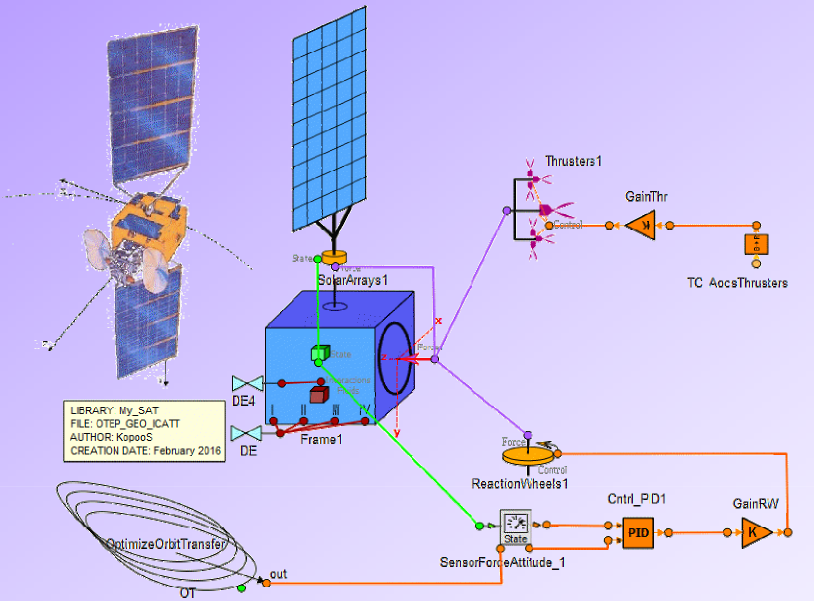
6th ICATT

Low Thrust Orbit Transfer Optimiser for a Spacecraft Simulator

(ESPSS -Ecosimpro® European Space Propulsion System Simulation)

Darmstadt, 14th to 17th March 2016



Christophe R. Koppel

KopooS Consulting Ind., 75008 Paris, France

Summary

- Introduction
- General review on orbit transfer strategies
- Low thrust optimiser principles
- Optimiser: Mipelec (from CNES)
- Integration Optimiser into EcosimPro® ESPSS
- Example of realistic optimised EP transfer
- Conclusions

ESPSS: European Space Propulsion System Simulation library

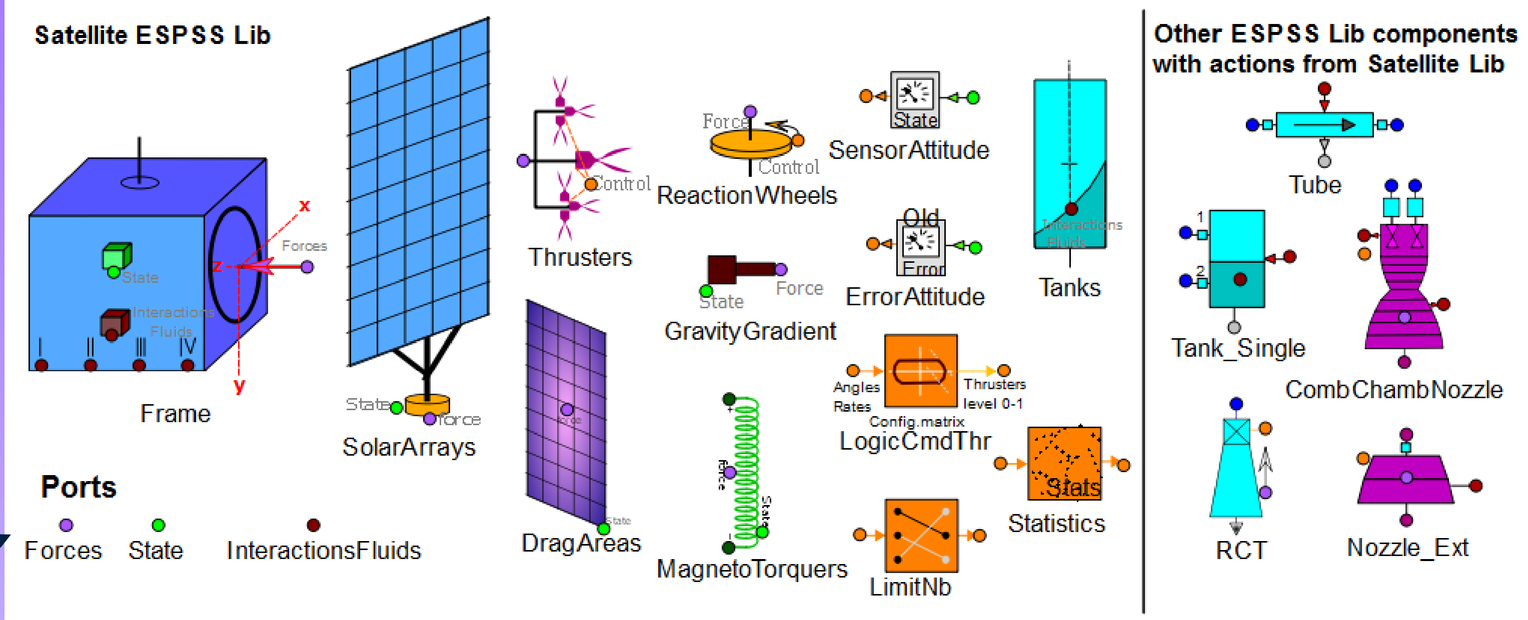
INTRODUCTION

- **EcosimPro® is a Physical Simulation Modelling tool**
  - Object-oriented tool dedicated for system analysis's.
  - That is a visual simulation tool
  - Solving dynamic systems differential equations and discrete events.
  - It can be used to study both transients and steady states.
  - The propulsion libraries *ESPSS* from ESA for example, allows the user to draw (and to design) the **propulsion systems.**
- **A new feature has been included into recent release of *ESPSS* for addressing the "Evolutionary behaviour of components" and "Coupling with the vehicle dynamics"**
  - Includes a "**Satellite library**" : the **flight dynamic** (orbit and attitude) for a full spacecraft including **orbital and attitude perturbations, Sun eclipses**.
- **In order to assess realistic Electric Propulsion missions**
  - **An optimiser for orbit transfer has been integrated**
  - **With design of few new components for interfacing the optimiser and the existing libraries of *ESPSS*.**

ESPSS: European Space Propulsion System Simulation library

ESPSS - SATELLITE library – Vehicle dynamics

- **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)

ESPSS: European Space Propulsion System Simulation library

Frame component

In charge of the flight dynamic: Orbit & attitude

- **Provide state to the components that need:** SAs, GGs, DAs, MTs, Tanks
- Input data: initial orbit, forces torques, angular momentum vectors from thrusters, solar arrays, RW, etc.. The body forces coming from Moon, Sun & all planets gravity are solved inside.

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

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

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

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

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

General review Low thrust orbit transfer strategies

- **Strategies from GTO to GEO : Lots of strategies have been investigated**
  - First : "apogee burns" (similar to the high thrust) ➔ increase of the *DV* needs but duration of the transfer prohibitive.
  - Second: super-GTO strategies with "apogee & perigee burns“ ➔ duration of the transfer could be decreased with increase of the *DV*
- Finally the **“continuous thrust strategy”** has been disclosed **in the Koppel patents** “…orbit transfer starting from an elliptical initial orbit that differs substantially from the target orbit, and in particular that has eccentricity that is different from that of the target orbit…”

  ➔decrease at a maximum the duration of the orbit transfer
    - without taking care of the propellant mass increase because using High ISP
    - ➔mass penalty due to higher *DV* is not really significant
  - Heavy optimization techniques, mainly ignoring possible service interruptions, conducted by S. Geffroy (CNES) have shown similar results as the proposed strategy.
  - This strategy has been compared with other patented strategy from A. Spitzer (HUGHES).
- Lots of practical advantages in favour of **“continuous thrust strategy”**

General review Low thrust orbit transfer strategies

- **Strategies from LEO to GEO: Spiral strategies allow the transfer without need of any optimization tool.**
  - **The well known Edelbaum equation (coming from optimization considerations) provide the ΔV needed between circular orbits at different inclination when** ignoring possible service interruptions.

$$\Delta V = \sqrt{V_0^2 - 2VV_0 \cos(\frac{\pi}{2} i) + V^2}$$

- However, when the power on board is not fully available for the thrusters (i.e. during eclipses), one cannot ignore the possible service interruptions (batteries may be not enough large) :
  - Thrust has to be switched off ➔only part of the orbit can be used for thrust
  - Major effect ➔orbit transfer degrades… ➔ increase eccentricity intermediate orbits
  - Thus the general problem about the optimal thrust orientation for minimizing the duration, starting from an intermediate elliptical orbit for reaching circular GEO orbit ➔ that goal fall into Koppel patents claims as pointed before
- **Finally: GTO to GEO; LEO to GEO ➔ "continuous thrust strategy"**

Optimiser: Mipelec (from CNES)

- Fast and very robust tool derived from S. Geffroy Thesis
  - Freely available on the CNES Website
- The spacecraft **thrust orientation** is commanded continuously during the transfer in order to **minimize the total transfer time**.
- 7 variables (including mass) are normalized by a small parameter (➔only low thrust)

$$\varepsilon = \frac{Thrust_{\max} \cdot Sma_{final}^{2}}{Mass_{initial} \cdot \mu_{Earth}}$$

Sma: semi major axis

- The full problem is reduced at first order in ε
  - Independent variable: Time, then True longitude finally becomes a New variable with fixed start/end: 0 to 1 for compatibility with TPBVP routine
  - The full optimisation problem is reformulated introducing an Hamiltonian and the minimisation of the Hamiltonian provide the optimal thrust and direction.
- Finally equations are averaged wrt fast rotating variable but keeping the command explicitly dependant on that fast movement.
  - But drawback no initial conditions on the true anomaly (always=0°)

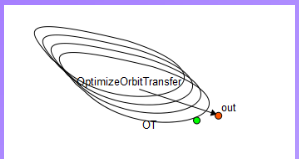
TPBVP: Two points boundary value problem for solving differential equations

Optimiser: Mipelec (from CNES)

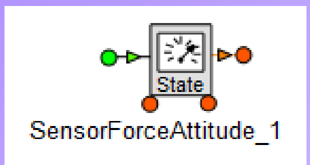
- Mipelec is a Fortran program
  - Few changes to make it compatible with Windows Fortran
  - Can be used with graphic tools for checks (Excel or 3D tools)
- But for being usable within a system tool like EcosimPro ESPSS ➔ transformation of the program into a function (a Fortran Subroutine called OTEP) is best suited
  - Function with input orbit and final orbit parameters
  - Output number of points with thrust direction and true anomaly for number of timesteps between given initial time and final time
  - Include that function in a Fortran lib (or archive) for declaration purpose within EcosimPro®
- Hence the call function OTEP can be performed anytime when needed by a component of EcosimPro ESPSS for initialization or updating the orbit transfer

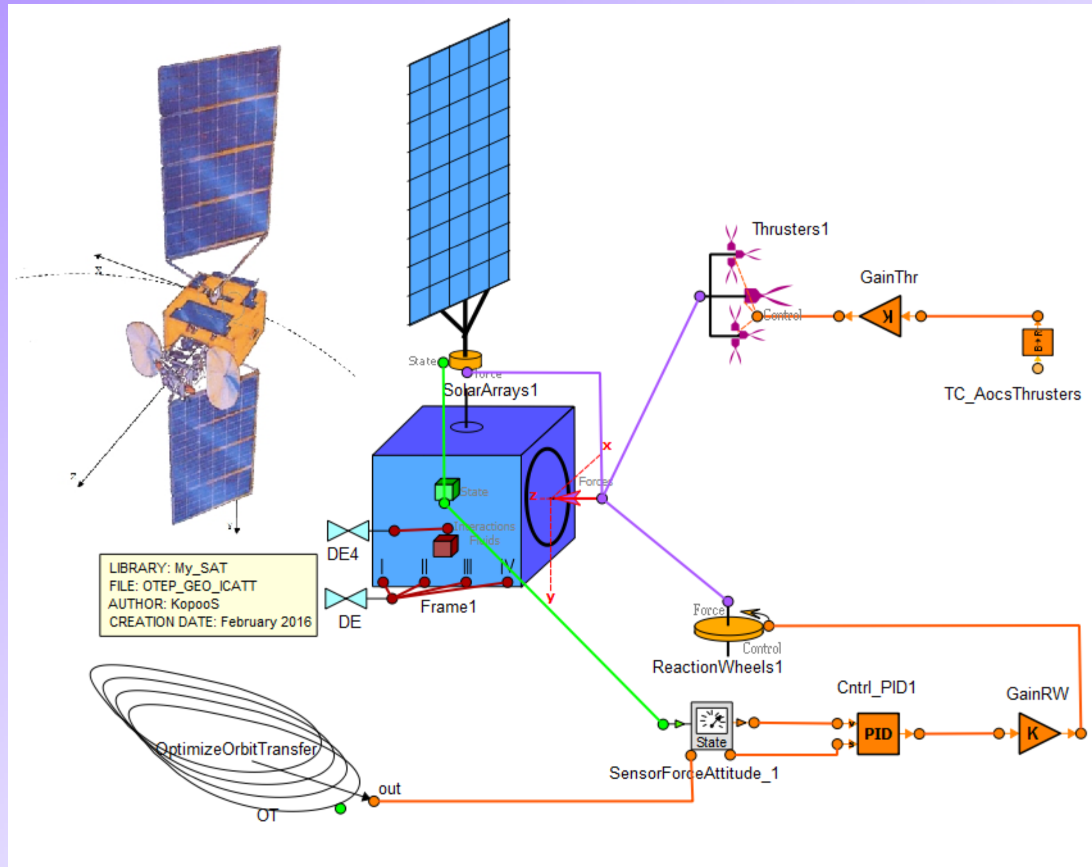
Integration Optimiser into EcosimPro® ESPSS

- Use of ESPSS-SATELLITE lib
- + updates
  - Component **OT** with DATA for the wanted Initial and Final orbit
    - **It call the FORTRAN Lib fucntion OTEP when the data change and when angle wrt perigee=0 (true anomaly=0)**



  - Component **Sensor**
    Receive the State and the best orientation commands from **OT** (and send back **OT** the “angle%perigee” or events for the timings);
    Feed the PID (for the reaction wheels for getting the wanted attitude)

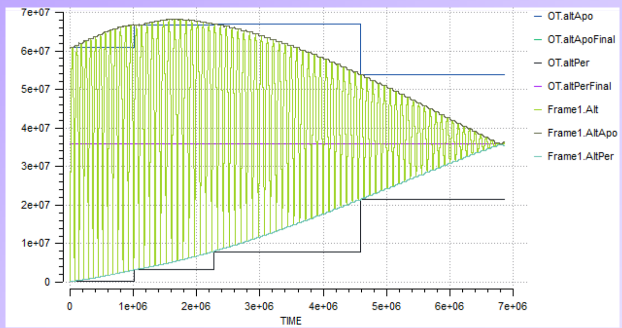


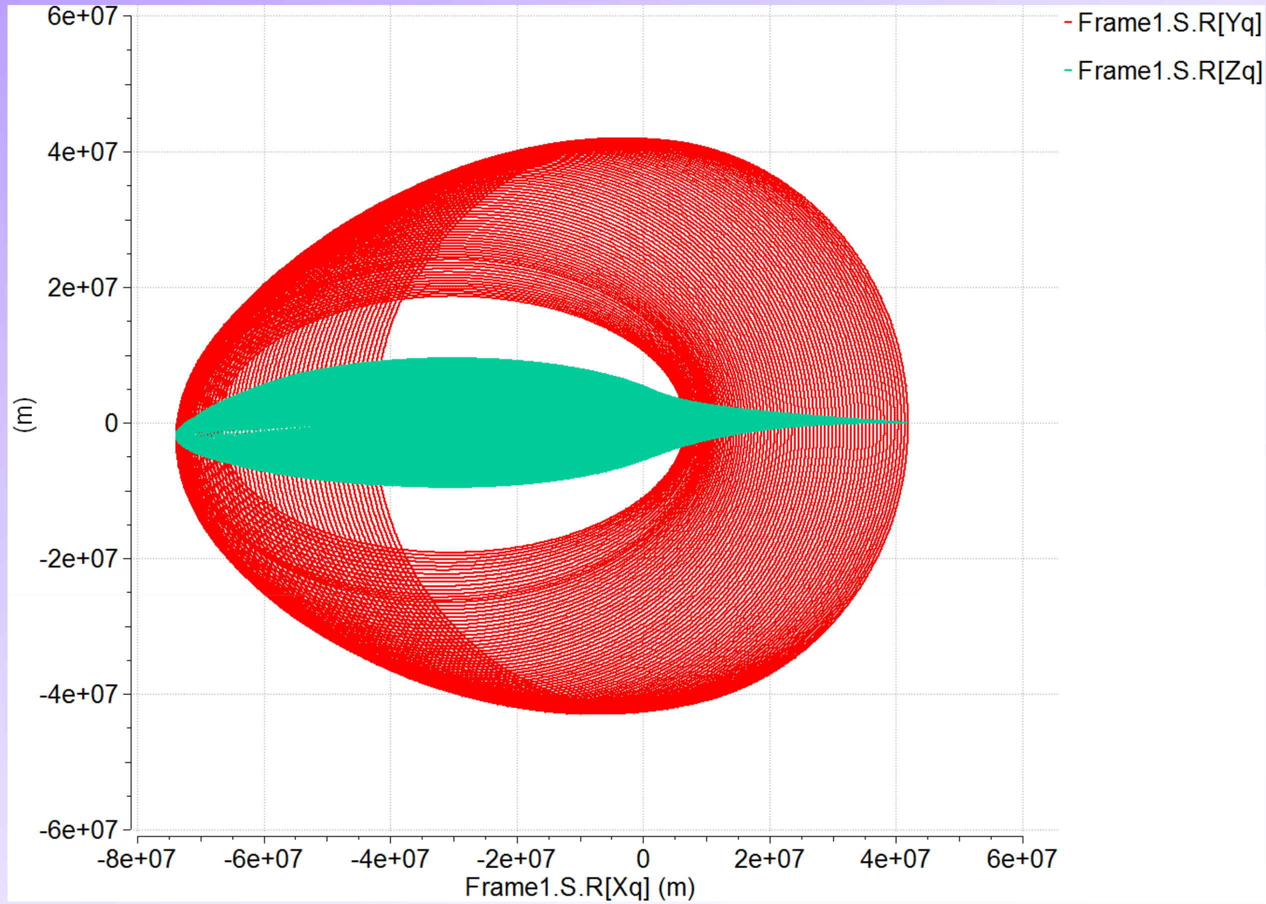


- Logic like a kind of “Bellman principle” : to optimise the "time to go" from every starting point with such optimiser that do not take any constrains or perturbations.

Example of realistic optimised EP transfer

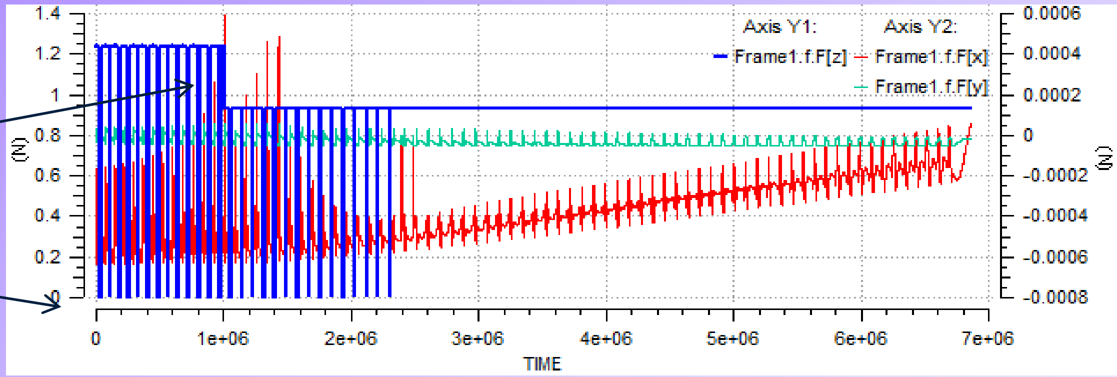
- Trajectory
  - Super GTO to GEO, inc=27°
  - Hall effect thrusters
  - Two levels of Thrust/Isp
  - All perturbations, including worst case eclipses
  - Only 4 calls to the optimizer (init, 1 call at thrust level change, 2 updates)

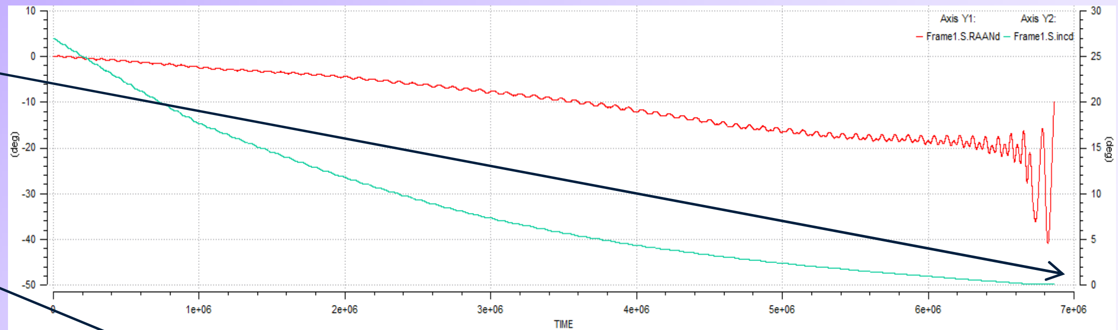


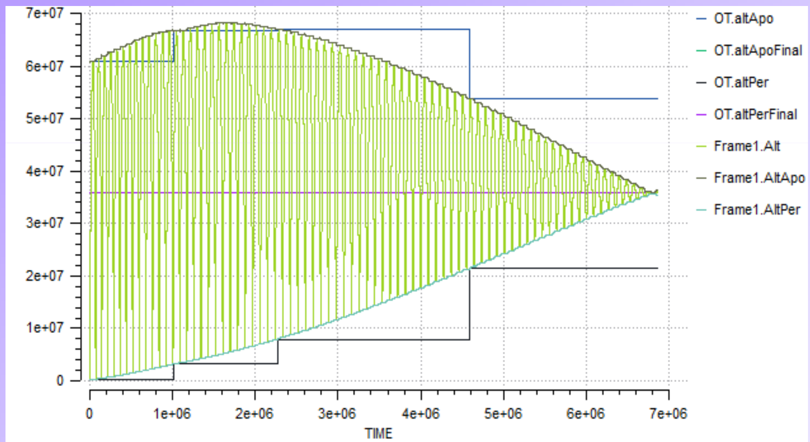


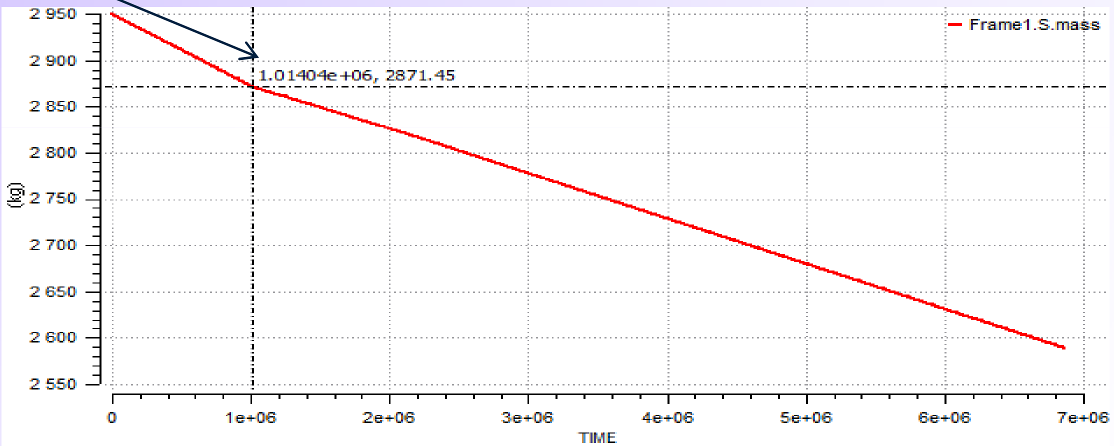
Example of realistic optimised EP transfer

- **Trajectory details**
  - **2nd thrust level at 10^{6}s**
  - **Thrust switched off during part of eclipses**
    - + perturbing forces
  - **Inclination reaches 0°**
  - **Mass and Isp levels**
  - **Altitude per/apo**







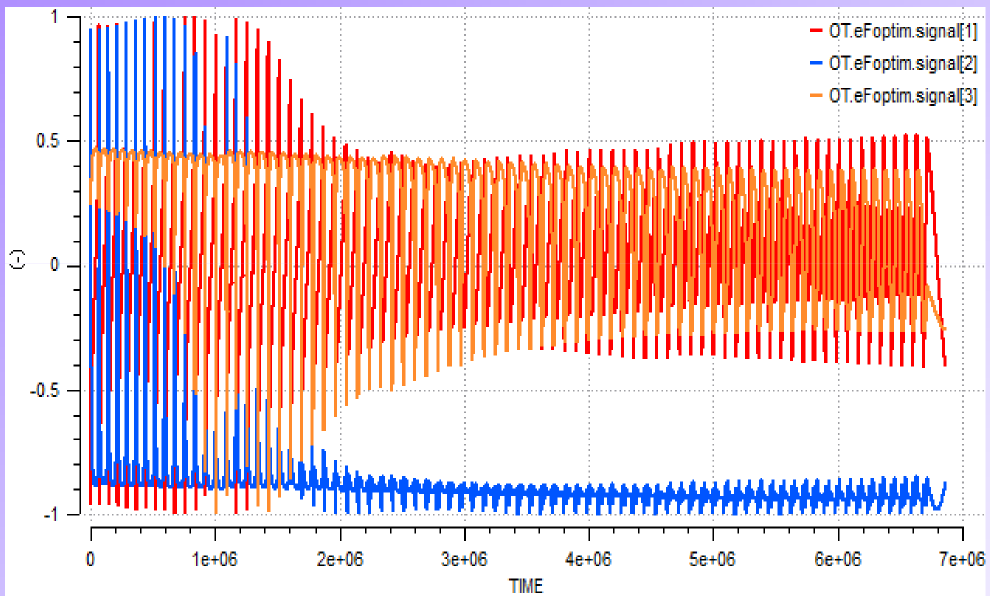


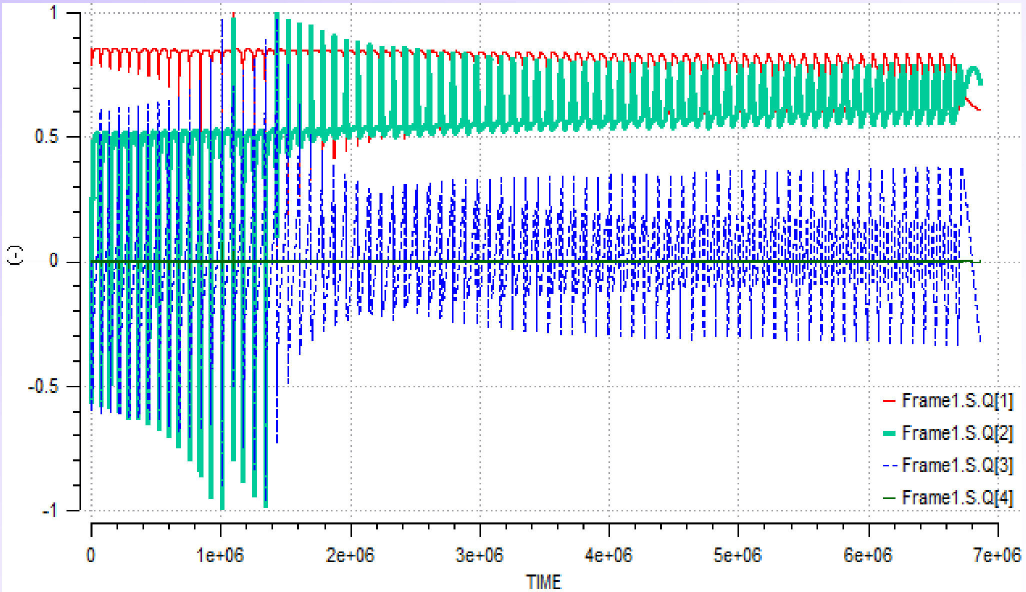
KopooS

Example of realistic optimised EP transfer

- **Trajectory details**
  - **Unit thrust vector components in ECI**
  - **Only 3 components but difficult to see anything obviously**
  - **…Better to see the quaternion➔**

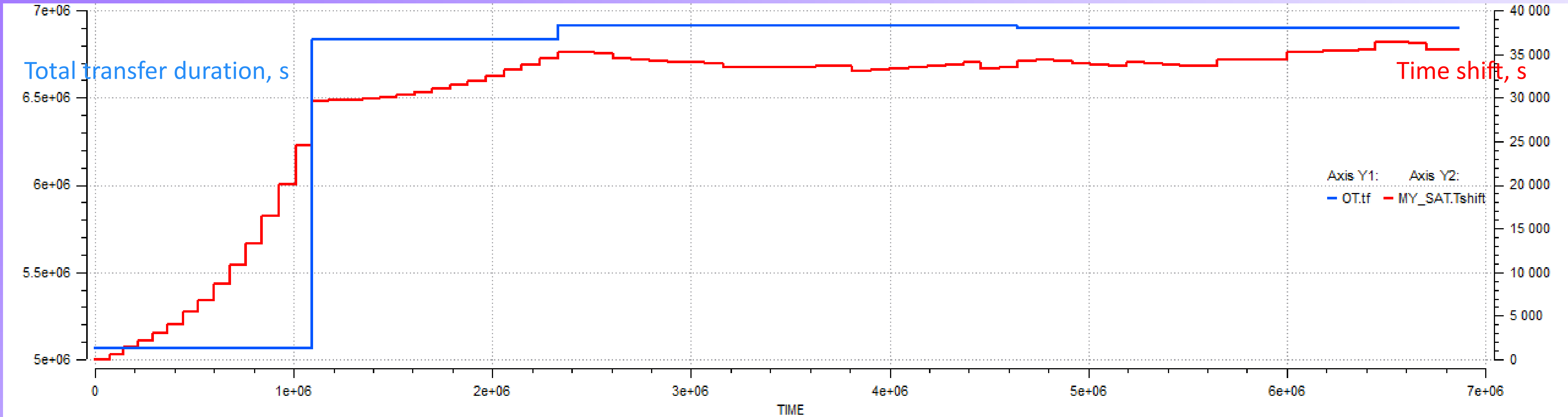
- **Trajectory details**
  - **Quaternion: 4 components**
  - **Very clear change of orientation at 1.5 millions s**
  - **The last quaternion Q4 is always null**
    - **Because that is the thrust axis**
    - **No need to have any rotation around it**





Example of realistic optimised EP transfer

- **Comments on synchronisation :Two aspects are to be pointed out**
  - **Quite large changes in optimized final time due to the 2nd thrust level**
  - **Synchronisation at the perigee passage (Tshift) reaching about 10 hours**



- **No attempt to perform more updates of the optimizer: results very good**
- **Of course, thanks to the very fast run, one can perform it at every orbit…**
- **Delta V needed 2337 m/s versus 2306 m/s theoretically: only +0.94% lost**

Conclusions

- **Review of Low Thrust orbit transfers strategies**
- **Successful integration of optimizer into EcosimPro® library**
  - **Transform a FORTRAN program in a subroutine Optimiser**
  - **Following some simples rules for being EcosimPro® compatible**
  - **The main advantages is that the “simple” FORTRAN program can be much more sophisticated in EcosimPro®**
    - Add perturbations: Sun-Moon, Earth pole flatness, Sun pressure, Eclipses
    - Changes allowed during transfer (e.g. Mid-course update) by running the Optimiser “in live” during the simulation
  - **To optimise the "time to go" from every starting point with such optimiser that do not take any constrains or perturbations.**
- **In conclusion: the integration into EcosimPro®**
  - **Make the Fortran tool much more operational and professional**
  - **Allow to simulate in “virtual” real conditions ➔ And get valuable results**
- **Continuous EP orbit transfers are for now…**

Thanks for your attention

Questions?

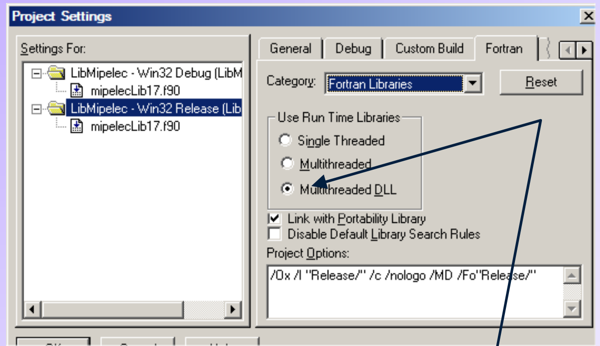
Ackowledgments

- The research leading to these results is a KopooS funding

Fortran Program to Library for EcosimPro®.

- **FOR➔ SUBROUTINE OTEP(prec, Pr0, Pr1, Fmax, m0, Isp, L1, DL, pb, auto, P, NBPTS, Big, errLib, time)**

  ! program mipelec Low-thrust optimisation program Copyright CNES
  ! Sophie Geffroy, Jean Fourcade
  INTEGER pb, auto, NBPTS
  REAL*8 prec, Pr0(5), Pr1(5), Fmax, m0, Isp, L1, DL, P(7), Big(5, NBPTS), errLib, time



- **Tips:**
  - **NO strings allowed for OUTPUT of Fortran function lib for EcosimPro®.**
  - **For INPUT permitted types for external FORTRAN functions are limited: only REAL, INTEGER, STRING, FUNC_PTR and ARRAY**
  - **Fortran arrays inversed in EcosimPro® :**

    Fortran Big(5, NBPTS) ➔ EcosimPro® Big(NBPTS, 5)
  - **Compile with MultiThreaded DLL Option (under Fortran PowerStation 4.0)**
  - **In FORTRAN, variables are always passed by reference (not by value)**
  - **Add a trim for matching timings between FORTRAN time and ECO time events**

- **ECO➔"FORTRAN" FUNCTION NO_TYPE OTEP(REAL prec, REAL Par0[5], OUT REAL Par1[5], REAL Fmax, REAL m0, REAL Isp, REAL L1, REAL DL, INTEGER pb, INTEGER Auto, OUT REAL P[7], INTEGER NBPTS, OUT REAL Big[NBPTS,5], OUT REAL errLib, OUT REAL time) IN "LibMipelec.lib" or "LibMipelec.a"**
- **Before using into EcosimPro®, place the "lib" or "archive" in C:\EcosimPro645419\USER_LIBS\MY_SAT\lib\compiler**