# Integrated Solution for Rapid Development of Complex GNC Software

J.-S. Ardaens & G. Gaias

Deutsches Zentrum für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

# Motivations

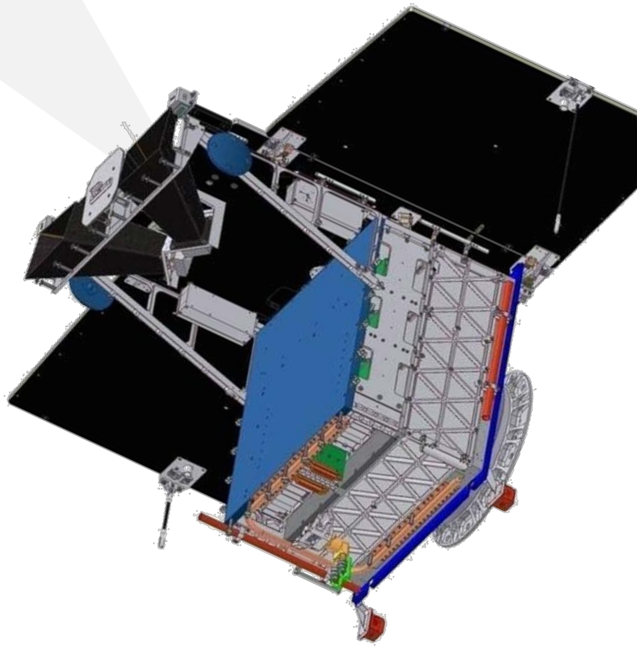
- The possibility to test a system in-orbit is rare and thus precious

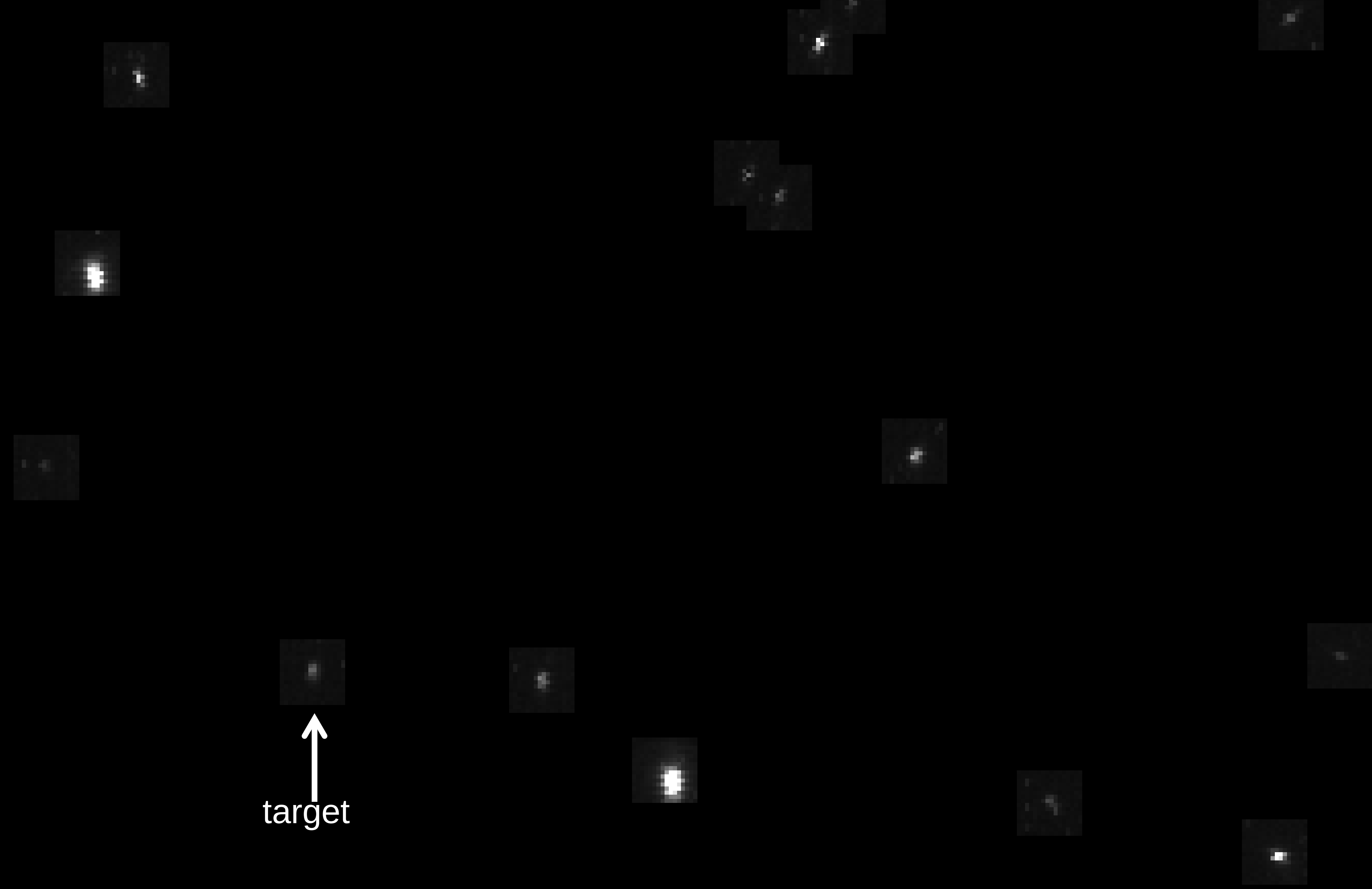
  $\Rightarrow$ unique opportunity to gain flight experience and improve the technology readiness level
- Might be a difficult and risky endeavor in the presence of constraints affecting the available resources:
  - Limited human resources
  - Limited development time
  - Limited time slot in orbit for validation and tests


- Need for tools and processes to speed up the development time without affecting the quality

# The AVANTI Experiment

- Autonomous Vision Approach-Navigation and Target Identification
- Spaceborne experiment using the DLR's **BIROS** satellite (launch in 2016)
- Paves the way for future on-orbit servicing missions
- Picosatellite embarked and ejected in orbit
- One star tracker employed as far-range camera to track the picosatellite
- Goal: approach to a non-cooperative object
  - fully autonomously
  - safe and fuel-efficient manner
  - low-cost sensor
- Based on the know-how gained during the ARGON experiment (May 2012) using the PRISMA formation-flying demonstrator



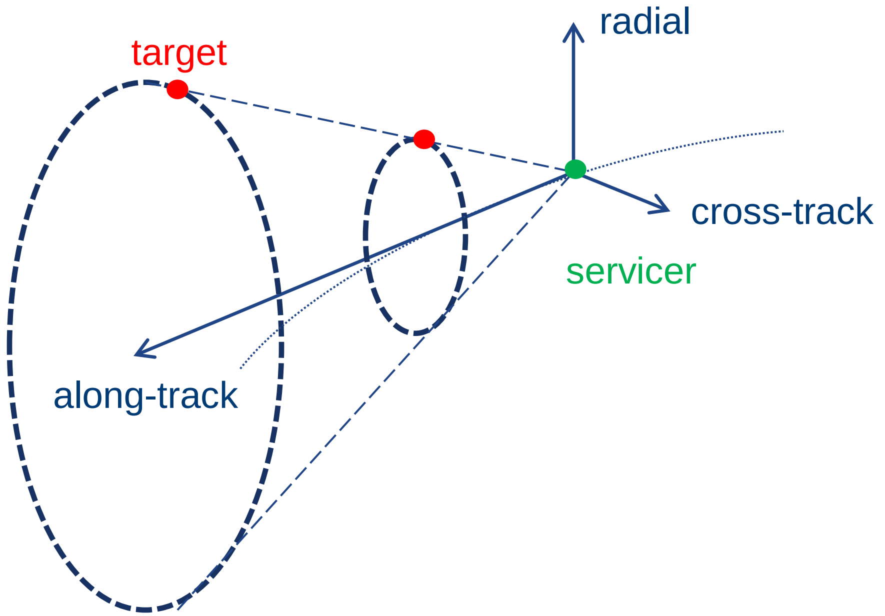



how to recognize a target object
at 30 km distance

image from the PRISMA mission during the ARGON experiment

# Angles-Only Navigation

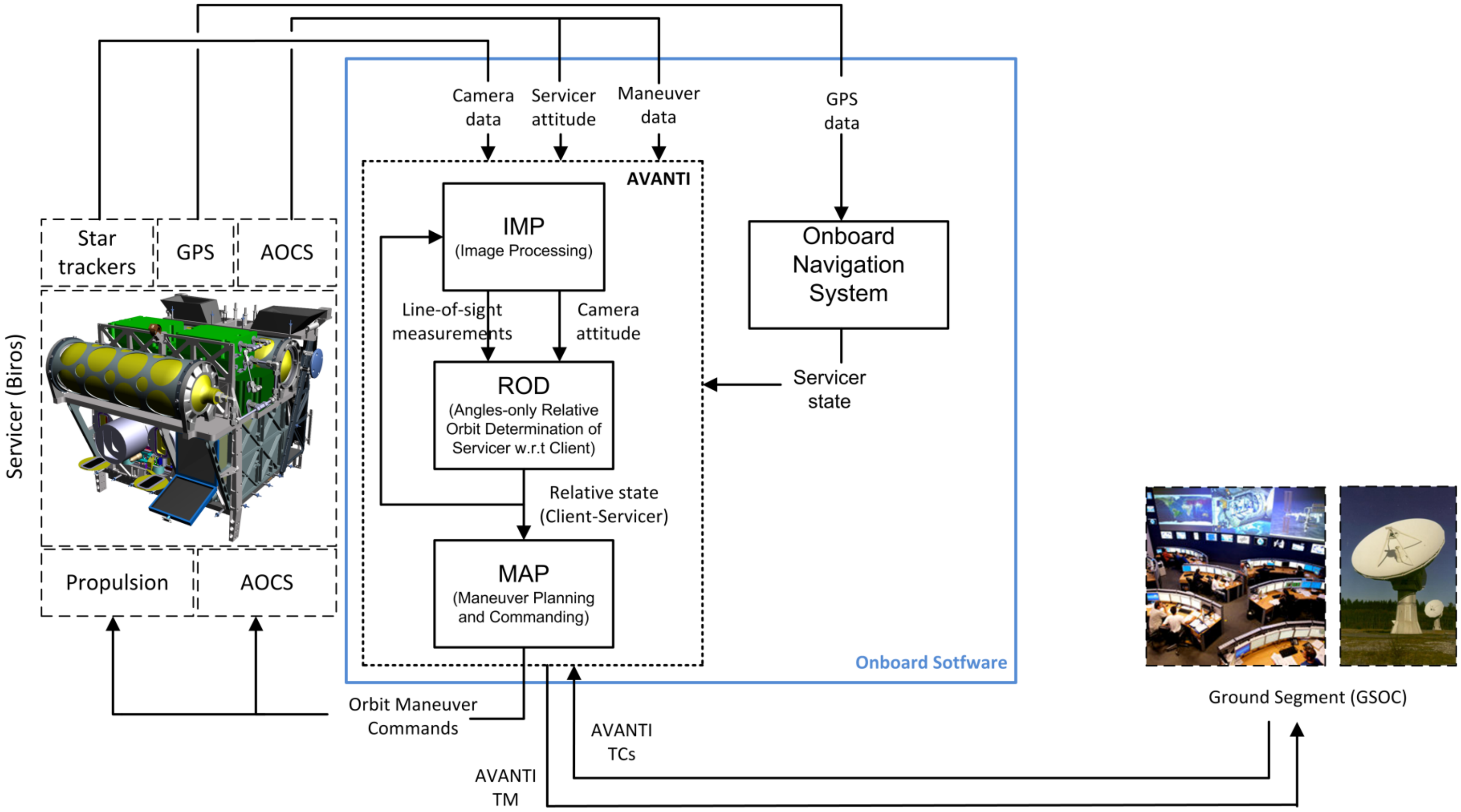
- No range measurement ⇒ very **weak observability**
- Infinity of solutions corresponding to a measurement profile
- Calibrated maneuvers are needed to improve the observability

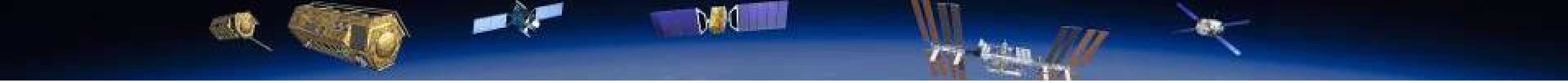


# Challenges

- **Reduced resources**
  - Small team (2 people)
  - Short development time (3 years)
  - Only one slot in orbit after the separation $\Rightarrow$ limited possibilities of software patch
  - Limited onboard computer
- **High system complexity**
  - Not easy to distinguish the target at far range from the surrouding stars $\Rightarrow$ dedicated algorithms are required for target detection
  - Angle-only navigation is weakly observable $\Rightarrow$ need for maneuvers to improve the observabilty
  - Maneuvers are part of a rendezvous profile which is computed onboard autonomously
  - The approach has to be safe (**risk of collision**), fuel-efficient and robust against contigencies
  - Pointing the star tracker towards the CubeSat affects the thermal and power budget
  - **BIROS** does not have 3D maneuver capabilities $\Rightarrow$ need to rotate the spacecraft during the maneuver $\Rightarrow$ loss of visibility
  - Visibility also affected by eclipses and camera blinding due to the Sun
  - High differential drag (500km altitude) impacts the navigation and guidance algorithm

# AVANTI Software Architecture





# Integrated Development Solution

- Early definition of interfaces required for the ground segment
- Early prototype required for satellite integration
- But the design and implementation of such a complex and novel GNC algorithm is time consuming

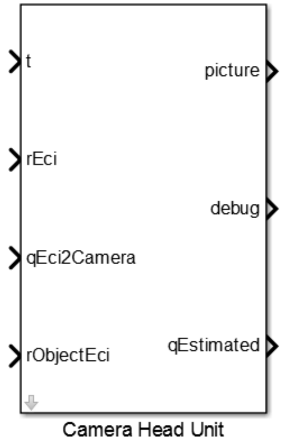
  ⇒ attempt to conduct algorithm design, implementation, integration & documentation simultaneously
- Problem: tools & development environment are not compatible
  - Simulink Environment for GNC Design & Development
  - Target OS: RODOS operating system (C++)
  - Miscellaneous formats to exchange data with partners
- Solutions retained for the development of AVANTI
  - **Interfacing Simulink directly with the flight software**
  - **Interface definition as unique meta data source**

# Simulink Simulation Environment

- Design and validation of the algorithms (target identification, robust angle-only navigation, autonomous rendezvous) using the **GSOC Multi-Satellite Simulator**
- Successfully employed for the design and validation of past formation flying experiments
  - TanDEM-X Autonomous Formation Flying System
  - GPS-Based Spaceborne Autonomous Formation Flying experiment on the PRISMA technology demonstrator
- **Simulink-based** environment comprising high-fidelity models:
  - Orbit propagation including orbital pertubations (drag, solar radiation pressure, third body perturbation)
  - Models for environment (Earth orientation parameters, position of the Sun, eclipses)
  - Implementation of attitude profile (target pointing, thruster firing mode, Earth pointing)
  - Sensors and actuators (camera, model)

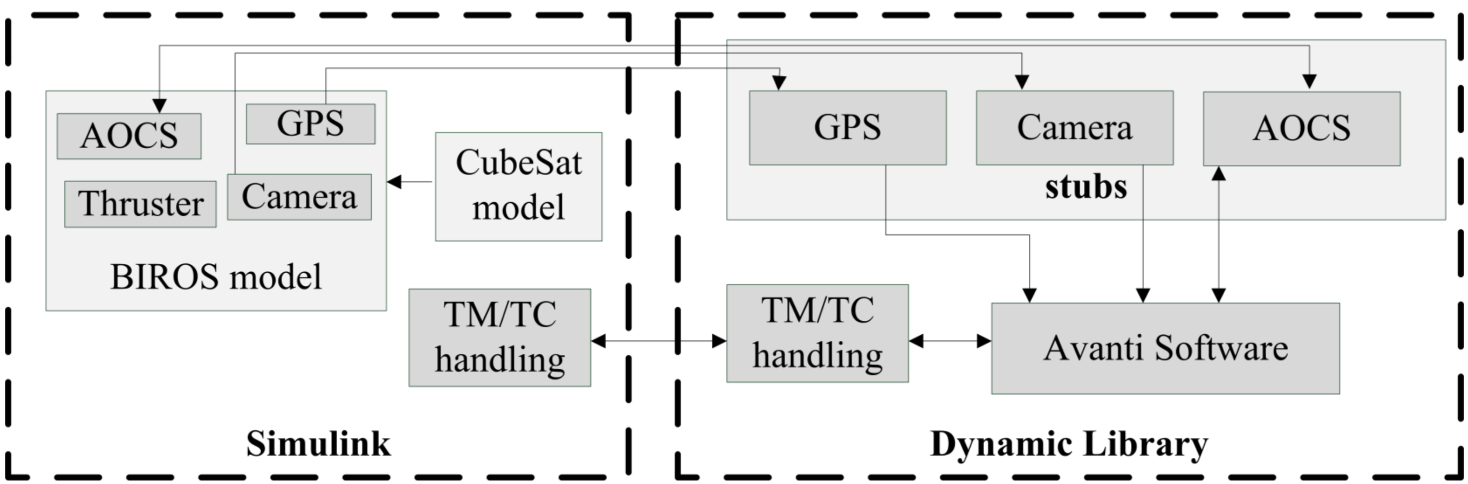
# Example: camera model

- High Fidelity Model required to prevent any bad suprise in orbit
  - image distortion
  - aberration
  - CCD model
  - radiometric model of the target
  - camera anomaly (hot spot)



# Interfacing Simulink with the Flight Software

- The flight software relies on OS-specific functionality (threading, timing, interprocess communication) and expect the presence of other components (like sensors and actuators).
- The flight software is compiled as independent dynamic library, based on a special library which emulates part of the RODOS functionalities on the host computer
- Components of the bus satellites are replaced by stubs fed by data coming from the Simulink model



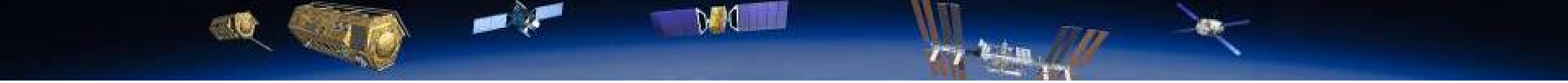
# Thread Scheduling

- Simulink takes care of task scheduling
- The flight software is written in the form of an infinite loop

```
class MyApplication: public Thread {
public:
  MyApplication (): Thread("name", priority) { }
  void init() { /*initialization*/}
  void run() {
    TIME_LOOP(offset, period) {
      /*work*/
    }
  }
}
```

- Retained solution: rewrite the TIME_LOOP function to be triggered by Simulink

```
#define TIME_LOOP(begin,period) for (count=0; waitForTrigger();count++)
```



# Interface Definition from a Unique Data Base

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | TC-Name | Parametername | Description | Format | ByteNr | |
| 2 | FILTER_INITIALIZE | p0_1 | Initial Position Vector - 1. component | double | 0 | |
| 3 | FILTER_INITIALIZE | p0_1 | Initial Position Vector - 2. component | double | 8 | |
| 4 | FILTER_INITIALIZE | p0_1 | Initial Position Vector - 3. component | double | 16 | |
| 5 | FILTER_INITIALIZE | threshold | Data editing threshold | ushort | 24 | |
| 6 | | | | | | |

interface exchange

```
class TC_FILTER_INITIALIZE
{
public:
  double p0[3];
  unsigned short threshold;
}
```

code generation

```xml
<telecommand>
  <name>FILTER_INITIALIZE</name>
  <description>Initialize the filter</description>
  <id>123</id>
  <parameter>
    <name>p0</name>
    <description>Initial Position Vector</description>
    <unit>m</unit>
    <dataType>double</dataType>
    <size>3</size>
  </parameter>
  <parameter>
    <name>threshold</name>
    <description>Data editing threshold</description>
    <unit>m</unit>
    <dataType>unsigned short</dataType>
    <size>1</size>
  </parameter>
</telecommand>
```

| Source | Output | Description |
|---|---|---|
| TM_GNC_LIDAR_IMAGE | | |
| Lidar | image | image taken by the LIDAR |
| TM_HOUSE_KEEPING | | |
| Lidar | status | status flag |
| Filter | t | Epoch of the state vector |
| Filter | y | State estimate |
| BIROS_HOUSEKEEPING | | |
| Spacecraft | unknown | unknown |
| Spacecraft | GPS_STATUS | status of the GPS |
| Spacecraft | ALAMANAC_WEEK | almanach week |
| Spacecraft | ONS_STATUS_FLAGS | status of ONS |
| Spacecraft | unknown2 | unknown |
| Spacecraft | ONS_WGS_POS_X | x-position |
| Spacecraft | ONS_WGS_POS_Y | y-position |
| Spacecraft | ONS_WGS_POS_Z | z-position |

GNC console

## Telecommand Description

### TC_FILTER_INITIALIZE

**Description**

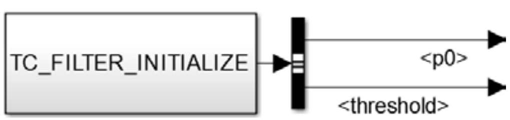
Initialize the filter.

**Parameters**

| Name | Description | Size x Data Type | Unit |
|---|---|---|---|
| **p0** | Initial Position Vector | 3 x double | m |
| **threshold** | Data editing threshold | 1 x ushort | m |

documentation

Simulink Bus



# Conclusion

- Development and integration time can be reduced by
  - Embedding directly an exogen flight software in Simulink
  - Generating all interface products automatically from a unique data base
- See you in 2016 for the conduction of the AVANTI experiment!