the Archimedes pressure at point $\vec{z}_b$ with respect to the point $\vec{z}_{free}$ (a single point belonging to the free surface is enough) is given by:

$$p_b - p_{free} = \left[-\rho\,\vec{F}/m\cdot\vec{r} + \tfrac{1}{2}\rho\left(\vec{\Omega}\times\vec{r}\right)^2\right]_{\vec{z}_{free}}^{\vec{z}_b} \quad (9)$$

where $\vec{F}$ is the sum of all non-body forces acting on the COM, i.e. surface or contact forces: thrust, perturbation forces due to Sun pressure, drag, etc. and also including the reaction force to the weight for the case of simulation on ground.
The free surface of the tank is given by a 3D discretisation of the tank using the equation (9) for the location of the liquid phase until the volume reaches the current given volume of liquid.
The shape of the free surface is a paraboloid having an axis parallel to $\vec{\Omega}$. Its deviation from $\vec{\Omega}$ located at the spacecraft COM depends on the strength and direction of the force $\vec{F}/m$.
For the tubes, the equation (8) is directly used. It is to be highlighted that in this equation $\vec{r}$ is the vector between the point *M* considered in the reference frame vector $\overrightarrow{OM}$ and the real current COM.
Because in a dynamic system the 3 vectors $\vec{F}/m$, $\vec{\Omega}$ and the vector $\overrightarrow{OCOM}$ (from the reference origin O to the COM) may change every time, a very powerful technique for fast integration of the system dynamic versus time is to use derived variables $\vec{A}ccel$ , $\vec{O}mega$ , $\overrightarrow{OCom}$ defined by

differential equations from the 3 vectors $\vec{F}/m$, $\vec{\Omega}$ and $\overrightarrow{OCOM}$. For example in (8) one uses $\vec{O}mega$ derived from $\vec{\Omega}$ by $\frac{d\vec{O}mega}{dt} = \frac{\vec{\Omega} - \vec{O}mega}{tau}$ where *tau* is a time constant. Eventually, one uses filtered variable which can be justified physically by some structure time response. And the equation used for the Archimedes pressure is in (11) :

$$p_2 - p_1 = \left[ \begin{array}{l} -\rho.\vec{A}ccel \cdot \left(\overrightarrow{OM} - \overrightarrow{OCom}\right) \\ \quad + \tfrac{1}{2}\rho\left(\vec{O}mega \times \left(\overrightarrow{OM} - \overrightarrow{OCom}\right)\right)^2 \end{array} \right]_{\vec{r}_1}^{\vec{r}_2} \quad (11)$$

The 3 dynamic vectors $\vec{A}ccel$, $\vec{O}mega$, $\overrightarrow{OCom}$ are set as global for every components of ESPSS, in the Fluid_Flow_1D library, so that there are no need to add any links between the tubes and the frame for being able to use the equation (11).

## 5. LogicCmdThr component

This component is a new component developed during the 4th phase. It allows to perform the active stabilisation with thrusters. Its input data are the force, location and orientation of the thrusters and some data regarding the stabilisation accuracy (angular rate limit and attitude angles limit).

The logic for the stabilisation follows the principle of the Dead-zone control with hysteresis: for each axis, the angular rate drives the evolution of the attitude angle. Once a switching limit (that is a combination of angle and angular rate) is reached, a torque shall be executed in order to control the attitude by reversing the sign of the angular rate. This kind of control is sketched in the phase space of one axis in Figure 2: without any perturbations, the limit cycle is the loop ABCDA indefinitely.

The command of the thrusters is performed with $[matCM]$, a rectangular configuration matrix (its columns are the torque components produced by each thruster activated at 100%) according to:

$$\vec{T}orque = [matCM]\vec{t}hrusterslevel$$

So, the pseudo-inverse of the rectangular configuration matrix allows selecting each thruster's level for a given set of torques needed to be performed. Because the thrusters can only produce thrust in one direction, a vector from the null space of the rectangular matrix $[matCM]$ shall be added in order to get only null or positive levels for each thruster:

$$[\vec{N}ullSpaceVectors] = [Identity] - [matCM\_pseudoInverse][matCM] \quad (12)$$

### 5.1. LimitNb component

Linked to the previous component, the equations of this component limit automatically the number of thrusters fired simultaneously and in case of need sequence the firing between the thrusters. Its input data are the maximum number of thrusters simultaneously fired and the period of sequencing.

### 5.2. Statistics component

Linked to the previous component, this component is exclusively used for transparently controlling the thrusters (directly from the input port to the output port) while keeping trace of the use of the different thrusters.

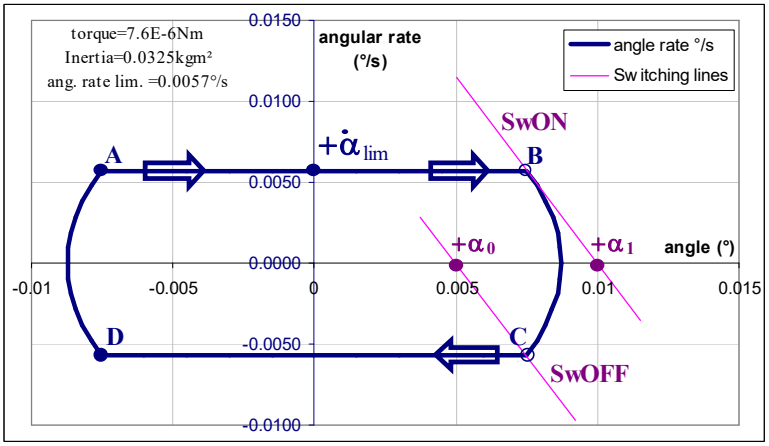



*Figure 2 Attitude Control Dead-zone with hysteresis*

## 6. Satellite library validation checks and use

The validation of the library including flight dynamic validation, attitude control validation and tank Archimedes pressure validation is described in previous paper [R 3],[R 10]. The new features of ESPSS have been checked carefully: the following example, part of those verifications, is dedicated to the test, checks and verification of the new features implemented into ESPSS, i.e. the coupling between propulsion and vehicle. It is derived from a launcher stage with bipropellant propulsion system, but it is more oriented for fast runs of long up to several hours of operations (within a run time of the order of 60s) with amplifications of the Archimedes effects.

### 6.1. Bipropellant Propulsion system with vehicle coupling by induced acceleration and angular rate

The model of the propulsion system considered is shown in Figure 3.

It comprises two main tanks: one for MMH on the top and one for NTO in the bottom, one pressure vessel for the Helium gas, a pressure regulator, several valves, non-return valve, tees and tubes, and a main rocket engine. The new features of the Satellite library allow the connection of the force vector from the main rocket engine to the Frame component, from each tank to the Frame component in order to set the vehicle mass, acceleration, inertia matrixes and angular rate (the moving mass of helium is neglected in this model). Those variables are taken into account for the Archimedes pressure in the tanks with right location of their free surface and into the whole manifold and finally at the main rocket engine injector. The main results of the simulation are shown in Figure 4 where the thrust phase of the main rocket engine lasts about 64 seconds (when the tanks are almost empty). The 3 filtered vectors $\vec{Accel}$ , $\vec{Omega}$ and $\overrightarrow{OCom}$ used for the fluid flow 1D components are shown in Figure 5. For this case the angular rate is maintained null thanks to the control loop with 3 axis reactions wheels added to the system for maintaining a fixed attitude of the space vehicle wrt the inertial frame (for this test, those RWs have very high limits to allow every wanted settings for their spin wrt inertial frame).

During the thrust phase, the mass decreases according the tanks depletion (coming from the rocket engine consumption through the tubing manifold of MMH and NTO), and after the thrust phase, the acceleration cancels.

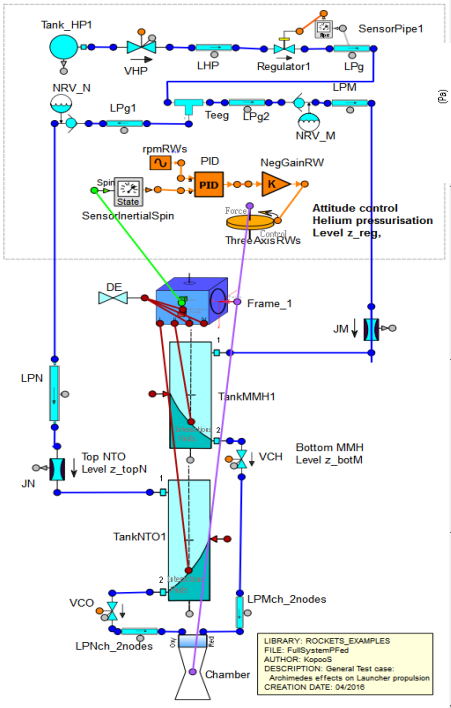



*Figure 3 Launcher propulsion system, propulsion coupling with acceleration, angular rate, COM*

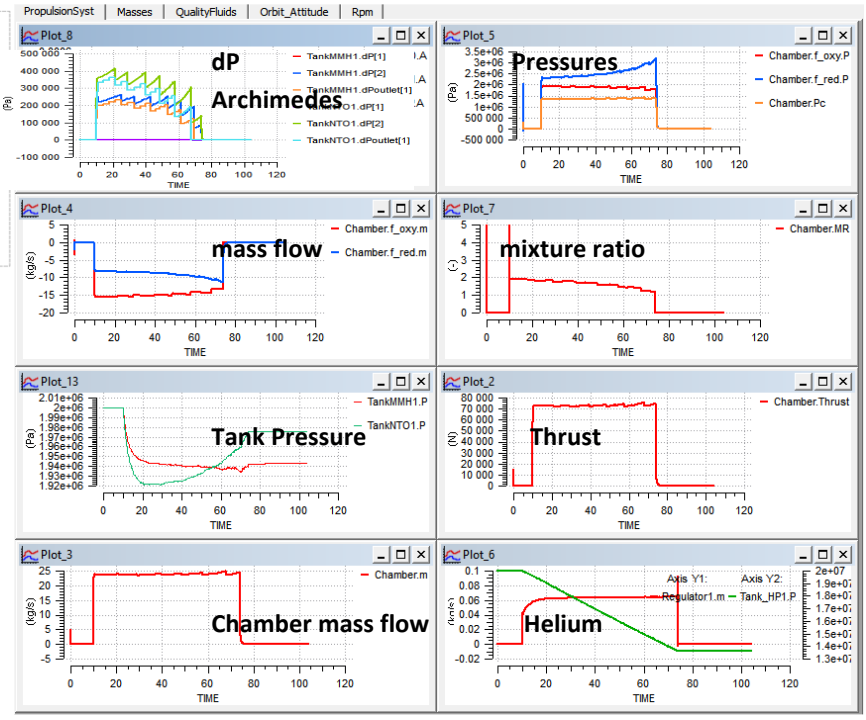



*Figure 4 Launcher propulsion parameters: Archimedes pressure, Injection pressure, mixture ratio, Thrust, etc...*

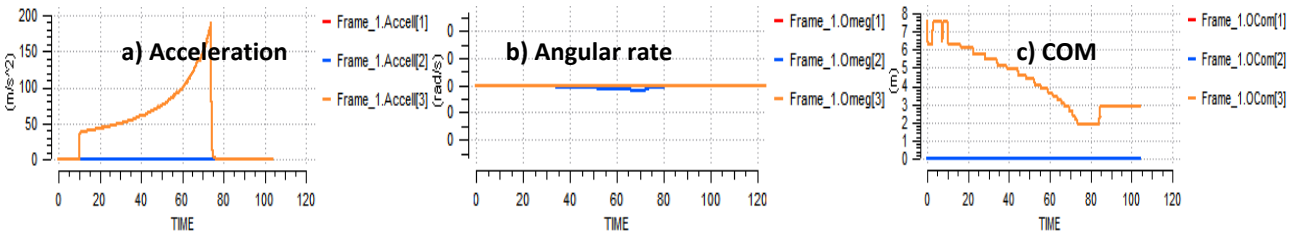



*Figure 5 Launcher system parameters: Acceleration, Angular rate and launcher centre of mass*

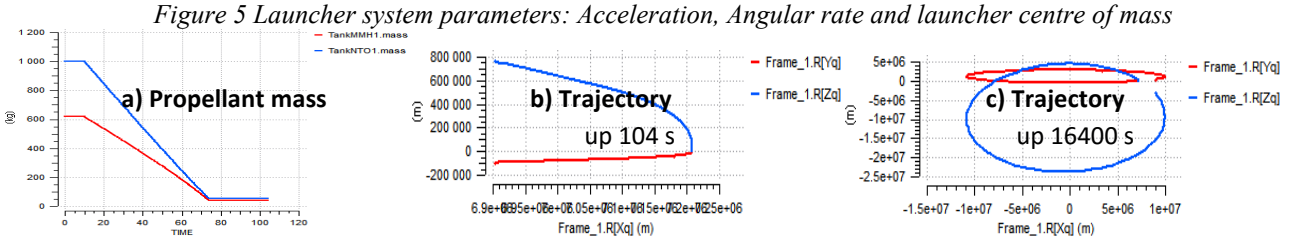



*Figure 6 Launcher mass and orbit at 104 s and at 16400 s*

A small jump in the axial component of the COM location occurs as the acceleration falls to zero because the weightlessness state is reached, sticking all the liquid fluid on the walls, which changes the COM location (see Figure 5 c). Further run up to 16400s requires only few seconds of run time, and produces the evolution of the trajectory for one orbit after the thrust pulse (see Figure 6 c).

Those results show that the new feature regarding the coupling between propulsion and vehicle are well managed and the propulsion system can be simulated with a great realism.

### 6.2. Active attitude control (AOCS) with thruster

The model of spacecraft propulsion system considered is shown in Figure 8. It comprises 4 thrusters for performing thrust in one main direction and with components along a transverse axis for performing a full 3 axis attitude control. The logic presented in previous chapter is used for the assessment of the system, taking into account the fact that only one thruster can be fired every second, and that a delay of one second is needed before using a thruster. The initial orbit is a LEO with all possible perturbations in attitude with a worst case of Centre of mass location (hence maximising the Sun pressure torques and drag torques). This case can represent a Cubesat propulsion system using thrusters of the kind of Liquid µPPT.

The goal of the example case is to stabilise the satellite attitude within ±0.2° for the 3 axis.

The plots in Figure 7 show in a zoom (for 150 s) that the constraints imposed for the propulsion system are well respected during the simulation: according to the value of the switch function of the Dead-zone control with hysteresis, zero or one of the thrusters is activated (level between 33% and 100 % of their nominal thrust) and for one second and when needed thrust pulses are successively performed every second. The overall result of the stabilisation within ±0.2° for the 3 axis roll, pitch, yaw is shown in their phase space in Figure 9, (right). On this example, one can notice that a two small overshoots occur for the control of the roll axis (for 28 seconds around time 5600 s, the roll axis overshoots up to -0.218°) representing less than 10% overshoot for less than 0.3% of the time. Such overshoots are coming from the complexity to manage the cross-coupling effects between the 3 axes. Although globally correct, a further more robust logic should be implemented in the next phase of the ESPSS improvement actions.

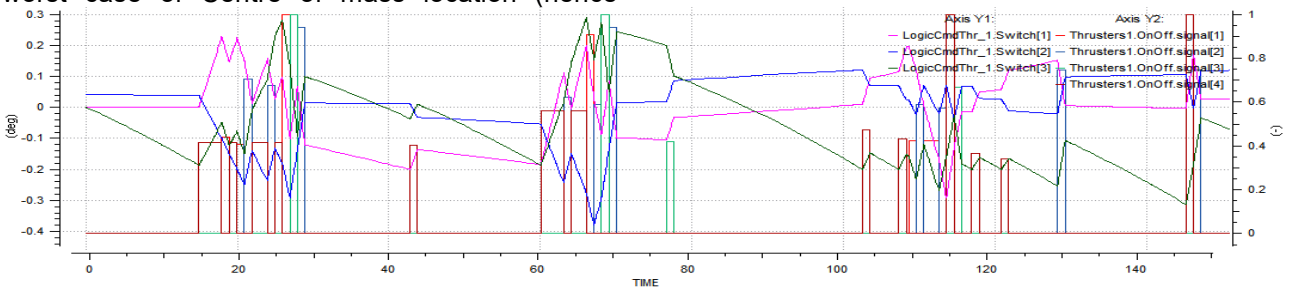



*Figure 7 Logic switch function (left scale) and thruster level (right) during a stabilisation (zoom 150 s)*

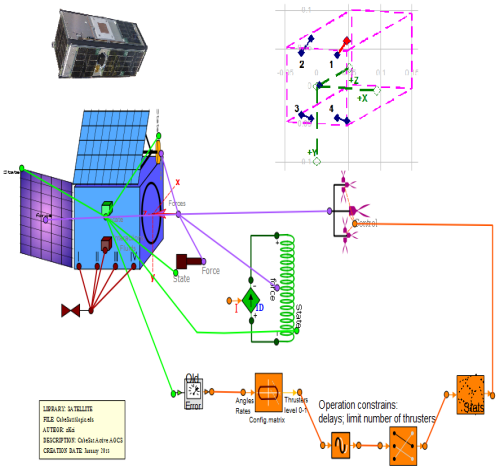



*Figure 8 Propulsion system for active AOCS*

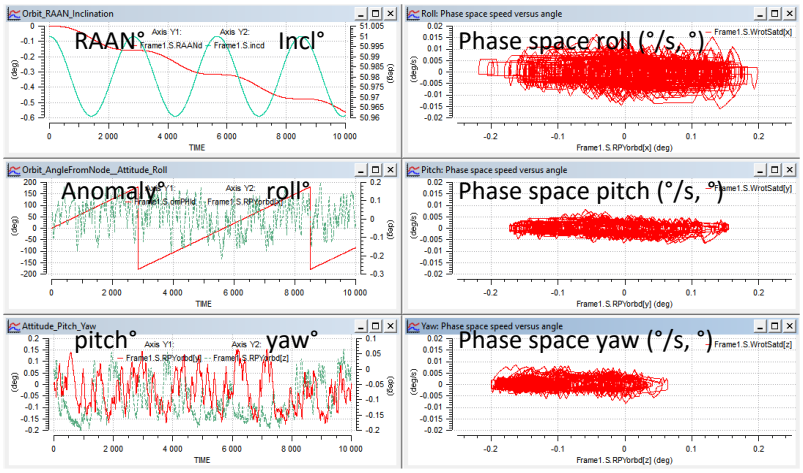



*Figure 9 Active stabilisation of attitude within 0.2° for 2 orbits, orbit and attitude angles; phase space at right*

## 7. Conclusions

The paper has presented in details the progress into the 4th phase of the work performed on the improvement of the ESPSS library (European Space Propulsion System Simulation) within the existing tool EcosimPro®. The significant advance is related to the implementation of the Archimedes pressure for the most general case of non-collinear acceleration and angular rate for the ESPSS fluid components. Thanks to the use of a new very compact equation this could be performed in a much easier form than in the previous more complex formulation. This has been successfully implemented for the components needed for a propulsion system. One example of propulsion systems has been presented for validation purpose with highlights on the coupling between propulsion and vehicle.

Among the progress, new components have been implemented successfully into the Satellite library allowing to simulate with "good" realism typical spacecraft missions, particularly of interest for propulsion applications as the active attitude control for a small satellite with thrusters.

## 8. Nomenclature

| | |
|---|---|
| *COM* | Centre of mass |
| $\vec{a}$ , $\gamma$ | acceleration (m/s²) |
| $\vec{\nabla} p$ | Gradient vector of the scalar p, $\left(\frac{\partial p}{\partial x} \quad \frac{\partial p}{\partial y} \quad \frac{\partial p}{\partial z}\right)^t$ |
| *ECI* | Earth Centred Inertial Frame |
| $\vec{F}$ | Non-body forces (N) |
| $\vec{F}_{thrust}$ | Resultant thrust vector (N) |
| $\vec{g}$ | Acceleration from body forces (m/s²) |
| G | Gravitation constant (N·m²/kg²) |
| $g_o$ | Constant 9.80665 (m/s²) |
| *LEO* | low Earth orbit |
| *m* | Spacecraft current mass (kg) |
| $\vec{M}$ | Moment (N·m) |
| $P_{perturb}$ | Perturbing forces vector from other celestial bodies, interaction forces (N) |
| p | Pressure (N/m²) |
| Q | Quaternion |
| $\vec{r}$ | Radius vector from focus source to spacecraft (m) |
| $\vec{r}$ | vector from COM to a point of the spacecraft (m) |
| *RCT* | reaction control thruster |
| *t* | Time (s) |
| $\vec{v}$ | Spacecraft velocity vector (m/s) |
| *DA* | Drag Area |
| *RW* | Reaction Wheels |
| *MT* | Magneto torquer |
| *SA* | Solar Arrays |
| *GG* | Gravity gradient |
| $\vec{\pi}$ , $\vec{\Pi}$ | Global vectors |
| ρ | density (kg/m³) |
| $\vec{\Omega}$ | Instantaneous rotation vector (rd/s) |
| × | vector product |

## 9. Acknowledgments

The research leading to these results has received funding from ESA contract N°4000111093/14/NL/PA

## 10. References

[R 1] *Steelant J., De Rosa M., Moral J. and Pérez R., "ESPSS Simulation Platform," Space Propulsion 2010, San Sebastian, Spain, 3-6 May 2010.*

[R 2] *Di Matteo F. and Steelant J, "Multi-Disciplinary Propulsion Simulations at Engineering Level by means of the European Space Propulsion System Simulation ESPSS," RTO / AVT/VKI Lecture Series on Fluid Dynamics Associated to Launcher Developers, EN-AVT-206-06, Von Karman Institute, St. Genesius-Rode, Belgium, 15-17/04/2013.*

[R 3] *C. R. Koppel, M. de Rosa, F. Rodriguez, J. Steelant, "The AOCS' effects on the Propulsion Subsystem using the ESPSS Satellite Library, " presented at 5th EUCASS 2013, Muenchen.*

[R 4] *C. R. Koppel, J. Moral, M. De Rosa, R.P. Vara, J. Steelant, and P. Omaly, "Satellite Propulsion Modelling With Ecosimpro: Comparison Between Simulation And Ground Tests, " presented at EUCASS 2009, published by EDP Sciences, 2011 in " Progress in Propulsion Physics 2 (2011) 743-764.*

[R 5] *ESPSS release 3.0.*

[R 6] *C R. Koppel, M. De Rosa, J. Moral and J. Steelant "Effects of a Satellite Mission on the Propulsion Subsystem," SP2012 2364281 Space Propulsion 2012, 7 – 10 May 2012,Bordeaux, France.*

[R 7] *C R. Koppel "ESPSS SATELLITE LIBRARY," ESPSS Workshop, ESA/ESTEC, Noordwijk, The Netherlands 21 May – 23 May 2013.*

[R 8] *Jacques Lewalle, "Incompressible Fluid Dynamics: Phenomenology, Concepts and Analytical Tools" (Lecture Notes), - Syracuse University, 2006.*

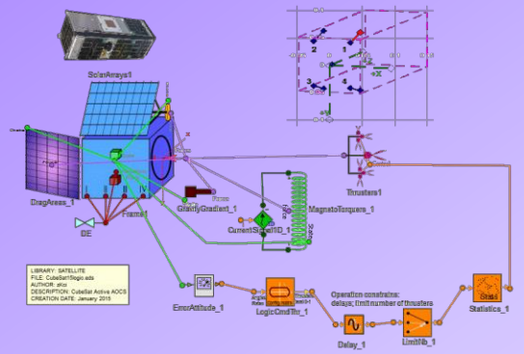
[R 9] *EcosimPro® http://www.ecosimpro.com/products/ecosimpro/ ESPSS release 3.0.*

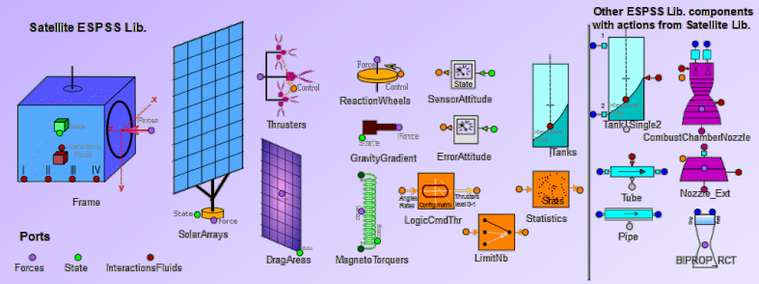
[R 10] *C. R. Koppel, F. Di Matteo, J.Moral, J. Steelant, "Coupled Simulation of the Propulsion System and Vehicle using the ESPSS Satellite Library, " presented at EUCASS 2015 Kracov, to be published by EDP Sciences.*

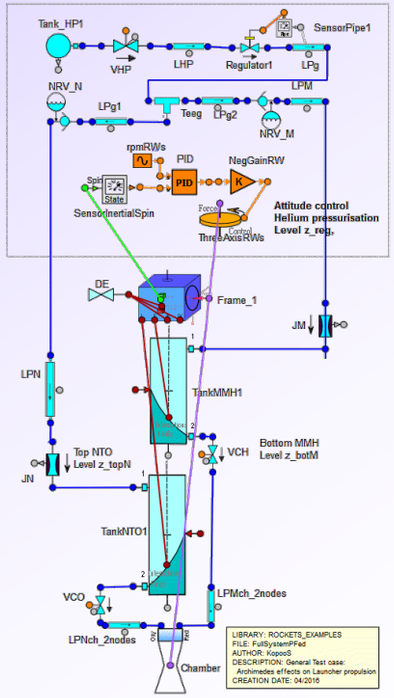
SPACE PROPULSION 2016

# Progress in Coupled Simulation Propulsion System and Vehicle with ESPSS Satellite Library

**Marriott Park Hotel, ROME, ITALY / 2–6 MAY 2016**







***Christophe R. Koppel***
***Francesco Di Matteo***
***Johan Steelant***
***Jose Moral***
***Javier Vila Vacas***






EMPRESARIOS AGRUPADOS

esa

KopooS

# Summary

- **Introduction**
- **ESPSS SATELLITE Library**

  ESPSS: European Space Propulsion System Simulation library

  - Frame component
  - References axis
  - Solar arrays, Thrusters components
  - Reaction wheels, Drag Areas, Gravity Gradient
  - Archimedes pressure new equation for tanks, tubes
  - **Check of Propulsion System coupled to vehicle**
    - Bi-propellant launcher stage
- **ESPSS SATELLITE Library**
  - Logic Command
  - **Examples of active AOCS with thrusters**
    - A cubesat case
- **Conclusions**

# INTRODUCTION

- **Presentation of coupled simulation of the propulsion system and vehicle**
  - **Part of 4th phase of the development/validation of ESPSS-EcosimPro®.**
  - **Comes after previous phase "evolutionary behaviour of components“**
- **A Satellite library has been set up allowing *AOCS* simulation**
  - **Present work includes a significant updated of the Satellite lib.**
- **The coupling is performed by updating the Fluid flow, Tanks and Combustion chamber components**
  - **Accelerations generated by thrusters, reaction wheels and other satellite components => influence the fluid-dynamic behaviour of the propulsion system.**
  - **Moreover, the propellant tanks’ filling level influences the satellite’s centre of mass and moments of inertia**
  - **Induce the effects on the pressure and flow conditions in fluid lines.**
  - **➔Archimedes pressure coming from acceleration (due to non body forces) and angular rates given by the vehicle has to be included.**

ESPSS: European Space Propulsion System Simulation library

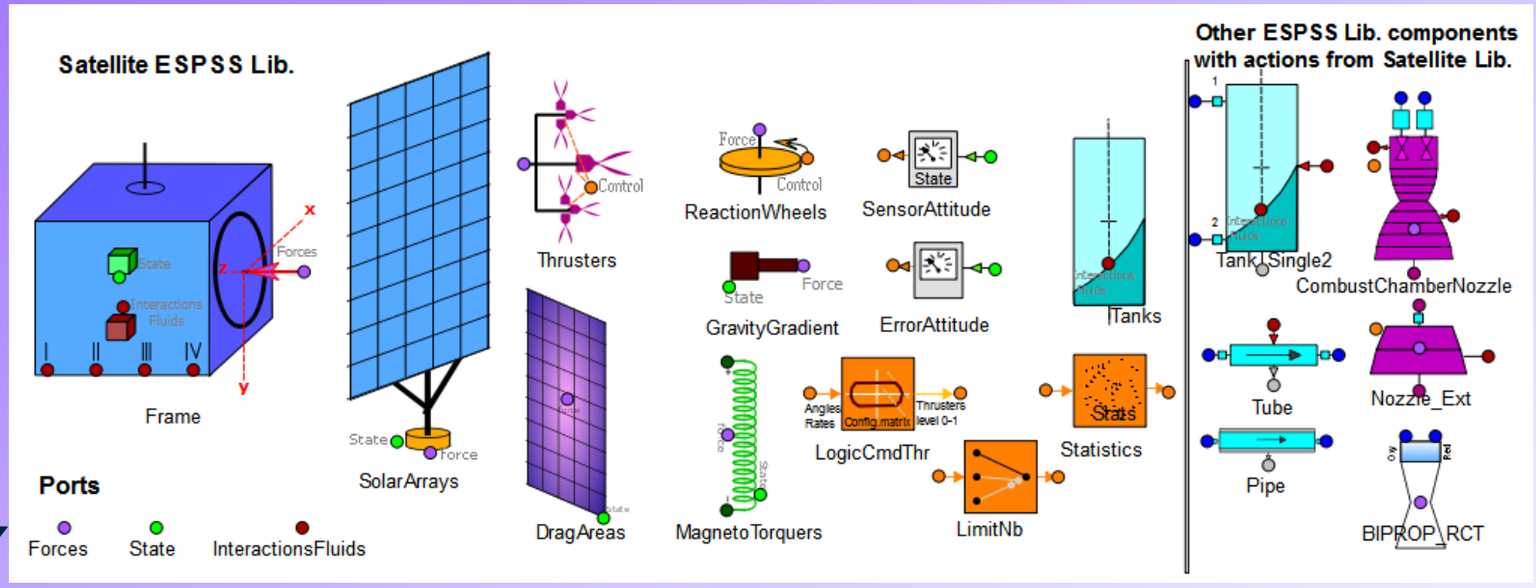




EMPRESARIOS AGRUPADOS esa KopooS

# ESPSS - SATELLITE library -- …

## Palette




## 3 Ports

- Port **Forces** : multifunction port for inputs from a set of Thrusters, Reaction Wheels, Solar Arrays, Drag Areas, Magneto Torquers and Gravity Gradient
  - **port directions IN for the satellite frame, OUT for all other components.**
  - **type SUM in order to automatically account for all mass flow rates, forces, moment, angular momentum, power coming from all connected components.**
- Port **State**: multipurpose port for the attitude and orbit control and for 3D visualization, as well as the needed inputs for some components
  - **port directions OUT for the satellite frame, IN for all other components**
- Port **InteractionsFluids**: to communicate the "mass" data from Tanks to Frame: mass of fluids, inertia matrix and location of the fluids Centre Of Mass (COM)

# Frame component

## In charge of the flight dynamic: Orbit & attitude

- Provide **State** to the objects that need: SAs, GGs, DAs, MTs, Tanks
- **Input data**: initial orbit, the forces coming from Moon, Sun & all planets gravity are solved inside, and from **Forces** port : Forces torques, angular momentum vectors ( from any connected object: thrusters, solar arrays, RW, etc..)

## Orbital dynamic under gravity, Moon, Sun perturbation, J2

$$\frac{d\vec{\pi}}{dt} = \vec{\Pi} \qquad \vec{\pi} = \begin{pmatrix} \vec{r} \\ \vec{v} \\ m \end{pmatrix} \qquad \vec{\Pi} = \begin{pmatrix} \vec{v} \\ -GM_{focus}\,\vec{r}/r^3 + \sum\left(\vec{T}_{thrust} + \vec{P}_{perturb}\right)/m \\ \sum -thrusters\_mass\_flow \end{pmatrix}$$

## Attitude dynamic

$$\frac{d\vec{H}}{dt} = \vec{M} \qquad \vec{H} = [I]*\vec{\Omega} \qquad \frac{d\vec{H}}{dt} + \vec{\Omega}\wedge\vec{H} = \vec{M}_{Control} + \vec{M}_{Perturbation}$$

$$\vec{M}_{Control} = \left\{ M_{thruster} - \sum_i \frac{d\vec{H}_{RWi}}{dt} \right\} \qquad \vec{M}_{Perturbation} = \left\{ M_{perturb} - \sum_i \vec{\Omega}\wedge\vec{H}_{RWi} - \sum_i \vec{\Omega}\wedge\vec{H}_i - \sum_i \frac{d\vec{H}_i}{dt} \right\}$$

## Attitude angles with quaternions

$$\frac{dQ}{dt} = +\frac{1}{2}[\Omega]\cdot Q$$

- The integration produces at each time *t* the quaternion enabling (with suitable conversion matrixes) orbital & attitude angles to be used for inputs

## COM & Inertia management: update only when changes occur by steps





EMPRESARIOS AGRUPADOS esa KopooS

# Reference Axis used

## ENUMERATIVES Used

- **ECI = {Xq, Yq, Zq} Earth Centred Inertial equatorial**
  - Equivalent to $R_{J2000}$
- **PoI = {Xp, Yp, Zp} Orbital polar system**
  - Natural frame for orbits
- **Orb = {X, Y, Z} Telecom Satellite orbital system**
  - Equivalent to $R_{ELOF}$ (Estimated Local Orbit Frame -in winter-), or SLO (Spacecraft *Local Orbit*) or ORF (Orbit Reference Frame)
- **Sat = {x, y, z} Satellite reference system**
  - Equivalent to $R_s$ at launcher interface, also known as SCB (S/C Body Frame)

# Frame component: EL listing of the Continuous part

EL: EcosimPro language

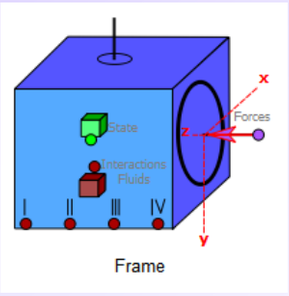
```
CONTINUOUS
FlagAccel=(NOmeg+NAccel)/WeightlessnessAccel  --simple ratio
EXPAND (t IN Sat) OCOMFluidsTanks[t]=(SUM(i IN 1,nTanks;iff.FluidMassComponent[i])*iff.OCOMFluids[t]+
TimeDay=TIME/86400+(Day1900+2)+dayshift  -- TimeDay to be compatible with Excel dates,--it has beenadjusted by+2 units (Day1900+2) with Excel check
JD=JD_o+TIME/86400  --current time Julian Day
COMFlag =SUM(t IN Sat; sqrt(abs(OC[t]))+ sqrt(abs(iff.OCOMFluids[t]))+sqrt(abs(iff1.OCOMFluids[t]))+s
----------
--MASS CONSERVATION
----------
FluidMassTanks=SUM(i IN 1,nTanks;iff.FluidMassComponent[i])+iff1.FluidMassComponent[1]+iff2.FluidMassComponent[
massfromflow'=f.mfr[1]+f.mfr[2]  --Option: case no tanks in model, so integral of thrusters mass flow rates Fuel and Oxydizer mfr
mass=IF (FlagTanksFlow) massDry_o+FluidMassTanks ELSE massfromflow  --FlagTanksFlow is a Global variable
----------
--ORBITAL DYNAMIC in ECI: from F=m.d²R/dt² in inertial frame
----------
EXPAND_BLOCK (j IN ECI)
R[j]' = V[j]  -- R could be named ReSat[j] form earth To Sat in ECI equatorial
V[j]'= - GMplanet[Earth]/R_module**3*R[j]+AccelThrusterSunPress[j]+Perturb[j]/mass*FlagPerturb+AccelJ2[j]*FlagPerturb
-- accessories
AccelThrusterSunPress[j] = SUM(t IN Sat; MatrixECItoSat[t,j] * Accel[t])  -- inverse matrix product to get in ECI from Sat
AccelJ2[j] = SUM(k IN Orb; MatrixECItoOrb[k,j] * AccelJ2Orb[k])  -- inverse matrix product to get in ECI from Orb
Perturb[j]=mass*SUM (k IN Sol EXCEPT Earth ; - GMplanet[k]*(RplanetSat[k,j]/RplanetSat_module[k]**3 + RePlanet[k,j]/R
            GMplanet[k]*(RplanetSat[k,j]/RplanetSat_module[k]**3 + RePlanet[k,j]/RePlanet_module[k]**3))
EXPAND (k IN Sol) RplanetSat[k,j]=-RePlanet[k,j]+R[j]  --vector addition
END EXPAND_BLOCK
-- external forces Acceleration evaluation in Sat
EXPAND (t IN Sat) Accel[t] = f.F[t] / mass
MoonSunECI( JD, RePlanet)  --Position from Earth to Moon and Sun and planets in ECI at current time Julian Day
--Orbit orientation: to get the Quaternion wrt ECI to orient ECI to Orb: Qorb
CoordRVEClToEulerAngles (R, V, RAAN, inc, omPHI, GMplanet[Earth], EarthR, AltApo, AltPer,Alt, Exc, om, PHI, sma) --gi
CoordEClEulerOrbQuater(RAAN, inc, omPHI, QQorb)  --give Qorb
CheckChoiceQuater(QQorb,Qorb)
CoordAqaterBmatrixEnum(Qorb, MatrixEClToOrb)
--J2 effect in the orbital frame 'Orb' Z along radius toward Earth; X in plane perpendicular to radius; Y = Z ^X
AccelJ2Orb[X]=+3/2*EarthJ2*GMplanet[Earth]*EarthR**2/R_module**4*-1*sin(2*omPHI)*sin(inc) **2
AccelJ2Orb[Y]=+3/2*EarthJ2*GMplanet[Earth]*EarthR**2/R_module**4*sin(omPHI)*sin(2*inc)
AccelJ2Orb[Z]=+3/2*EarthJ2*GMplanet[Earth]*EarthR**2/R_module**4*-1*(3*sin(omPHI)**2*sin(inc)**2-1)
----------
--ATTITUDE DYNAMIC in Sat: M=dH/dt in inertial frame with H= [InertialMatrixSatFrozen].Wrot ==> M=dH/dt+Wrot ^ H all in Sat frame
----------
EXPAND_BLOCK (t IN Sat)
-- dHsat/dt+Omega x Hsat = Torques written as: Inertia.dOmega/dt+Omega x Hsat = Torques
SUM(u IN Sat;InertiaMatrixSatFrozen[t,u]*Wrot[u]')+SUM(u IN Sat;CMWrot[t,u]*Hsat[u])=TorqueCOM[t]+TorqueROT[t]
Hsat[t]= SUM(u IN Sat; InertiaMatrixSatFrozen[t,u] * Wrot[u] )
TorqueCOM[t]=f.M[t]+MomentCO[t]  --torques Mthruster+Mperturb at Frame COM
TorqueROT[t]=-f.H[t]'-SUM(u IN Sat; CMWrot[t,u] * f.H[u] ) -- torques RWs and other rotating devices
END EXPAND_BLOCK
XcrossY ( f.F, OC, MomentCO)  --CO*F=F^OC=MomentCO to be added: Moment/C= sum CFi ^ Fi = CO ^ sum Fi+sum OFi ^Fi = f.F ^ OC+f.M /O => Moment[t] /C=MomentCO[t]+f.M
CrossMatrixEnum ( Wrot, CMWrot)
----------
--ATTITUDE ANGLES : in matrix dQ/dt= + 0.5 DQmat(Wrot) .Q from: in quaternion algebra dQ/dt=0.5 Q.Wrot (with Wrot as a pure quaternion wrt Sat) ref. Guizou, cours SCAO, 2000.
----------
EXPAND (i IN 1,4) Q[i]' = + 0.5*SUM(j IN 1,4; DQmat[i,j] * Q[j] )-- give by integration the general quaternion Q wrt ECI to orient ECI to Sat.
DervativeQmatrixSatAxis(Wrot,DQmat)-- Derivative of the quaternions (kinematics equations) with Wrot into satellite axis. In quaternion algebra one also have dQ/dt=0.5 WROT
CheckChoiceQuater(Q,QQ)-- compute Qsat wrt Orb to orient Orb to Sat.
Qinverse(Qorb,Qorbinv)
--Q1product Q2(Qorbinv, QQ, QQsat, "operation Qorbinv, Q ")-- in quaternion algebra: Q=Qorb.Qsat so Qorbinv. Q=Qsat
Q1Q2(Qorbinv,QQ,QQsat," operation Qorbinv,Q ",0)-- in quaternion algebra: Q=Qorb.Qsat so Qorbinv. Q=Qsat
CheckChoiceQuater(QQsat,Qsat)
CoordAquaterBcardan(RPYorb[z], RPYorb[y],RPYorb[x],Qsat)  --gives yaw pitch roll wrt Orb
CoordAquaterBcardan(RPYeci[z], RPYeci[y],RPYeci[x], Q)  --gives yaw pitch roll wrt ECI
--compute the general frame orientation matrix ECI to Sat from the general quaternion Q wrt ECI. For all vector in Sat to be known in ECI : thruster force for example by using the inverse matrix.
CoordAqaterBmatrixEnum (Q,MatrixEClToSat)
explPower= f.Power --POWER in explicit
----------
-- STATE VECTOR (much larger than the strictly needed for control problems)
----------
S.mass=mass
-- S.OCOM=OC
S.InertiaCOM=InertiaMatrixSatFrozen  -- added for connected devices
--S.Accel = Accel -- no more needed because in Global...only the non gravity forces (non volume forces) must be provided to other components)
EXPAND_BLOCK (t IN Sat)
S.Wrotd[t] = degfull(Wrot[t])  --wrt angular rate inertial frame, but coord in body
S.Wdotd[t] = degfull(dWrot[t])
```

COM update flag

Mass of fluids & Mass conservation

First order derivatives: R'=V …

## Orbital dynamic



Frame

## Attitude dynamic

Quaternions

## State vector for State port

# Not a code… only equations (with several functions…)





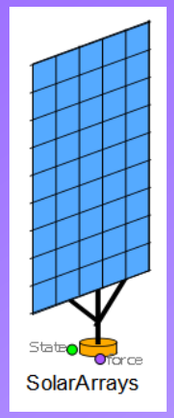
EMPRESARIOS AGRUPADOS

esa

KopooS

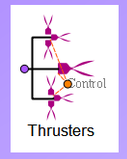
# Solar arrays, Thrusters components

## Solar arrays (vector component)



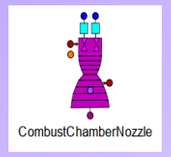

- Automatic orientation of the solar arrays around their axis with respect to the Sun or Fixed orientation wrt satellite.
- Receives from State port the Satellite to Sun vector and other information.
- Classic equations of Sun pressure interaction on the solar array with input coefficients of specular reflection and absorption.
- The output to Forces port are the 3D forces due to the Sun pressure, their moments and the power produced by the solar cells

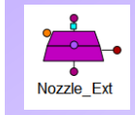
## Thrusters (vector component)

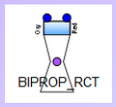



- When activated (on its Control port), this component outputs to the Forces port : thrust vector with respect to the satellite, moment induced by the thrusters forces (and the mass flow rates).
- In case of electric propulsion, the electric power consumption is provided as well.

## Rocket Engine (Comb.Chamber Nozzle, BIPROP_RCT,…)




- Idem: thrust , mass flow rate and moments to Forces port



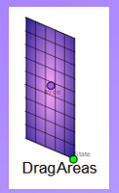


EMPRESARIOS AGRUPADOS esa KopooS

# Reaction wheels , Drag areas, Gravity Gradient

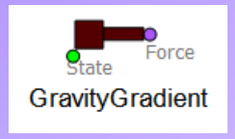

## Reaction wheels (vector component)

- Once activated (on its Control port) outputs to Forces port: angular momentum vector with respect to the satellite as well as the electric power consumption of all reaction wheels considered.
- Limit torque, Rpm rate, Rpm max speed: differential equations included




## Drag area component

- Inputs from State port: position wrt Earth. Outputs to Forces port : drag force coming from the dynamic pressure in flight.
- Earth atmosphere type is taken from ECSS standard with several cases depending on the Sun activity.




## Gravity gradient (replacing Gravity Booms)

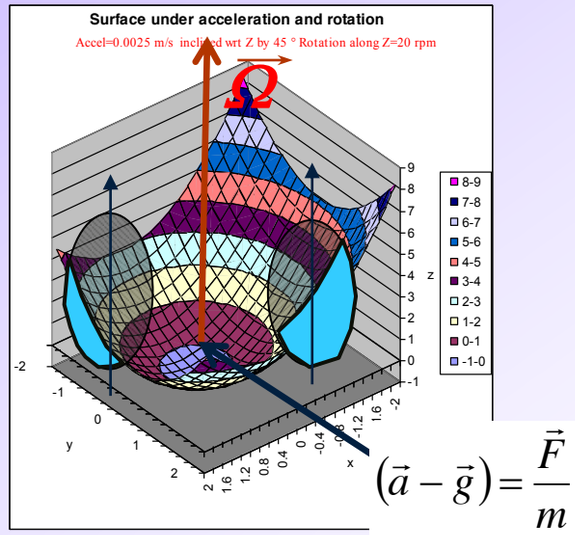
- Based on the input from the State port for position wrt Earth.
- Provides Forces port : moment vector with respect to the satellite without any forces (pure torque).

$$\vec{M} \approx \frac{3GM_{earth}}{R^3}\vec{e}_{earth} \times \left( [Inertia] \cdot \vec{e}_{earth} \right)$$

# New equation: Archimedes pressure for Tank, Tube, Pipe

- Former classic Archimedes pressure under acceleration $\gamma$ :
  "$dp = -\rho.\gamma.(z - z_0)$"
- For Archimedes hydrostatic pressure under acceleration $\vec{a} - \vec{g}$ and constant rotation $\vec{\Omega}$ at vector $\vec{r}$ from the *COM* gives new equation for the gradient

$$\vec{\nabla} p = -\rho\left(\vec{a} - \vec{g} + \vec{\Omega} \times \left(\vec{\Omega} \times \vec{r}\right)\right)$$

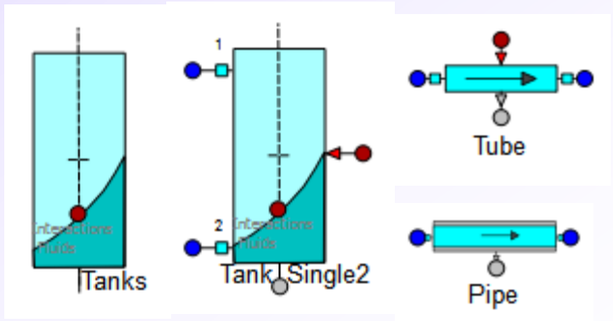



$$(\vec{a} - \vec{g}) = \frac{\vec{F}}{m}$$

- With constant density $\rho$, gives new equation for the pressure difference $\Delta P$ and free surface

$$p_2 - p_1 = \left[ -\rho\left(\vec{a} - \vec{g}\right) \cdot \vec{r} + \tfrac{1}{2}\rho\left(\vec{\Omega} \times \vec{r}\right)^2 \right]_{\vec{r}_1}^{\vec{r}_2}$$

- Even the square of cross product can be simplified as a square of dot product… $\vec{\Omega}^2\vec{r}^2 \sin^2 = \vec{\Omega}^2\vec{r}^2 - \vec{\Omega}^2\vec{r}^2 \cos^2$
- For faster runs: Acc., COM and rot. are filtered at 1st order
- Acc., COM and rot.: global hidden variables of Fluid_Flow_1D lib.
  ➔ thus no links needed for using them
- The same vectors are written in the [Sat] frame for simpler handling



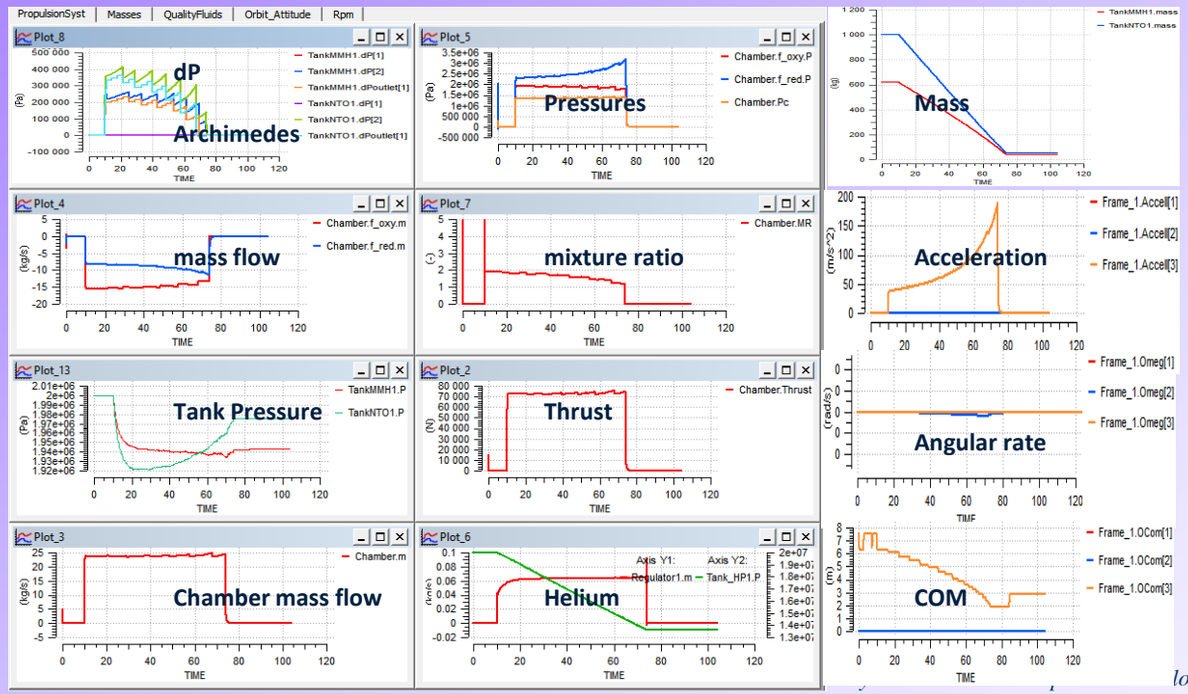


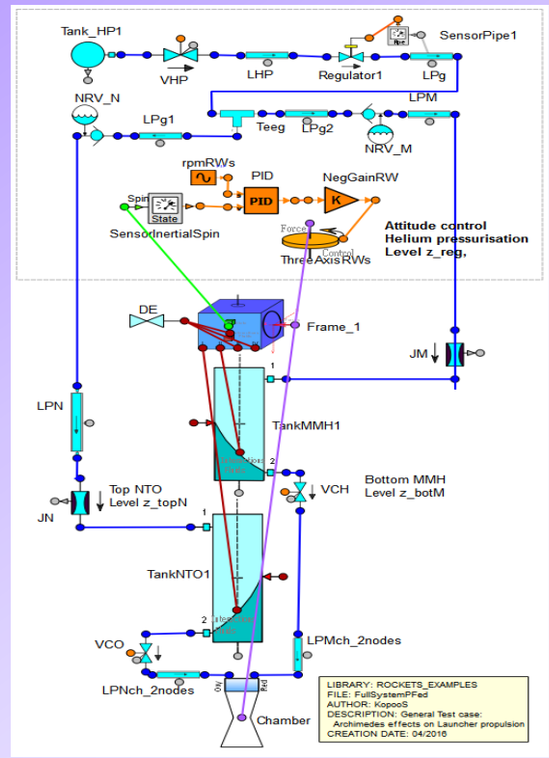



EMPRESARIOS AGRUPADOS esa KopooS

# Check of Propulsion System coupled to vehicle

- **Bi-propellant launcher stage**
  - 2 main tanks (NTO + MMH)
  - Several valves, regulator, tees and tubes
  - 1 bipropellant rocket engine
- **Results : Thrust pulse 64 s**




- Mass decreases … COM changes
- Acceleration increases along Z axis
- Inertial Angular rate: maintained null with PID
- ➔Archimedes pressure changes in all parts of the propulsion system
- Here, thrust almost constant…
- Check successful





EMPRESARIOS AGRUPADOS esa KopooS

# SATELLITE library : LogicCmd component…


## LogicCmd

- parameters: number thrusters “nthr” and DOF (degrees of freedom for Forces & Torques)
- Dedicated to the attitude control with thrusters: dead zone with hysteresis that provides the torques required $\vec{F}$

1. Select the thrusters $\vec{T}$ thrust level to perform the required $\vec{F}$
   - Configuration matrix A (rectangular) such that $\vec{F} = [A].\vec{T}$
   - Use the pseudo inverse $[A]^{-1ps} = [A]^t \left[ [A][A]^t \right]^{-1}$
   - to get $\vec{T} = [A]^{-1ps}.\vec{F}$
2. Deal with the “one sided” thrusters: avoid negative thrust in vector $\vec{T}$
   - add a “null vector” of $[A]$ : $\vec{0} = [A].\vec{T}_{null}$ ➔ $\vec{T}_{one\_sided} = [A]^{-1ps}.\vec{F} + \vec{T}_{null}$
3. Attitude control with switching lines: dead zone with hysteresis
   - …

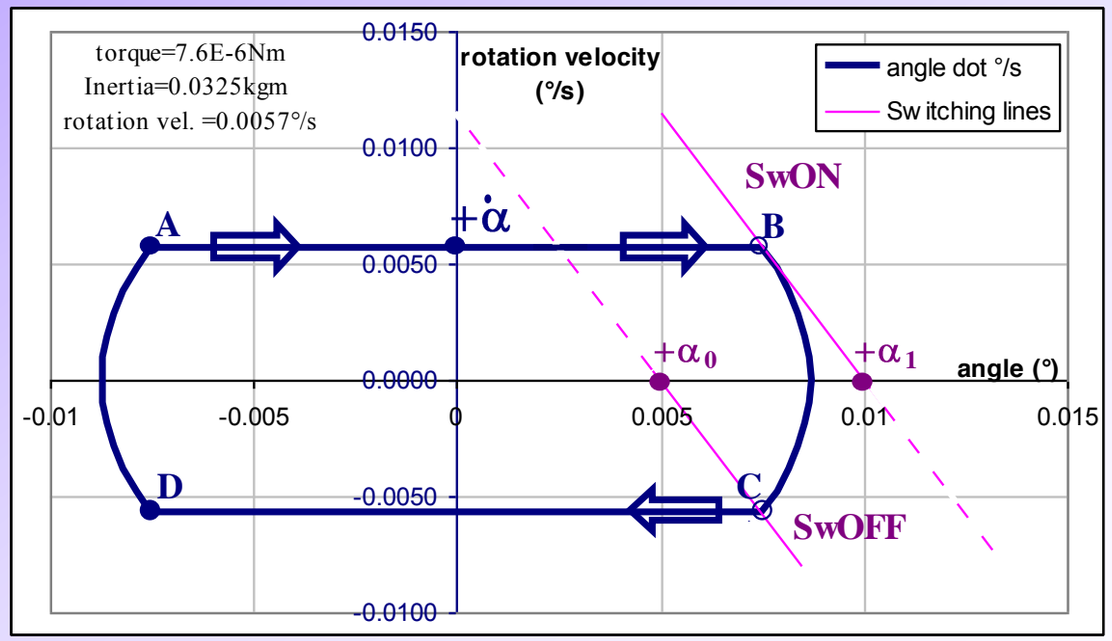
# …LogicCmd component
## DEAD ZONE WITH HYSTERESIS


- **Principles**
  - Permanent compensation of the external torques
  - Active attitude control with thrusters switched On and Off
  - →Use of Schmitt triggers: dead-band and switching functions
- **Limit cycle (without perturbations) for each axis**
  - **Dead-band cost constant**

    Cost of Dead-band control accuracy in " torque impulse/year ($T_{iy}$) "

$$K_{DB} = \frac{T_{iy}}{I} \cdot \left( \frac{\alpha_1}{\dot{\alpha}^2} \right)$$

  - **$K_{DB}$** is about **700 000** units $Ns^3(kg.m.deg.year)^{-1}$ with $T_{iy}$ in *Nms/year*




- **The Dead-band cost must be added to the consumption for the simple compensation of the external torques…**

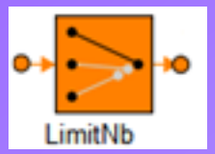




EMPRESARIOS AGRUPADOS esa KopooS

# SATELLITE library : components for LogicCmd

## LimitNb component




- Linked to the LogicCmd component
- Equations limit automatically the number of thrusters fired simultaneously
- In case of need sequence the firing between the thrusters.
- Its input data are the maximum number of thrusters simultaneously fired and the period of sequencing.

## Statistics component


- Linked to the LogicCmd component
- Exclusively used for transparently controlling the thrusters (directly from the input port to the output port)
- while keeping trace of the use of the different thrusters.

## The SATELLITE library is ready for use with realism



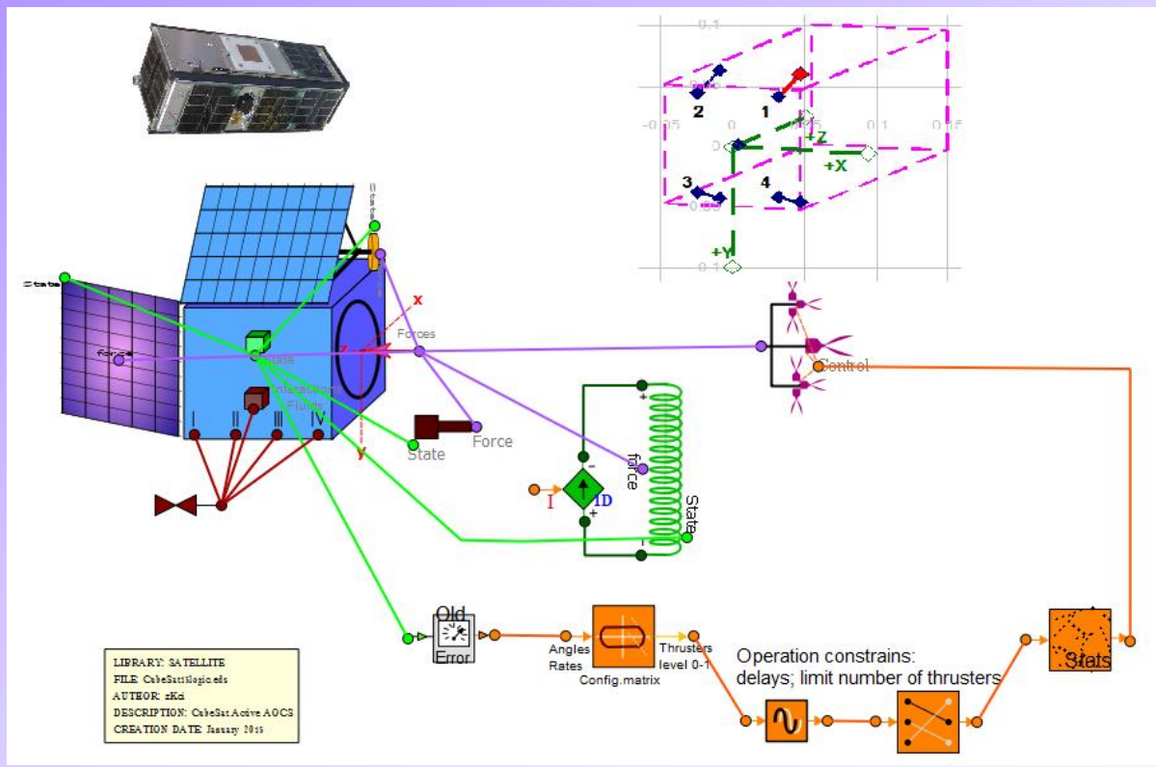


EMPRESARIOS AGRUPADOS esa KopooS

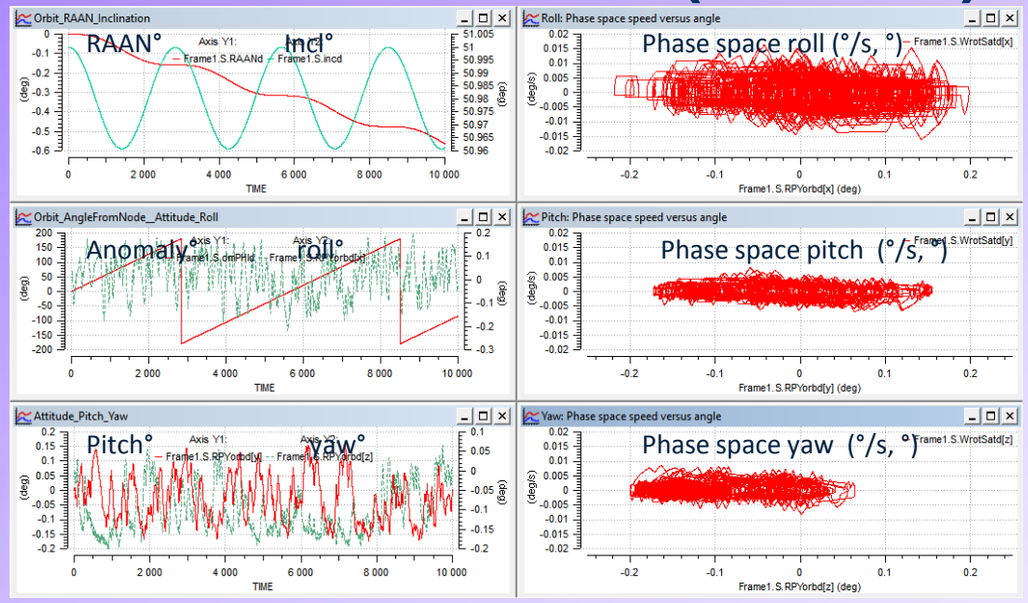
# Examples of active AOCS with thrusters

## A Cubesat Case System

- 4 thrusters
- DragAreas, SolarArrays, Magneto-torquers (only as parasitic torque)
- COM worst case deviation
- Logic to control 3 axis
- Cubesat limit: only 1 thruster on max




## Results : 10 000s (2 orbits)



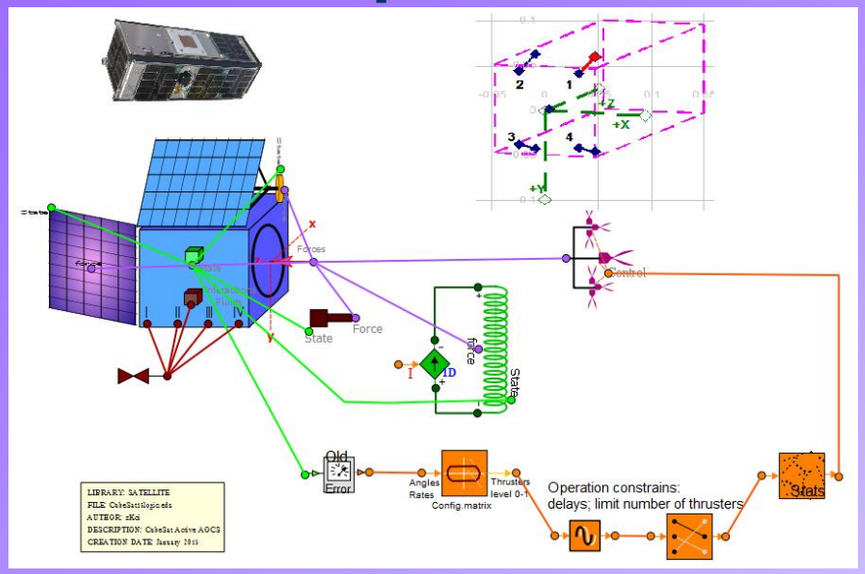

- Stabilisation within 0.2° for the 3 axis
- Real cycles far different from the LIMIT CYCLE shown before
  - Due to real perturbation torques
  - Drag torques, Sun pressure torque,..
  - Small overshoot for roll (<0.3% of the time)





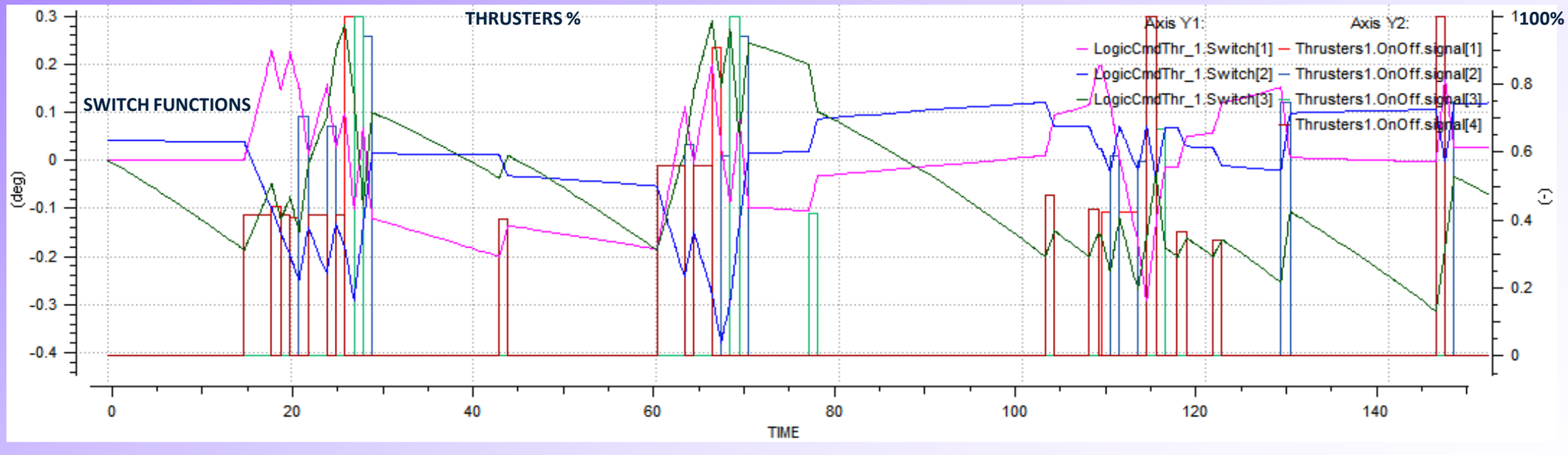
EMPRESARIOS AGRUPADOS esa KopooS

# Examples of active AOCS with thrusters - ZOOM




## Results Zoom on 150 s

- Switch functions (3 axis)
- Thrusters command (33% to 100%)
- Logic constrains : never use the thrusters simultaneously, but the ones after the others➔ works well








EMPRESARIOS AGRUPADOS esa KopooS

# Conclusions

- As an extensions of the 4th development phase of the European Space Propulsion System Simulation (ESPSS) the updated SATELLITE library has been described in details
- Significant advance in implementation of the Archimedes pressure for the ESPSS fluid components (most general case of non-collinear acceleration and angular rate).
  - … with a new very compact equation

$$p_2 - p_1 = \left[ -\rho \frac{\vec{F}}{m} \cdot \vec{r} + \tfrac{1}{2} \rho \left( \vec{\Omega} \times \vec{r} \right)^2 \right]_{\vec{r}_1}^{\vec{r}_2}$$

- Case of general launcher propulsion system with coupling has been presented
- Simulation with “good realism” can be performed for active AOCS
- ESPSS under EcosimPro is ready for ESA member states Users
- Of course many improvement can be foreseen … see next phase

AOCS: Attitude and Orbit Control System





EMPRESARIOS AGRUPADOS esa KopooS

# Thanks you for your attention





EMPRESARIOS AGRUPADOS

esa

KopooS

# The library : Other components & Ports

- **Satellite Frame**
  - Thrusters
  - SolarArrays, DragAreas
  - Reaction Wheels
  - GravityGradient
  - MagnetoTorquers
- **All those components may produce forces F, moment M, angular momentum H, power P and may use a mass flow-rate mfr.**

- **The port Forces is used as OUT for any component, IN for the frame with automatic SUM (➔ automatic conservation laws)**

```
PORT Forces "Forces port" --IN for the frame, OUT for the components in order to get the right sums
        SUM REAL F[Sat] UNITS "N" "External forces "
        SUM REAL M[Sat] UNITS "N*m" "External moments "
        SUM REAL H[Sat] UNITS "N*m*s" "External Angular momentum "
        SUM REAL mfr[2] UNITS "kg/s" "For components, negative: mass flow rate used
        SUM REAL Power UNITS "W" "For components, negative: electrical power used; positive: produced"
END PORT
```

- **The port State is used as IN for components, OUT for frame**

```
PORT State "State port" --Satellite state (for control, forces and rotation interactions)
        EQUAL REAL mass UNITS "kg" " Satellite actual mass "
        EQUAL REAL InertiaCOM[Sat,Sat] UNITS "kg·m^2" " Satellite inertia matrix
        EQUAL REAL Wrotd[Sat] UNITS "deg/s" " Body Sat frame angular velocity wrt ECI, in Sat axis"
        EQUAL REAL WrotSatd[Sat] UNITS "deg/s" " Sat axis angular velocity wrt Orb frame, in Sat axis"
        EQUAL REAL Wdotd[Sat] UNITS "deg/s^2" " Satellite axis angular acceleration "
        EQUAL REAL RPYorbd[Sat] UNITS "deg" "Roll Pitch Yaw angle Sat wrt Orb "
        EQUAL REAL RPYecid[Sat] UNITS "deg" "Roll Pitch Yaw angle Sat wrt ECI "
        EQUAL REAL RAANd UNITS "deg" "PrecessionZA angle Pol wrt ECI equatorial "
        EQUAL REAL incd UNITS "deg" "NutationU angle Pol wrt ECI equatorial "
        EQUAL REAL omPHId UNITS "deg" "Rotation propre angle Pol wrt ECI equatorial "
        EQUAL REAL omd UNITS "deg" "argument perigee "
        EQUAL REAL PHId UNITS "deg" "true anomaly "
        EQUAL REAL sma UNITS "m" "semimajor axis "
        EQUAL REAL exc UNITS "-" "eccentricity "
        EQUAL REAL periodWrtEarth UNITS "s" "current
        EQUAL REAL JD UNITS "day" " Julian Day "
        -- Sun and Earth in Sat (for Sun pressure perturbation, electrical power, gravity gradient)
        EQUAL REAL RSatSun[Sat] UNITS "m" " satellite COM to Sun position
        EQUAL REAL RSatEarth[Sat] UNITS "m" " satellite COM to Earth
        EQUAL REAL RSatPlanet[Sol,Sat] UNITS "m" " satellite COM to planets
        EQUAL REAL Vsat[Sat] UNITS "m/s" "Orbital satellite COM velocity,
        --Orbital state and for 3D real time visualization
        EQUAL REAL R[ECI] UNITS "m" "Orbital satellite COM position in ECI"
        EQUAL REAL V[ECI] UNITS "m/s" "Orbital satellite COM velocity in ECI"
        EQUAL REAL PZZZQ UNITS "s" " Pilot Date of day from 1900 and time for quaternion
        EQUAL REAL Q[4] UNITS "-" " quaternion to orient ECI to Satellite axis "
        EQUAL REAL Qsat[4] UNITS "-" " quaternion to orient Orbit axis Orb to Sat axis "
        EQUAL REAL Torques[Sat] UNITS "N·m" " all torques applied to the satellite
        EQUAL REAL TorqueRot[Sat] UNITS "N·m" " the torque from only
END PORT
```

- **The port InteractionsFluids is used as OUT for the tanks, IN for the frame (similar to Forces) but without any SUM**





EMPRESARIOS AGRUPADOS esa KopooS