GNC 2011
8th International ESA Conference on Guidance, Navigation & Control Systems
Karlovy Vary, Czech Republic 5 - 10 June 2011
esa European Space Agency

FORMATION FLYING TESTBED AT DLR'S GERMAN SPACE OPERATIONS CENTER (GSOC)

G. Gaias[(1)], J.-S. Ardaens[(1)], S. D'Amico[(1)]

[(1)]Deutsches Zentrum für Luft- und Raumfahrt (DLR), German Space Operations Center (GSOC)
Münchner Str. 20, 82234 Wessling, Germany.

gabriella.gaias@dlr.de, jean-sebastien.ardaens@dlr.de, simone.damico@dlr.de

ABSTRACT

This paper describes the Formation Flying Testbed (FFTB) developed by the German Space Operations Center (DLR/GSOC) and presents key results from its usage in the frame of formation flying missions currently operated at GSOC.
The FFTB constitutes a mission-independent framework for the design, implementation, rapid prototyping, testing and validation of advanced guidance navigation and control (GNC) systems for multi-satellite missions. It is intended to support all phases during the lifetime of a GNC system, from the preliminary design to the system functional and performance verification to the post-facto analysis during mission operations.
The realization of the FFTB takes advantage from the achievements and the experience gained during the development activities of some pioneers autonomous formation flying missions like PRISMA and TanDEM-X. In particular the DLR's GNC contributions to these missions have been validated making use of the FFTB described in this paper.
The complete project is implemented in a Matlab/Simulink environment, which offers model-based capabilities, a user-friendly interface, high modularity, and is well-known among the aerospace community. The applications of interest can be executed on different platforms in a fully consistent manner. As illustrated in the paper this allows a seamless transition between offline simulations using flight data and real-time hardware-in-the-loop tests comprising real GPS receivers and a 2x12 channels Spirent GSS7700 GPS signal simulator. Furthermore the complete GNC application can be ported to a Real Time Executive for Multiprocessor Systems (RTEMS) environment in a LEON-3 FPGA board, representative of spaceborne onboard processors.
As a demonstration of its versatility, the FFTB has been recently integrated with the DLR's European Proximity Operations simulator (EPOS 2.0) to enable the verification of algorithms for real-time rendezvous and docking.
As shown during the PRISMA and TanDEM-X missions, the FFTB can be used to reliably verify the behavior of on-board software in-orbit. In this case orbit and attitude propagation functionalities as well as sensors and actuators emulators are simply substituted by Simulink functions able to read real flight data from the telemetry stream. Since the flight software is fed with the same inputs received in orbit, the FFTB can be used to support the tuning of the on-board algorithms. Finally flight data retrieved from the ongoing missions give the possibility to cross-validate the environment emulator of the FFTB. To this end real observations are compared with the simulated environment produced by the models employed in the FFTB.

1. MOTIVATIONS

Precise and robust guidance, navigation and control (GNC) of spacecraft are fundamental for successful multi-satellite systems. When designing a GNC system, key aspects like performance, robustness, integration within the space segment or operational utilization need to be analyzed already during the preliminary design and consolidated throughout all the project phases. These analyses require dedicated simulation and validation tools. The scope of functionalities provided by these tools can be as broad as the quick insight into the behavior of the algorithms, the validation of interfaces, the realistic assessment of performance, the inclusion of specific hardware components in the loop, or the analysis of usage of resources.

The Formation Flying Testbed described in this paper aims at answering to this need by providing a complete framework for the design, implementation, rapid prototyping, testing and validation of complex GNC systems [1]. The objective is to provide a modular and flexible development environment used for every kind of formation flying projects and research activities; no matter how mature they are and which objective they follow. Thanks to this philosophy, all projects benefit from the development and heritage coming from the other projects based on the same framework and, as a direct consequence, inherit a validated and powerful simulation environment as well as a collection of precious analysis tools.

The main challenge is to provide a collection of generic and validated models, functionalities and tools which can interoperate together and can be easily adapted to any mission and, at the same time, to always grant the possibility of mission-specific additional development. This challenge has been solved by implementing the Formation Flying Testbed under the Simulink [2] environment, widely spread among the GNC community, which offers the advantage of modularity, flexibility, and easy porting to other platforms.

The Formation Flying Tested benefits from the heritage of the two pioneer missions in the field of autonomous formation flying PRISMA [3] and TanDEM-X [4]. It takes advantages of numerous models and tools developed during the different phases of the project and makes intensive use of the existing models provided by the DLR/GSOC's C++ GHOST library [5]. Special effort has been made to collect the experience gained during these projects and to harmonize the tools and models, strengthening the overall reliability and user-friendliness of the testbed.

Moreover effort has been dedicated to include tools and functionalities suitable for other typologies of multi-satellite projects. The idea was to exploit all possible common features between pure formation flying and proximity operations missions, while keeping the same approach in the design process. On-orbit servicing activities, close-range autonomous relative navigation, rendezvous and docking, in fact, introduce various challenges in several areas of the mission design. Here, the capability to test the complete GNC plant, including a facility that operates in real-time and that runs hardware-based demonstrations, could constitute a key factor to achieve the technological maturity necessary to proceed with flight demonstrations and, eventually, a mission. This motivated us to interface the FFTB with the new European Proximity Operations Simulator (EPOS 2.0) facility. The result turned out into a flexible, application- and mission-independent facility capable to focus on the critical phase of separation ranging from 25m to 0m [6].

Therefore the Formation Flying Testbed has evolved into a more complete Multi-Satellite Simulator (MSS), whose structure, characteristics and possible applications will be explained in this paper.

2. OVERVIEW OF THE MULTI-SATELLITE SIMULATOR (MSS)

A. SYSTEM DESCRIPTION

The Multi-Satellite Simulator MSS is a mission-independent framework. As such it is designed to offer a high flexibility in order to be utilizable for a large variety of mission profiles and research topics. Focus has been given to the simplicity and transparency of utilization. To that end the complete project is implemented using Simulink, which offers a user-friendly interface, a high modularity and is usually well-known among the aerospace community. Furthermore it makes the complete system easily upgradeable and extendable.

The MSS is composed of Simulink libraries offering ready-to-use functionalities and models, which allows the quick and easy creation of highly sophisticated simulations. The Simulink [2] blocks composing the libraries are implemented using either C++ based S-functions or Matlab [7] Embedded functions. This choice enables the possibility for automatic generation of C/C++ code, which is of great interest when porting the models to other platforms. Within the space field, this approach has been successfully applied in the onboard software design of the SMART-1 (Small Missions for Advanced Research in Technology) Moon probe [8] and in the PRISMA Formation Flying mission [9].

Overall the Simulink blocks may be divided into seven categories:

- **Environment**: models describing the environment acting on a spacecraft.
- **Conversion**: functions for time and frame transformation.
- **Mathematics**: implementation of specific operations which are not provided in the standard Simulink libraries (e.g., quaternion algebra).
- **System and Interface**: components which ease the handling of data and the interoperability with other systems or simulations environments.
- **Sensors and Actuators**: models of spacecraft sensors and actuators.
- **Onboard Functionalities**: useful existing pieces of flight software coming from previous projects or newly developed.
- **Vehicles**: high-level models of spacecraft, built from the low-level models comprised in the libraries described above.

In addition, MMS offers a collection of tools, Matlab scripts or programs written in C++, whose purpose is to facilitate the preprocessing and treatment of data.

B. RAPID PROTOTYPING

The main driver underlying the design of the Multi-Satellite Simulator is the ability to conduct in short time scales realistic and meaningful analyses of a given problem. This is made possible by the strategy of having a collection of ready-to-use models and functionalities which are already validated. Another important aspect is the ability to port rapidly and to run the algorithms on dedicated target systems. The goal of rapid prototyping is not to provide flight versions of GNC algorithms, but to get quickly relevant information concerning the ability of the prototype to fulfil mission requirements and to evaluate the resources needed, in order to take the proper design decisions at the early stages of the development. To that end, special attention has been paid to ensure the compatibility of the libraries with Real-Time Workshop [10], the automatic code generation system of Matlab, which allows the instantaneous porting of existing Simulink models to other platforms. The operating systems for true real-time applications VxWorks [11] and RTEMS [12] are currently supported.

C. SOFTWARE MANAGEMENT

The high complexity and the continuous improvements of the MSS simulator require a rigorous software management. For this reason, the project is handled by a version control system. In addition to the obvious advantages brought by a version control system, like the easy tracking of individual contributions or the access to the complete history of changes for every file, this strategy offers the capability of keeping different versions of the MSS at the same time. This is particularly of interest for Simulink blocks, whose interfaces are inclined to change very often. The version control system allows the user to freeze the status of the libraries at a given time and to refer to one particular version when creating a new simulation based on these libraries. This ensures the continued availability und functioning of short term studies or past projects over years, even if their development has been stopped a long time ago. A graphical example is shown in Fig. 1. The past project is a Simulink model whose development is definitively stopped. In-between the libraries might have evolved. In order to avoid any compatibility breaking, the status of the libraries at the epoch of the model development has been frozen and the project is simply pointing to these versions. The current project instead uses the most recent (and supposedly most advanced) version of the libraries, while the recycled project reuses an old project and improves only part of it by using the newest version of one library.

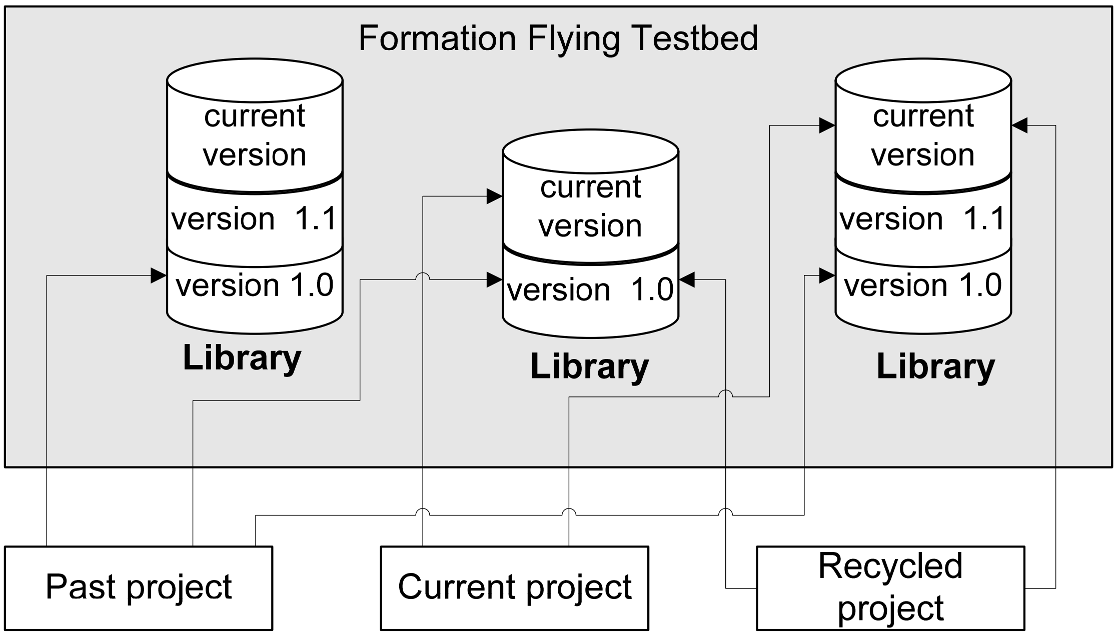

Fig. 1: *Example of software management.*

D. HARDWARE-IN-THE-LOOP CAPABILITIES

A real-time hardware-in-the-loop, closed-loop testbed is of relevance for advanced functional and performance tests. The Multi-Satellite Simulator comprises special system components to enable the utilization of some hardware components in the loop. Currently the following hardware is supported:

- **Phoenix GPS receivers [13]**. The use of up to two single frequency GPS receivers in the loop is done by steering in real-time a GPS signal simulator (GSS) and connecting directly the GPS receivers to the outputs of the simulator. This ensures that the flight software is able to interface properly the sensors and able to handle the failures that may occur. In addition, ultimate assessment of navigation, guidance and control performances is achieved by avoiding any software model of the sensor.
- **LEON-3 board with RTEMS operating system [14].** The introduction of a target computer representative of the onboard computer on which the flight software will be running is of great help for the analysis of software resource usage. This is achieved by making use of the Matlab automatic code generation capabilities applied to the algorithms developed under Simulink to generate C++ code and compile it for the target computer.
- **EPOS 2.0 Facility [15].** The MSS has the ability to interoperate with the DLR's European docking and rendezvous simulation facility: the new European Proximity Operations Simulator (EPOS 2.0) to enable the demonstration of algorithms for real-time rendezvous and docking.

The inclusion of hardware units in the loop introduces special limitations and issues, such as the mandatory time synchronization and communication between components. Dedicated system functions have been developed to enable the synchronicity and interoperability between all the subsystems.

E. VALIDATION AND TESTING

The constantly growing complexity of the Multi-Satellite Simulator makes a rigorous validation and testing strategy mandatory. This is achieved on different levels. Unit testing applied to the elementary blocks composing the libraries guarantees that the single blocks execute properly their tasks and prevents them from any future software regression. System testing is instead intended to verify the MSS is able to fulfill system requirements related to the functionalities, performance, interfaces and resource usage. A collection of unit tests is associated to the libraries to ensure at any time the proper functioning of the testbed. In order to reduce the testing efforts, the unit and system testing is fully automated.
The ultimate validation of the testbed is achieved when comparing simulation results with real flight data. To this aim, flight data coming from the PRISMA [3] and from the TanDEM-X [4] formation flying missions, both launched in June 2010, can be used to verify the overall performance of the testbed. As a representative demonstration, in the sequel of this paper, the comparison between a simulation run to plan a primary PRISMA experiment and the actual telemetry gathered from the satellites in April 2011 is presented.

3. EXAMPLES OF APPLICATIONS

Thanks to its modularity and flexibility, the Multi-Satellite Simulator is able to support a large variety of applications. In order to help the reader getting an idea of the potential offered by the testbed, typical utilizations are summarized in the sequel.

A. CLOSED-LOOP GPS-BASED GNC SYSTEM FOR SPACECRAFT FORMATION FLYING

The design, implementation and test of GNC algorithms in a pure Simulink environment represent most of the applications of the Multi-Satellite Simulator. Fig. 2 depicts an example of such a project. In the selected demonstration case, the user aims at developing a GNC algorithm to control actively the relative position of two spacecraft flying on a high elliptical orbit. More precisely, the goal is to control the relative position when GPS data are available, i.e. at the perigee passage. The goal is to set up a realistic simulation environment in which the control functionality (blocks painted in cyan on Fig. 2) is implemented and tested. For this purpose, it is necessary to simulate the behavior of GPS receivers flying on two spacecraft, to derive the navigation solution based on the GPS observations, to compute the control action based on this navigation solution, and to include this control action in the propagation of the satellites. In order to speed up the development, the user makes intensively use of the models and functionalities available in the MSS.

- The orbit and the attitude of the both spacecraft are propagated using a generic, fully parametric model of the spacecraft (yellow blocks) coming from the **Vehicles** library. This block can be set to control automatically the attitude in any desired attitude mode (in order to ensure an optimal tracking of the GPS signal). The block takes as input a control action which is then used when propagating the spacecraft.

- The GPS receivers are emulated using already available software models of the GPS receivers (blue block). The models for GPS receivers and orbit and attitude propagations belong both to the library for **Sensors and Actuators**.
- In the current show case, the user is more interested in the control part of the problem. As a consequence, he prefers to reuse existing software for precise relative navigation based on GPS measurements, and makes use of the navigation filter developed in the framework of the PRISMA mission (orange block). This block belongs to the library for **Onboard Functionalities**.
- Small functionalities (magenta blocks) are in addition needed to ease the interfaces and extend the capabilities of Simulink. The small blocks implement the handling of the time using the GPS time format, while the big magenta block on the right is an improved version of the standard Simulink block used to save data in a file. These magenta blocks belong to the library for **System and Interface**.
- Finally, the green block is used to implement the conversion of direction cosine matrix to the quaternion representation. This block belongs to the library for **Mathematics**.

B. HARDWARE-IN-THE-LOOP, CLOSED-LOOP GNC SYSTEM FOR RENDEZVOUS AND DOCKING

The fact that the Multi-Satellite Simulator is implemented in Simulink offers the possibility to port easily the models to other platforms. In the following example, a GNC system for Rendezvous and Docking [6] is tested using the new European Proximity Operations Simulator (EPOS 2.0) facility (Fig. 3). For this purpose, a complete simulation environment comprising the propagation of the dynamics, the GNC algorithms and the models of the actuator is first developed with Simulink, as described in the previous section. Once the development is achieved, a block for interfacing the EPOS 2.0 facility is added and the complete model is translated into C/C++ and compiled for VxWorks using the Matlab Real-time Workshop and a GNU cross compiler. The model can then be run in real-time and steer the displacement of robots that mount true hardware sensors. The measurements of these sensors are finally fed back to the GNC system running in the VxWorks environment.

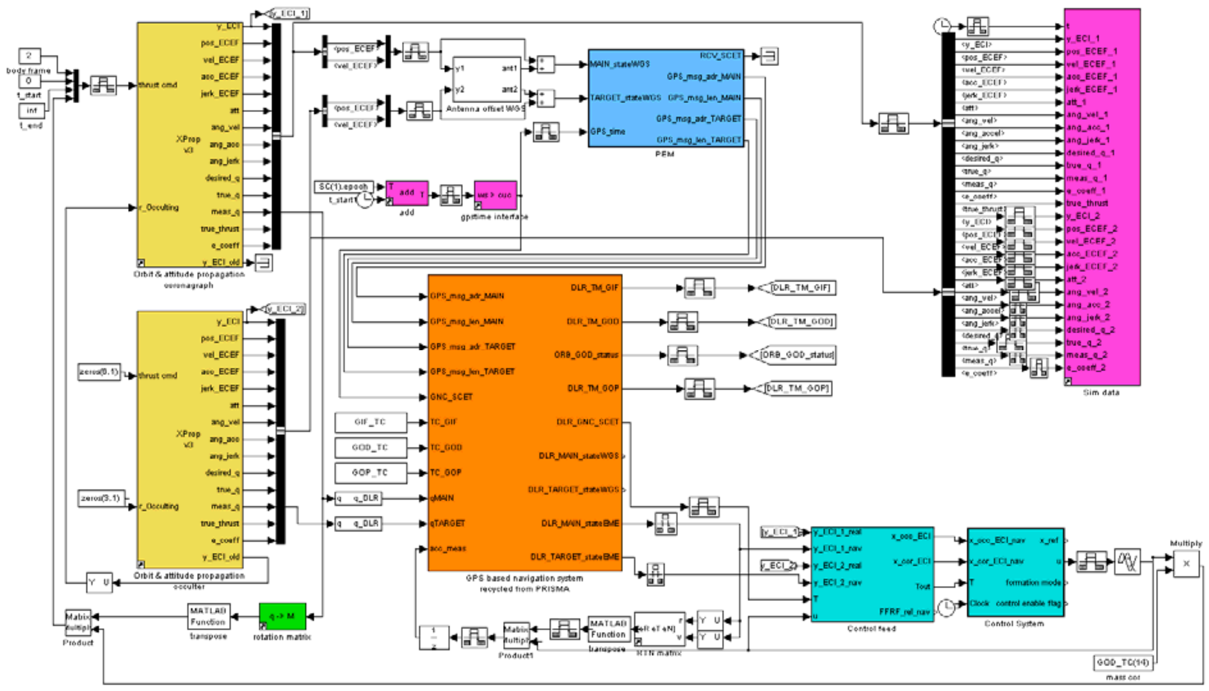

Fig. 2: Example of pure software simulation based on the MSS.

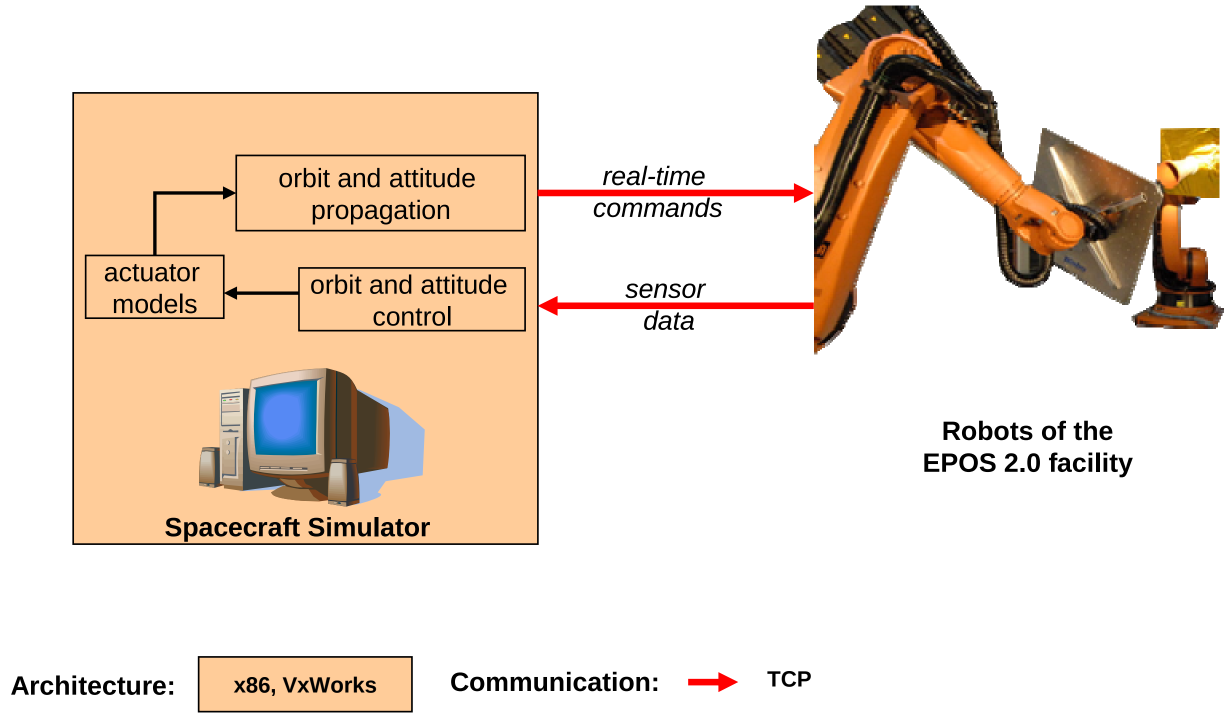

Fig. 3: Hardware-in-the-loop closed-loop GNC system for rendezvous and docking.

C. HARDWARE-IN-THE-LOOP, CLOSED-LOOP GPS-BASED GNC SYSTEM FOR DUAL SPACECRAFT FORMATION FLYING

At a certain phase of development, the user might be interested in using true GPS receivers in the loop instead of software emulations. The Multi-Satellite Simulator comprises several functions for the creation of GPS-based hardware-in-loop simulations. Among them, the block for steering a Spirent GPS signal simulator from a Simulink model s a key element which allows interfacing complex simulation environments with GPS receivers. In this case the MSS is not anymore comprised in a single software model but becomes a facility composed of several units, which need to work together. Other functionalities are implemented to facilitate the exchange of information through a network or to synchronize the different units. Fig. 4 depicts a possible configuration, in which not only the GPS receivers are used in-the-loop, but also target computers representative of the onboard computer. In this example, the spacecraft simulator is implemented using a Simulink model. Motion commands are sent in real-time to the computer steering the GPS Signal Simulator. This functionality as well as the mutual synchronization is ensured by a block for steering the GPS signal simulator. The GPS receivers process the radio-frequency signals coming from the GPS signal simulator and send the GPS observations to the onboard computers, on which the flight software is running. The control actions sent by the flight software are finally sent back to the spacecraft emulator, which uses advanced models of the dynamics and the actuator to propagate the translational and rotational motion. It is as well possible to emulate the sensors which are not physically in the loop in the spacecraft simulator and send the resulting simulated measurements to the flight software.

The choice of TCP as communication protocol between the system components allows the possibility to create easily additional applications to monitor and control the system. The Simplified Ground Segment depicted in Fig. 4 is for example intended to offer to the user a simple way to interact in real-time with the system, i.e. to send telecommands and receive telemetries, and thus to address some operational aspects when testing the flight software.

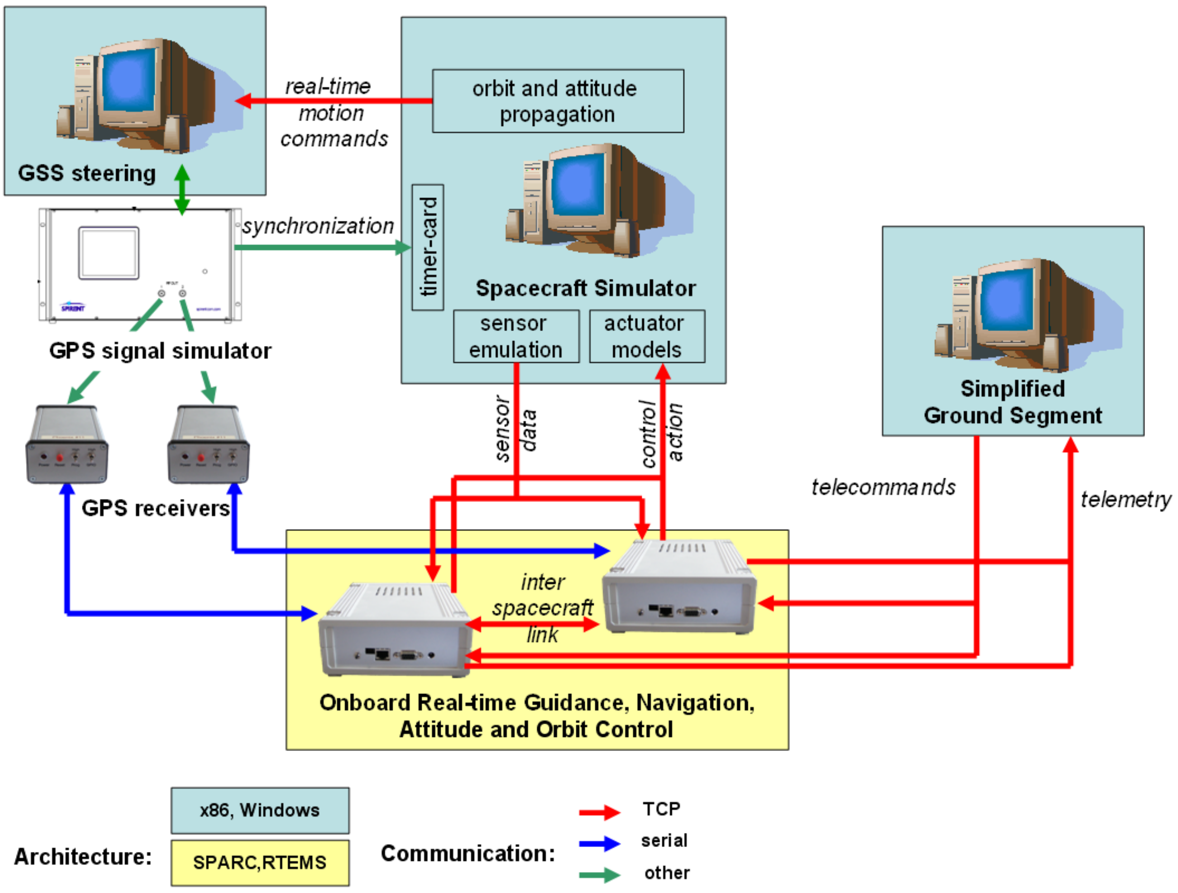

Fig. 4: *Hardware-in-the-loop, closed-loop GPS-based simulation environment.*

D. OPERATIONAL SUPPORT OF FLIGHT SOFTWARE

The efforts done to harmonize the components and tools as well as the flexibility offered by the testbed make very easy the creation of applications focused on dedicated purposes but sharing the same technical background. The Replay Tool developed for the post facto analysis of the DLR’s contribution to the onboard software of the PRISMA and TanDEM-X missions is a good illustration for this concept [16]. Based on the simulation environment used to develop and validate the flight software, the Replay Tool replaces simply the simulation part (orbit and attitude propagation associated with models for the sensors and actuators) with a Simulink block for reading actual flight data. As depicted in Fig. 5, the other blocks remain identical, which allows the rapid replay of the flight software for offline comparison. In this case, the flight software is fed with the same inputs as it was during the flight and it is checked that the off-line behavior is identical with the in-flight one. Such a tool allows the tuning of the software, whose goal is to analyze how would have behaved the flight software in the same conditions but with different settings.

4. EXAMPLE OF UTILIZATION OF THE MSS DURING THE PRISMA MISSION

In the previous sections the structure of the Multi-Satellite Simulator was described and an overview of its capabilities and possible utilizations was provided.
Each of these applications has been developed to support different projects. Subsequently, thanks to the design approach of the simulator itself, the whole simulator has evolved taking benefit from the heritage coming from each project. Among the various utilizations of the MSS, a great contribution to its development was provided by the design, testing and validation associated to the DLR’s contributions for the PRISMA [3] and TanDEM-X [4] missions. Particularly [17] gives an idea of the activities accomplished as it describes the software- and hardware-in-the-loop validation of the GPS-based real-time navigation system for the PRISMA mission.

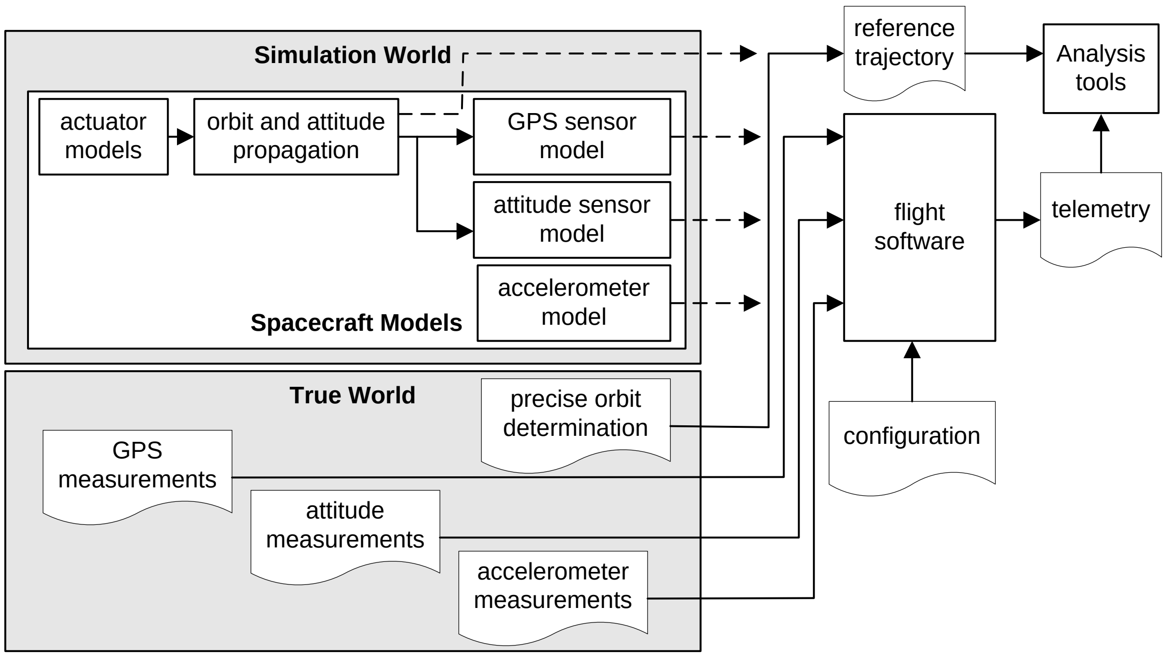

Fig. 5: *Schematic view of the Replay Tool.*

In this section, however, our attention is focused on an operational use of the Multi-Satellite Simulator. In this case, the MSS is employed during the real mission to plan possible scenarios for the experiment that is going to be executed by the satellites. Therefore the outcome of these simulations is the set and schedule of commands that the operators shall send to the spacecraft in order to fulfill the planned strategy.
In particular here it is shown how MMS was used to plan a portion of the second slot of the Spaceborne Autonomous Formation Flying Experiment (SAFE) which took place on the PRISMA mission in April 2011. The validation of the testbed, that is the assessment of the level of reliability of the predicted behavior, is then obtained by comparing the simulation results with the flight data gained during the mission itself.

A. EXPERIMENT PLANNING

The space segment of the Swedish PRISMA mission comprises the fully maneuverable Mango (150 kg) small satellite as well as the passive Tango (40 kg) sub-satellite which have been launched on June 15th 2010 into a Sun-synchronous orbit at 750 km altitude. Mango features a three-axis reaction-wheel-based attitude control and accommodates a hydrazine-based thrusters system that provides delta-v for orbit control. Tango applies a coarse sun- magnetic-based three-axis attitude control and does not have orbit control capability.
Among the various experiment sets within PRISMA, DLR developed and performed the Spaceborne Autonomous Formation Flying Experiment (SAFE) [18], which was intended to demonstrate fuel-efficient, collision free, accurate and robust relative motion control on a routine basis to fulfill the typical requirements of future distributed sensors for Earth observation like synthetic aperture radar interferometers and gravimeters. The SAFE experiment comprised two operational slots of Autonomous Formation Control (AFC), respectively AFC1 from September 20th to October 6th 2010, and AFC2 from March 17th to April 4th 2011.
During the AFC2 slot, numerous nominal relative orbits were commanded from ground to the AFC software on-board on Mango. Consequently AFC planned and executed maneuvers in such a way that those orbits were autonomously acquired and maintained over the prescribed time intervals. The detailed description of the AFC2 experiment together with the analysis of the flight data is described in [19].

The planning of AFC2, in terms of choice and timings of the nominal relative orbits, was done making an extensive use of the Multi-Satellite Simulator in its pure software configuration introduced in section 3.A. In particular MSS was configured as follows.

- The orbits of both Mango and Tango were propagated using a fully parametric model from the **Vehicles** library, in which the spacecrafts' characteristics were loaded. Nominal attitude was provided by a specific functionality that reproduces all the attitude modes offered by the PRISMA GNC segment. On top of this nominal behavior, attitude determination and control errors were introduced. The tuning of such error-models was accomplished using flight data already collected. Such a tuning revealed to be crucial for the plausibility of the simulations. Attitude errors, in fact, combine with the GPS signal tracking influencing the navigation accuracy, and, consequently, the control action needed to fulfill the aimed accuracy requirements.

- The GPS receivers were emulated using the Phoenix Receiver Emulator (PEM) model from the **Sensors and Actuators** library. The tuning of the GPS measurements noise also revealed fundamental to achieve an accurate representative behavior of the AFC flight software in space.
- The DLR's on-board software which implements the GPS relative navigation and the AFC orbit control functionalities was embedded in the simulation environment through the usage of S-functions. These functionalities belong to the library for **Onboard Functionalities**.
- A model of the Digital Video System Camera (DVS) mounted on the Mango satellite was also included. This is a new functionality offered by the **Sensors and Actuators** library. Here it was used to select when shooting pictures, in order to obtain images fruitful for future utilizations for testing tools for visual based navigation.
- Finally, simplified ground segment functionality was also included. It allowed giving as input to the AFC software block the geometries that were tested. At the same time, it organized the output data with the same structure of telemetry packets defined for the AFC experiment. As consequence, results could be plotted and analyzed with the same tools developed for monitoring the flight data.

The MSS was used to quickly generate reliable scenarios, necessary condition to face the typical decisions related to an experiment planning activity. In the case of AFC2, for example, the experimenter is interested in establishing sequence of geometries that could give him the maximum scientific outcome. The planning, however, must cope with eventual current operational constraints. According to them, reconfigurations might need too much Delta-v. Or satellites can ask more time to achieve proper conditions, thus introducing time shifts in the aimed plan.
Another crucial issue during planning is the estimation of the Delta-v consumption and the verification that it is compliant with the budget allocated for the experiment slot. Finally, MSS could be used for quick experiment re-planning, in case of unexpected behavior of the flight software or eventual contingency that could have occurred at system level.

B. VALIDATING THE SIMULATION WITH FLIGHT DATA

The data presented in this section refer to the planning of the activities between the 2nd and the 4th of April 2011. The simulation performed with the Multi-Satellite Simulator started at the 00:11:10 of the 2011/04/02 however the comparison between the data becomes meaningful after some hours, when the navigation filter has converged and the simulation has settled into the proper conditions. Therefore all plots here presented span from the 04:11:10 of the 2011/04/02 to the 16:11:10 of the 2011/04/04.

Table 1 shows the AFC2 flight plan over the time span under analysis: it contains the nominal relative orbital elements sent via telecommand to the AFC software. Each set of relative orbital elements identifies a specific formation geometry which needs to be acquired and maintained autonomously by the GNC system. With "geometry ID" it is meant the formation geometry identification defined in [19]. In particular, relative orbit "N" presents an ellipse-like motion in the radial/along-track plane of semi-axis 15 m and 30 m, centered at -100 m in the along-track direction. Geometry "O" is mainly an oscillation in the cross-track direction of 15 m amplitude, while being displaced 100 m behind Tango. Finally, geometry "P" is a hold-point at 100 m behind Tango in the along-track direction.

Table 1: *TCs sent to the AFC software during the simulation.*

Geometry ID (see [19])	Day [dd/mm/yy]	Time [hh:mm:ss]	δa [m]	$a\delta e_x$ [m]	$a\delta e_y$ [m]	$a\delta i_x$ [m]	$a\delta i_y$ [m]	$a\delta u$ [m]	AFC orbit control modes
Initialization	02/04/11	00:11:10	0	0	200	0	-200	4971.2	CL-T
N	02/04/11	04:11:10	0	0	15	0	-0.2	-100.02	CL-T
O	02/04/11	19:00:10	0	0	0.2	0	-15	-102.15	CL-R
P	04/04/11	06:11:10	0	0	0.2	0	-0.2	-100.02	CL-R

Fig. 6 illustrates the comparison between simulation and real telemetry, respectively marked in black and red, in terms of the relative motion of Mango w.r.t Tango. To better understand the high level of accordance between these results, one could investigate how the AFC software behaved in order to guarantee such a relative motion. To this aim the best features to be considered are the "maneuver counter" and the Delta-v consumption, respectively shown in Fig. 7 and 8.

The maneuver counter keeps track of the number of maneuvers commanded by the AFC software between each re-initialization. Operationally, a re-initialization is telecommanded together with each set of nominal relative orbital elements sent; in order to make the onboard software aware of the change occurred. This phenomenon can be clearly noted in Fig. 7, where both counters start from 0 every time that geometries N-O- and P are telecommanded.
Due to the design of the AFC controller, its action in the orbital plane is decoupled from the one out-of-plane. Moreover in-plane maneuvers always occur in couples, separated by half orbit, and are counted in pairs. The typology of in-plane maneuvers vary depending on the AFC orbit control mode, indicated in the right-most column of Table1. When operating in closed-loop along-track (CL-T), Mango performs only thrusts in the along-track direction. When in closed-loop radial (CL-R), instead, in-plane maneuvers involve Delta-v both in radial and along-track directions. As a consequence the control action in CL-R is almost the double expensive. But, at the same time, given the constraint on minimum thrust level, CL-R control is more accurate. The AFC state machine starts computing the location of a maneuver, or of the first of the couple, as soon as the correspondent control box is violated. The size of this control box is related to the aimed orbit control accuracy and is set via telecommand.

Therefore, taking into account how the AFC controller works, it is now evident why a good accordance of the behavior of the maneuver counter between simulation and flight data reflects an accurate reliability of the simulation itself. In Fig. 7, one could note that the simulation and Mango accomplished almost the same total number of maneuvers both in-plane and out-of-plane. Moreover, the distribution of the locations of those maneuvers behaves very similarly. That means that control box violations, hence, orbit control accuracy, are definitely of the same order of magnitude.

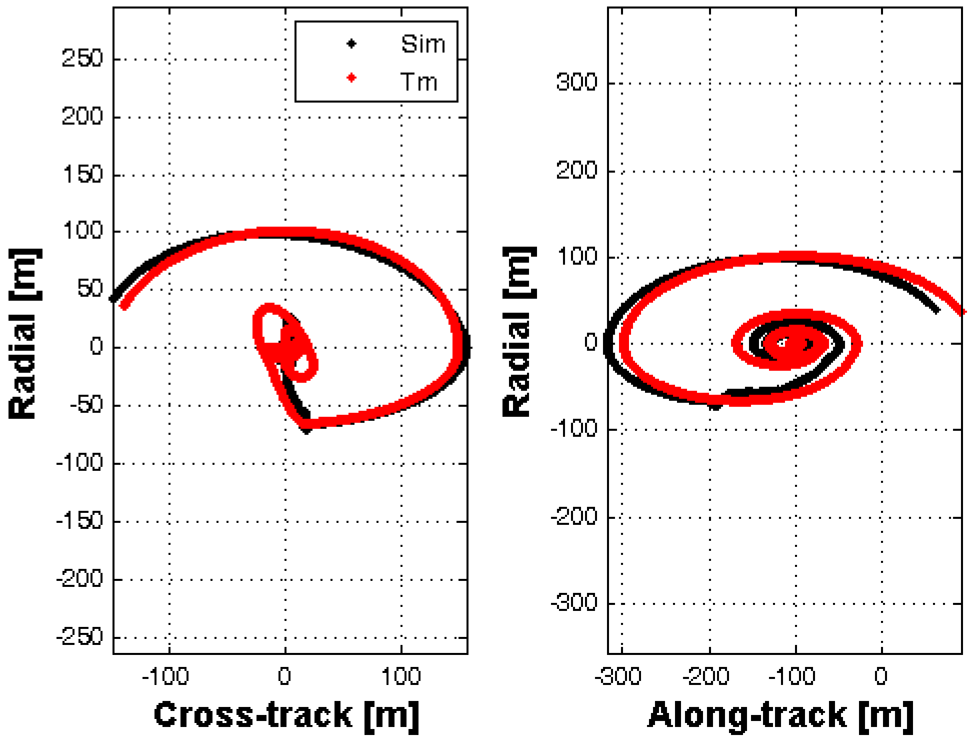

Fig. 6: *Comparison of the relative motion of Mango w.r.t. Tango in the RTN frame. Black markers identify simulation data, whereas red ones refer to the flight telemetry. Such a plotting convention is kept also in Fig. 7 and 8.*

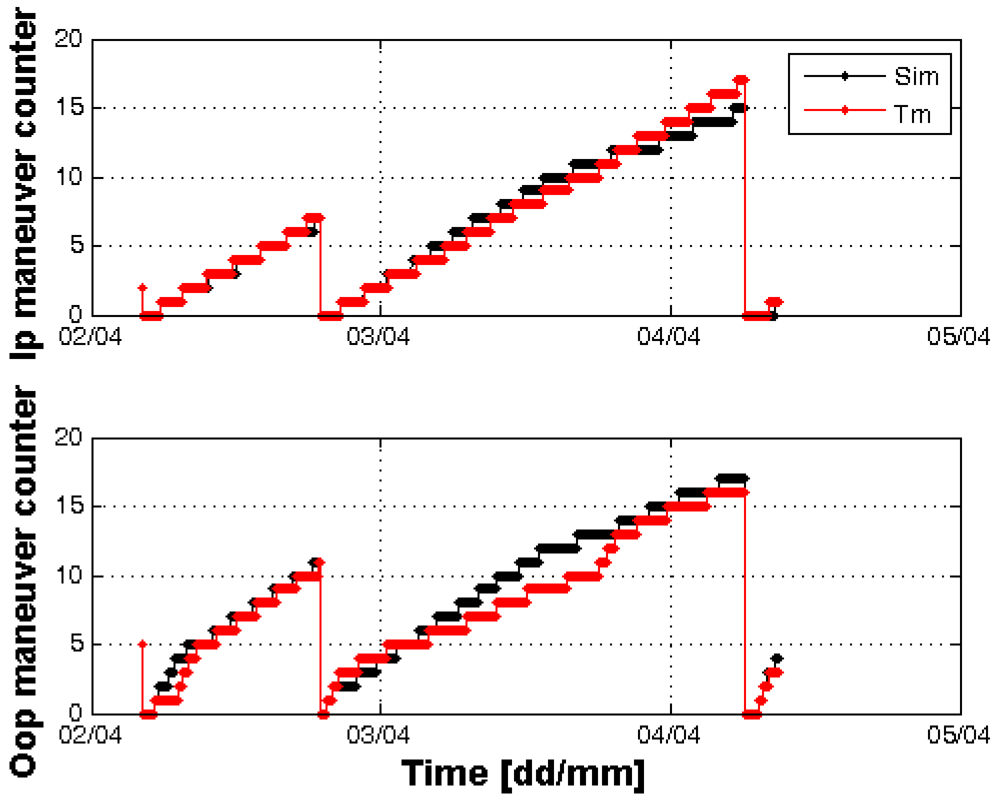

Fig. 7: *In-plane and out-of-plane maneuver counters commanded by the AFC software.*

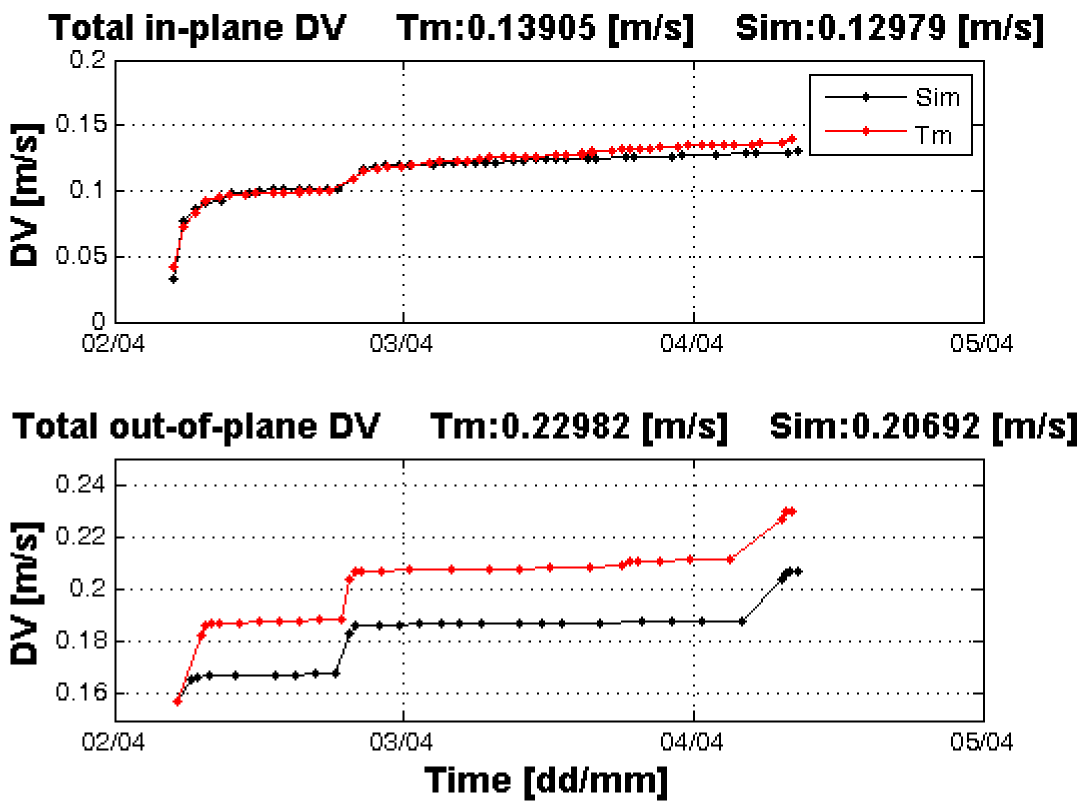

Fig. 8: *Comparison between simulated and real Delta-v budget.*

Finally, the good accordance in the maneuver counter trends reflects on the estimation of the Delta-V consumption. Referring to Fig. 8, it can be noted that the error scored by the prediction is less than the 10% of the total consumption for both the in-plane and out-of-plane corrections.

C. PLANNING THE DIGITAL VIDEO CAMERA'S ACTIVITIES

As mentioned in section 4.A, when planning the AFC2 experiment, the Multi-Satellite Simulator was set up including the model of the Digital Video System Camera (DVS) mounted on the Mango satellite.
Such a model was developed to simulate the images that can be shot in order to decide when to activate the camera during the operations. This information is crucial to collect proper images in view of future visual based navigation analysis or for testing image-processing algorithms with images taken in-space. Moreover this tool can improve the insight of the real images, especially when the interpretation is difficult due to the small size of the subject.

DLR used the DVS camera for the first time during AFC2. The first images collected were used to tune and validate the camera model. To this aim, the model was fed by orbit and attitude information coming from the precise orbit determination POD products [16]. The simulated-image produced at a given time was compared to the real image shot at the same time.
During the time of the simulation discussed in the previous section, three sequences of DVS images were planned. Fig. 9 and 10 refer to the second of those sequences. It started at the 19:01:15.0 UTC of the 2011-04-02. Pictures were shot every 75 s for a total of 6000 s (80 pictures). In that period relative geometry "O" was just telecommanded and Mango was starting the reconfiguration.

In Fig.9 ten pictures of that sequence are superposed. As a result one can have the idea of the motion Mango described and of the relative distance it had, referring to the changing size of Tango. Fig.10, instead, shows the simulated-images obtained during the simulation run with the MSS. Comparing these two figures gives a qualitative estimation of the reliability of the prediction. Discrepancies are mainly consequence of the relative position error that the simulation has with respect to the reality (Fig.6), mean error both in x and y coordinates is below 2 m.

Referring to Fig.10, the red and green segments in the center of the plot respectively indicate the +x and +y direction of the camera-fixed frame. DVS camera is mounted with axis parallel to the Mango body-frame and, during that period, Mango +x axis was pointing to Nadir. The layout of the Tango spacecraft is simplified as a box whose darkest face stands for the solar panel. In Fig.10 it is not visible due to the current attitude and relative position. Tango body-frame is marked with dashed red-green-blue segments, respectively referring to plus x-y-z axis. This helps in completing the rough representation of the spacecraft configuration: the +y axis lies between the two tilted formation flying radio frequency antennas [3], the +z axis coincides with the normal to the solar panel. The last two segments dashed yellow and black mark the Sun and Nadir directions in Tango body-frame.

5. CONCLUSIONS AND WAY FORWARD

This paper described the current status of the Multi-Satellite Simulator employed at DLR/GSOC to support various projects in the fields of formation flying and proximity operations.
The underlying development philosophy and different possible configurations of the MSS were discussed. For each of them a project was brought as example to easy the understanding of its functioning.
Among the possible applications, it was presented the utilization of the MSS during the operational phase of a mission. A comparison between simulated and real flight data was shown in order to assess the level of reliability of the simulations produced by the MSS.

Future activities will exploit the huge amount and variety of data gained during the ongoing PRISMA and TanDEM-X formation flying missions. The cross validation activities will continue, to further refine and validate the available functionalities. Furthermore the MMS will be used to improve the behavior of the flight software and, upon necessity, release and upload new patches to the spacecraft.

On a long-term perspective the Multi-Satellite Simulator will enable the full exploitation of the renewed European Proximity Operations facility of DLR. Efforts will be put in the implementation of processor-, software-, and hardware-in-the-loop capabilities to support the development, validation and operations of camera-based navigation and control systems for future on-orbit servicing missions.

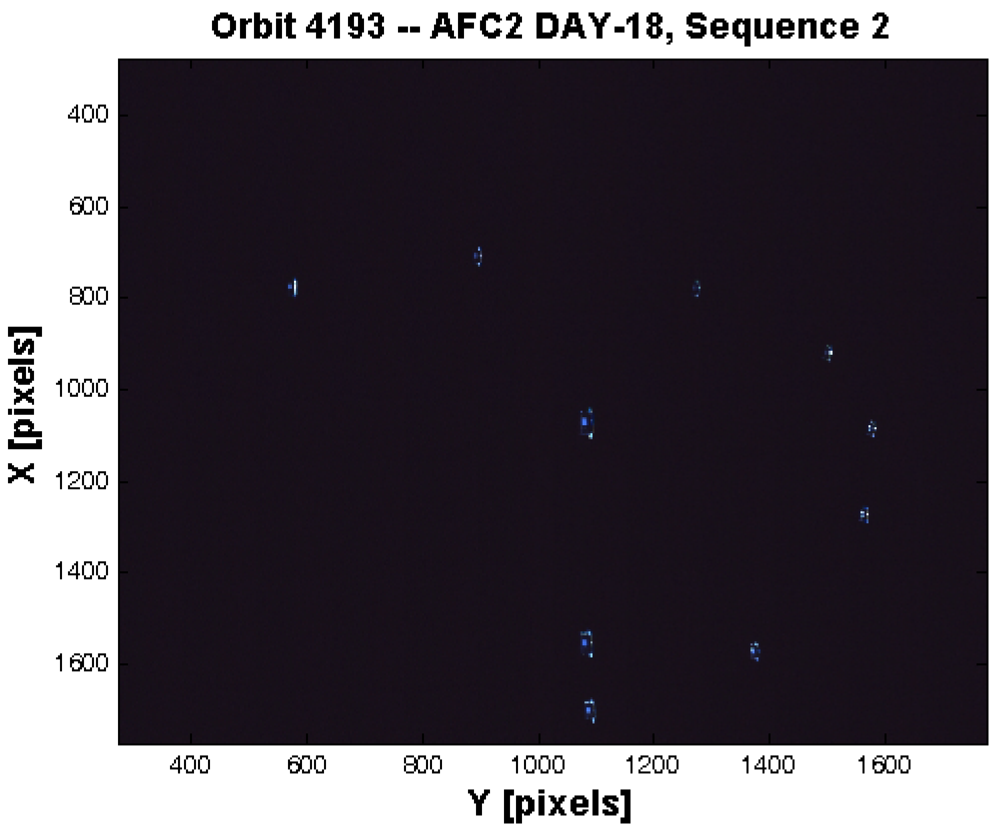

Fig. 9: *Some views of Tango collected in the DVS sequence performed in orbit 4193 (Courtesy of Swedish Space Corporation SSC)*

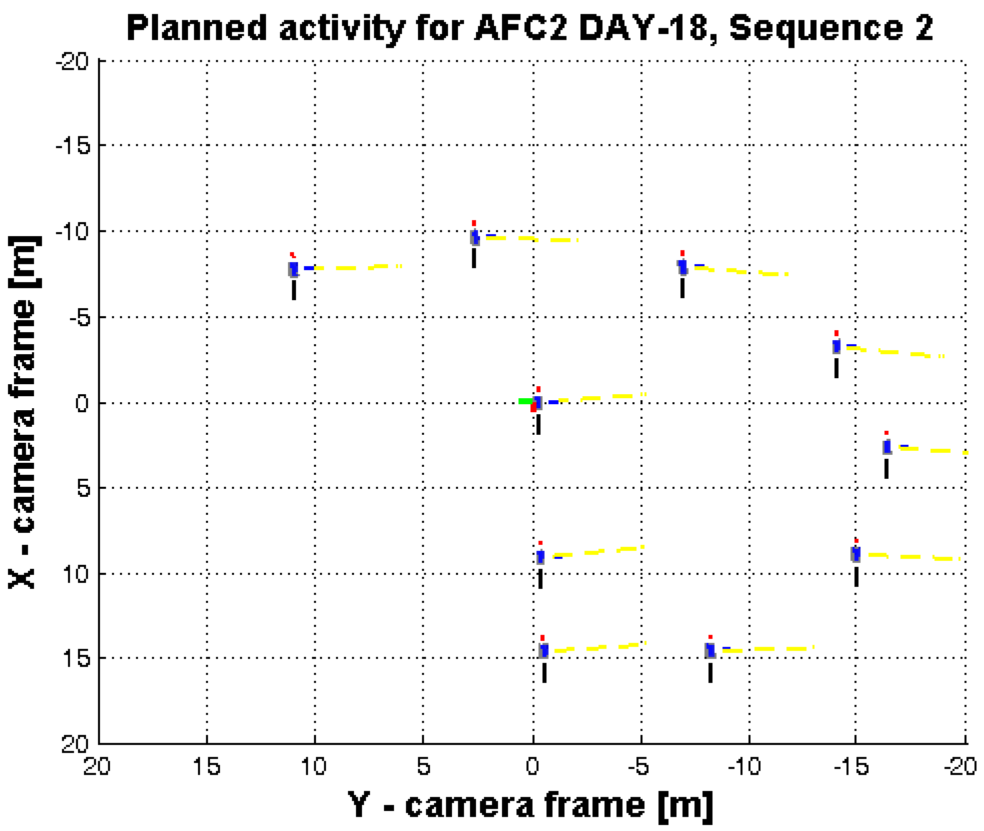

Fig. 10: *Simulated-images produced at the times in which pictures of Fig.9 were shot.*

REFERENCES

[1] J.-S. Ardaens and S. D'Amico, *Formation Flying Testbed; DLR-GSOC TN 09-01*, Deutsches Zentrum für Luft- und Raumfahrt, Oberpfaffenhofen, 2009.

[2] Simulink® 7 Reference, The Mathworks, Inc. (2010).

[3] S. Persson, B. Jakobsson and E. Gill, "PRISMA, Demonstration Mission for Advanced Rendezvous and, Formation Flying Technologies and Sensors," Number 05-B56B07, *56th International Astronautical Congress*, Fukuoka, Japan, International Astronautical Congress (2005).

[4] A. Moreira et. al., "TanDEM-X: A TerraSAR-X Add-on Satellite for Single-Pass SAR Interferometry," *IGARSS*, Achorage, USA; 2004.

[5] T. Van Helleputte, "User manual for the GHOST orbit determination software," FDS-SUM-3110; Deutsches Zentrum für Luft-und Raumfahrt, Oberpfaffenhofen (2004).

[6] G. Gaias, S. D'Amico, J.-S. Ardaens and T. Boge, "Hardware in-the-Loop Multi-Satellite Simulator For Proximity Operations," *11th Int. WS on Simulation & EGSE facilities for Space Programs*, SESP 2010, 28-30 September, ESTEC, Noordwijk, the Netherlands (2010).

[7] Matlab® 7 Getting Started Guide, The Mathworks, Inc. (2010).

[8] P. Bodin P et al., "Development, Test and Flight of the SMART-1 Attitude and Orbit Control System," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Fransisco, California, 15-18 August, (2005).

[9] T. Olsson and A. Edfors, "Model-Based Onboard Software Design: The PRISMA Case Study," *DAta Systems In Aerospace*, Berlin, Germany, 22-25 May, (2006).

[10] Real-Time Workshop® 7 Reference , The Mathworks, Inc. (2010).

[11] VxWorks Reference Manual, Edition 1, 5.3.1, WindRiver Systems (1998).

[12] RTEMS C User's Guide, Edition 4.6.5, for RTEMS 4.6.5, On-Line Applications Research Corporation (2003).

[13] O. Montenbruck, B. Nortier and S. Mostert, "A Miniature GPS Receiver for Precise Orbit Determination of the SUNSAT2004 Micro-Satellite," *ION National Technical Meeting*, 26-28 Jan. 2004, San Diego, California (2004).

[14] J. Gaisler, The LEON Processor User's Manual, Version 2.3.7, (2001).

[15] EPOS – Facility Manual, Robo-Technology GmbH, Puchheim, Germany (2009).

[16] J.-S. Ardaens, S. D'Amico and O. Montenbruck, "Final Commissioning of the PRISMA GPS Navigation System," *22nd International Symposium on Spaceflight Dynamics*, 28 Feb. - 4 March 2011, Sao Jose dos Campos, Brazil (2011).

[17] S. D'Amico, S. De Florio, J.-S. Ardaens and T. Yamamoto, "Offline and Hardware-in-the-loop Validation of the GPS-based Real-Time Navigation System for the PRISMA Formation Flying Mission," *3rd International Symposium on Formation Flying, Missions and Technology*, 23-25 April 2008, ESA/ESTEC, Noordwijk (2008).

[18] S. D'Amico, J.-S. Ardaens, S. De Florio, O. Montenbruck, S. Persson and R. Noteborn, "GPS-Based Spaceborne Autonomous Formation Flying Experiment (SAFE) on PRISMA: Initial Commissioning," *AIAA/AAS Astrodynamics Specialist Conference*, 2-5 August 2010, Toronto, Canada (2010).

[19] S. D'Amico, J.-S. Ardaens and R. Larsson, "In-Flight Demonstration of Formation Control based on Relative Orbital Elements," *4th International Conference on Spacecraft Formation Flying Missions & Technologies*; 18-20 May 2011, St-Hubert, Quebec (2011).