SpaceOps-2023, ID # 237

# A mission-agnostic onboard autonomy software solution to streamline spacecraft operations

**Paolo G.Madonia[a]*, Riccardo Maderna[b], Christian Cardenio[c], Alessandro Benetton[d], Temenuzhka V. Avramova[e], Gabriele Giordana[e], Johan Westin[f], Annalisa Toselli[f], Tobia Giani[g], Federico Fontana[h], Francesco Tosetti[i], Francesco Caronte[f], Luca Manca[l], Nadir Casciola[f]**

[a] Product Manager, *AIKO s.r.l., Via dei Mille 22, Torino, Italy, 10123*, papers@aikospace.com,
[b] Head of Autonomous Systems, *AIKO s.r.l*
[c] Head of Software Development, *AIKO s.r.l*
[d] CTO, *AIKO s.r.l.*
[e] Mission Autonomy Engineer, *AIKO s.r.l*
[f] Software Engineer, *AIKO s.r.l*
[g] Head of Simulation Department, *AIKO s.r.l*
[h] Head of Embedded Systems, *AIKO s.r.l*
[i] Embedded Engineer, *AIKO s.r.l*
l Deep Learning Engineer, *AIKO s.r.l*
*Corresponding Author

**Abstract**

Spacecraft autonomy is a promising way to address the increasing need for real-time responsivity by in-orbit platforms and operations scalability. Through onboard decision-making, we foresee a novel framework for space operations that will boost mission performance, will reduce the cost of satellite mission operations and will open new opportunities in the use of space.

In this paper, we present an example of how this approach to space operations can be achieved through the implementation of artificial intelligence-based software. We discuss the case of orbital_OLIVER, a software solution that streamlines spacecraft operations. The implemented technology is based on a modular architecture and a three-stage autonomous pipeline. The proposed technology is flexible enough to be abstracted from a specific mission scenario, making it adaptable to a variety of use cases, including remote sensing, telecommunications, and scientific applications. Finally, an overview of the current verification and validation status is provided.

**Keywords:** AI, automation, operations, autonomy

**Acronyms/Abbreviations**

- AI – Artificial Intelligence
- CCSDS – Consultative Committee for Space Data Systems
- Command & Data Handling
- COTS – Commercial-off-the-shelf
- CPU – Central Processing Unit
- DL – Deep learning
- ECSS – European Cooperation for Space Standardization
- EO – Earth Observation
- EPS – Electrical Power System
- ESA – European Space Agency
- FDIR – Fault Detection, Isolation and Recovery
- GNSS – Global Navigation Satellite System
- HW - Hardware
- ML – Machine Learning
- MO – Mission Operations
- P/L – Payload
- PUS – Packet Utilisation Standard
- RO – Radio Occultation
- TM - Telemetry

## 1. Introduction

As the space economy becomes increasingly competitive, the number of satellites and constellations in orbit is expected to grow significantly in the coming years. In order to maintain a competitive edge in scientific, technological, and commercial endeavours, a higher level of autonomy in space missions will be crucial. As per NASA's definition [1], autonomy is "the ability of a system to achieve goals while operating independently of external control". Autonomous robotics is expected to be one of the key pillars in the future evolution of the space sector. Artificial intelligence (AI) applications are already prevalent in a wide range of industries and research fields, and the space industry is poised to be the next frontier for AI.

During the last few years, various autonomous features and automated system-level abilities have been shown and employed in spacecraft operations. This is especially the case of threshold analysis of telemetry data for Fault Detection, Isolation and Recovery (FDIR) purposes, which has been now in adoption for many decades. However, spacecraft are still mostly dependent on ground-based systems to evaluate circumstances and make decisions on the next steps, utilizing pre-written command sequences. Indeed, while we experienced exceptional advancements in hardware and spacecraft miniaturization (which boosted the recent growth of the space economy), little has changed concerning flight software and the way operations are carried out.

The commercial space race and the increased public interest in space-related activities are now driving the development of autonomous platforms that can overcome operational constraints and break the last barriers that prevent the adoption of autonomous technologies. Indeed, the space environment presents several challenges for robotic systems, from low Earth orbit up to deep space, and automated or autonomous approaches have often been hard to be implemented with sufficient safety or with low enough costs.

To date, we can identify three major aspects that will benefit from the adoption of increasingly autonomous space systems: responsivity, platform complexity, and operations scalability.

*Responsivity* - Historically, satellites, spacecraft, constellations, rovers, and landers have all faced uncertain mission environments and unexpected events, and their ability to efficiently react, adjust, and explore is still limited. In low Earth orbit, latency, scattered communication, and limited communication windows represent a significant bottleneck for the success of both scientific and commercial spacecraft missions, since any decision or action has to pass through the control of the ground segment, which sometimes may be out of reach for several hours.

*Platform complexity and mission efficiency*- As technology continues to advance and commercial interest in the space economy grows, satellite platforms will become increasingly complex, with more capabilities and advanced subsystems. As platforms become increasingly powerful and capable, the way we use them must follow suit, to ensure that we are exploiting the full potential of the hardware and software that composes the platform.

*Operations scalability* – Thanks to the miniaturisation of satellite platforms, in the last 10 years we have experienced a surge in the number of satellites operating in Earth orbit. At a current value of above 4000 units, this number is set to grow exponentially over the upcoming decade, mostly due to the launch of several constellations of satellites. The current approach to operations -heavily reliant on ground control and human decisions- is not equipped to scale along with this increase in the number of operating assets.

These problems will affect both commercial spacecraft operations as well as institutional missions [2], and the search for their solutions has been marked as a priority. Over the next decade, spacecraft autonomy is expected to be a major enabler for missions of different natures, including Earth Observation (EO), telecommunications, defence, space exploration, and even crewed missions [3]. In light of this, the integration of AI-based architectures is crucial to increase the operational capabilities of robotics systems in the space domain while reducing the workload on the ground.

The remainder of the paper is organized as follows. Section 2 provides a broad overview of a proposed software solution to enable onboard autonomy. Section 3 describes the modular structure of the software detailing the functions of each module. Section 4 presents the verification and validation activities, with an example of the software applied to an EO scenario. Finally, Section 6 reports the conclusions and describes the next steps that are expected in the software development and validation.

## 2. orbital_OLIVER: overview

This paper presents orbital_OLIVER, a specific software implementation of an onboard autonomous agent that relies on AI as its main building block. orbital_OLIVER is a software solution developed by AIKO to streamline spacecraft operations and augment mission performance by making satellites less dependent on ground control. This tool has been specifically intended for use by spacecraft owners and operators that need to improve mission

performance and increase scalability by reducing the human workload on the ground. This software can support many types of space missions, from small satellites to flagships, in low Earth orbit and deep space.

### *2.1 The Pipeline*

This technology is based on a modular architecture and a three-stage autonomous pipeline (Figure 1): i) platform and environment sensing; ii) reasoning and goal definition; iii) planning and scheduling. This breakdown is an adaptation to the most generalized abstraction of an autonomous system [4].

In orbital_OLIVER, the abstraction scheme is mapped onto a modular structure composed of three independent (but interlinked) components: the *sensing* module, the *reasoning module (reasoner),* and the *planning module (planner)*. With this approach, the satellite is capable of: perceiving its surroundings and its condition through onboard data processing; using the acquired data to reason on the best goals to pursue, or to trigger an emergency procedure in the case of an anomaly; re-planning the mission schedule, if the goals generated at the second step differ from the directions provided by the ground segment or if an anomaly has been detected.
A more detailed description of the three modules is provided in section 3.


**Figure 1 -** The three-step pipeline of the autonomous agent.

### *2.2 Autonomous spacecraft operations*

In the canonical approach to operations (Figure 2) the satellite periodically receives a schedule generated by the ground segment. That schedule will be valid until contact with ground is established again and the task list is updated. While this approach has been extremely solid since the start of space exploration, the advancements in data processing -specifically, onboard data processing- may make this an obsolete flow.

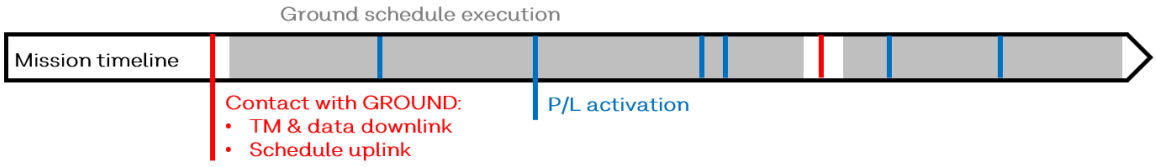


**Figure 2 -** Conventional operations approach.

In a mission equipped with onboard data processing capabilities and onboard autonomy tools, like orbital_OLIVER, the flow of the operations would look like the one shown Figure 3. It is worth highlighting a few differences and analogies with the previous scenario.
*First*, the combined action of the sensing and reasoning modules allows the software to ingest the platform telemetry and monitor the spacecraft's health in near-real time, ensuring faster response to anomalies; thanks to this, it will not be needed to downlink the full telemetry and analyse them on Earth, but the operators will be able to select and downlink only the most relevant data packets to monitor on ground.
*Second*, a schedule from ground can still be uploaded as needed, ensuring complete control over the platform.

*Third*, with onboard autonomy, if a specific event is detected by the sensing-reasoning components, the third item in the chain (the planner) is triggered; in this case, the planner adapts the mission schedule almost in real-time, to take advantage of opportunities that were unforeseen at the time of the last schedule instantiation.
*Fourth,* upon contact with the ground station, the acquired data is downlinked, as well as the schedule generated onboard; the mission can then proceed either referring to the ground-generated or the onboard-generated schedule,

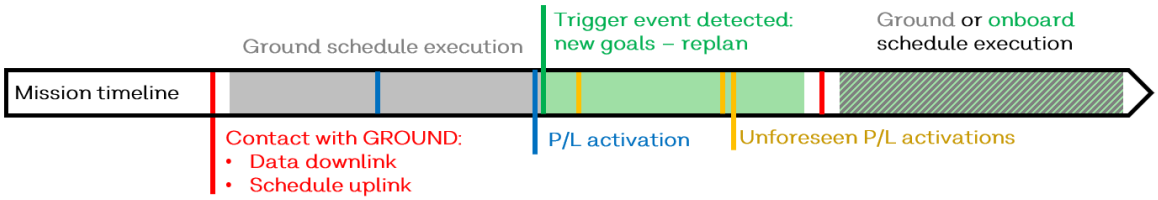


**Figure 3 -** Operations enhanced by onboard autonomy.

*2.3 Software design*

To allow for extra flexibility for the users, this software solution has been designed to be compatible with commercial-off-the-shelf (COTS) platforms. Since orbital_OLIVER is not tied to a specific hardware platform on which it has to be installed before the flight, this design choice also allows for the software to be deployed on the fly, uploading it onto assets that are already in orbit. To maximise software delivery speed, extensibility, testability and maintainability, this software tool is based on a distributed microservice architecture. The software design is compliant with the following space standards: CCSDS 520.0-G-3 (Mission Operations Services Concept); CCSDS 521.0-B-2 (Mission Operations Message Abstraction Layer); ECSS-E-ST-40C (Software Engineering). Data exchanges occur through PUS/MO services.

*2.4 Features of this implementation*

The software implementation described in this work provides several features and added capabilities to a space mission. In the following subsections, we briefly describe the five major advantages provided by the software.

*2.4.1 Telemetry data analysis*

The software is capable of detecting and preventing potential contingencies before they affect satellite operability. To do that, it exploits state-of-the-art of Deep Learning (DL) technologies for time-series analysis to monitor telemetry and housekeeping data streams from the platform. The double diagnostic and prognostic actions are further discussed in section 3.1.1.

*2.4.2 Timely data analysis*

The software can extract actionable information in real or almost-real time, both from telemetry data and from payload data. This analysis is performed using Machine Learning (ML) and DL techniques. This feature solves the long-standing issue of ground contact latency in mission operations, providing the needed information to take decisions even without the human-in-the-loop.

*2.4.3 Payload product and bandwidth optimization*

By continuously evaluating whether the current schedule is the most appropriate for the current mission objectives, the software ensures the pursuit of an optimal mission plan, improving data acquisition. On the other hand, whenever the software detects that low-quality data has been acquired, its downlink can be discarded or deprioritized. Ultimately, since data is preliminarily processed onboard, if needed due to system constraints, the downlink of information can be limited to just actionable metadata. All these actions reduce the amount of data that is exchanged with ground control on Earth, allowing maximum payload exploitation in limited bandwidth constraints scenarios.

*2.4.4 Tailorability to scenarios*

The core of the software is an algorithmic structure which ensures the execution of the Pipeline presented in the previous section. Alongside this, a mission-specific knowledge base completes the software package. This latter component ensures ease of customization to different mission scenarios through a set of configuration files. The tailoring of the software can be achieved by adjusting the knowledge base, without having to act on the AI

algorithms. This knowledge base contains information about platform consumable resources, platform subsystems, mission objectives, ground segment, and orbital configuration.

## 3. Modular architecture

As anticipated in the previous section, the software has a tripartite modular structure. This section describes the functioning of each module.

### *3.1 Sensing module*

The role of this module is to extract relevant information from the spacecraft platform. This includes platform telemetry, payload telemetry, and payload data.

#### *3.1.1 Telemetry data processing*

The sensing module features a Machine Learning (ML)-based inference engine dedicated to telemetry data processing. Thanks to this element, the sensing module enhances the classic FDIR approach (based on threshold analysis) and provides a tool for monitoring the platform's health more efficiently. Indeed, by using ML, the sensing module enables both health *diagnostics* functionalities (i.e. to investigate failures and understand the root cause), and *prognostics,* allowing the prediction of potential failures before they occur, thus extending the lifetime of the platform. Before the mission, the engine is trained on historical data (if available) and/or synthetic data. During the mission, the ML model can be retrained as needed using real data from the platform, and then it can be uploaded again to update the inference engine onboard and deliver more accurate telemetry analysis.

#### *3.1.2 Payload data processing*

Depending on the source, payload data can be fed directly into the sensing module as raw or can be preprocessed by an external inference engine. This latter case is especially relevant for EO, where external onboard processing can provide insights that can then trigger decisions by the reasoning module. Examples of such kinds of inference can be image segmentation for cloud coverage estimation, object detection, and SAR data focusing onboard. The sensing module acts as an input interface, making sure the other services of the autonomous agent are provided with the maximum amount of actionable information.

### *3.2 Reasoning module*

The role of this module is to explain (when needed) the data collected by the sensing module and devise the optimal goals to pursue, based on those data and higher-level mission goals.

The core of this module is based on *Symbolic AI* technology. This AI-based software uses scenario and system-level information stored in a knowledge base to solve problems that would usually require a human expert, thus preserving its knowledge in a database. An inference engine is applied to the knowledge base to derive information starting from already-known facts. Lower-level information is queried during the inference process until a known fact is encountered, thus reconstructing the actual system and mission knowledge state.

In light of this, the reasoning module takes as input the information extracted from the telemetry and/or payload data, placing it in the mission context by using the specific knowledge defined in the design phase, to generate an optimal high-level goal that must be pursued by the spacecraft. In addition to this, the module can also be provided with goals defined by the operators.

### *3.3 Planning module*

The planning module is tasked with the last activity in the pipeline: generating an actual mission plan based on the goals defined by the reasoning module. Once a goal has been identified, it is divided into smaller tasks. This breakdown is defined as well within the mission knowledge. Then, the planning module processes this list, associating each of these tasks with a specific execution time, based on: task duration, task priority, onboard resources utilization, and task precedences. To produce the schedule, the module adopts an algorithm based on linear integer programming, which optimizes the allocation of tasks within the satellite's time horizon.

## 4. Heritage, verification and validation

The verification and validation of the software solution presented in this paper started soon after the beginning of this project. The development of orbital_OLIVER started in 2017 under the name MiRAGE. In 2018 the project received support for development and verification through the Horizon 2020 program of the European Commission [8].

*4.1 Hardware integration*

The software has been successfully deployed and tested on COTS processors with ARM 32bit/64bit (Cortex-A series) and X86_64 architectures. The AI-based inference has been tested on Hardware accelerators with both specific and multi-platform runtimes. Table 1 summarizes the current compatibility status of the software.

**Table 1 -** Hardware compatibility

| **CPU compatibility** | | **AI inference compatibility** | |
|---|---|---|---|
| 32-bit ARMv7 | *Torpedo SOM* | HW accelerators with specific runtimes | *Intel Myriad VPU (OpenVINO)* |
| | *Raspberry Pi3* | | *Google Coral Edge TPU (TFLite with Edge TPU runtime)* |
| 64-bit ARMv8 | *NVIDIA Jetson Nano* | | *Xilinx FPGAs (Vitis AI)* |
| | *Xilinx Zynq Ultrascale* | | *NVIDIA GPUs (TensorRT)* |
| | *Raspberry Pi4* | CPUs with multi-platform runtimes | *TensorFlow Lite* |
| 64-bit X86 | *AMD Ryzen Embedded* | | *ONNX Runtime* |
| | | | *ArmNN* |

*4.2 In-orbit validation plan*

Since 2021, the in-orbit verification and validation have been supported by the European Space Agency's (ESA) InCubed program [9]. Thanks to this initiative, in 2022 the product has been released to a pool of four companies as part of an *Early Adopters Program.* In the context of this program, orbital_OLIVER will fly on four different missions between 2022 and early 2024.

*4.3 First in-orbit deployment*

The first verification phase has been ongoing since August 2022, when orbital_OLIVER has been deployed on a 6U CubeSat platform. During this phase, the software has been running on synthetic data that had been uploaded onboard. The data reproduce an operations scenario of a GNSS-Radio Occultation (RO) mission, in which the software manages limited power and storage resources to maximise the amount of data acquisitions. This choice was made to test just that the software runs on the platform, without introducing any risks by allowing the software to act directly on the spacecraft. For this reason, this test is only regarded as a verification, and validation in an operative scenario will follow the successful completion of these trials. This phase is currently ongoing, and tests are expected to be completed by Q2-2023.

*4.4 Validation in an Earth Observation scenario*

To prepare for the in-orbit validation, the Simulation team in AIKO has developed a EO use case to demonstrate and measure the advantages of onboard autonomy in a specific scenario, featuring a 3U CubeSat. The following configuration (reported in Table 2) has been adopted. On the satellite, two software tools are installed: orbital_OLIVER, to enable onboard autonomy, and a ML-based data processing tool to identify clouds in optical images. In this simplified use case, orbital_OLIVER monitors the telemetry from the Electrical Power System (EPS) and Command & Data Handling (CDH) subsystems.

The simulated satellite is equipped with two remote sensing, multispectral payloads: a low-resolution *monitoring camera* with fixed acquisition rate and a high-resolution *primary camera*, with variable acquisition rates. Three payload operative modes are allowed: low-rate monitoring, low-rate acquisition, high-rate acquisition. The switch between these phases is triggered by orbital_OLIVER depending on the cloud coverage. Acquisitions are intended to be continuous. Two other operative modes are defined for *downlink* and *recharge* phases.

**Table 2 - Orbit and ground segment configuration**

| | |
|---|---|
| **Orbit type** | Sun-Synchronous Orbit (SSO) |

| | |
|---|---|
| **Orbit altitude** | 543 km |
| **LTAN** | 12AM |
| **Eccentricity** | 0.0245 |
| **Inclination** | 97.7° |
| **Ground segment** | One station, 45°N latitude |

In this scenario, the satellite receives from ground control an acquisition schedule for the low-resolution payload. As the monitoring camera gathers data, the cloud detection tool processes them and extracts information about the cloud coverage. This information is passed on to orbital_OLIVER: if the cloud coverage is below a mission-defined threshold (50%, in this case), the high-resolution/low-frequency acquisition mode is triggered. Figure 4 presents an example of this transition, visible in the center-top panel displaying the mission timeline, as the "monitoring" timeline is interrupted and a "Acquisition LF" is started. The coloured timelines refer to activities scheduled or predicted by the onboard agent, while the grey ones are the activities included in the schedule uploaded from ground. The left-side panel reports information on the battery status, onboard available memory status, orientation. The right-side shows a preview of the image acquired by the payload in use.

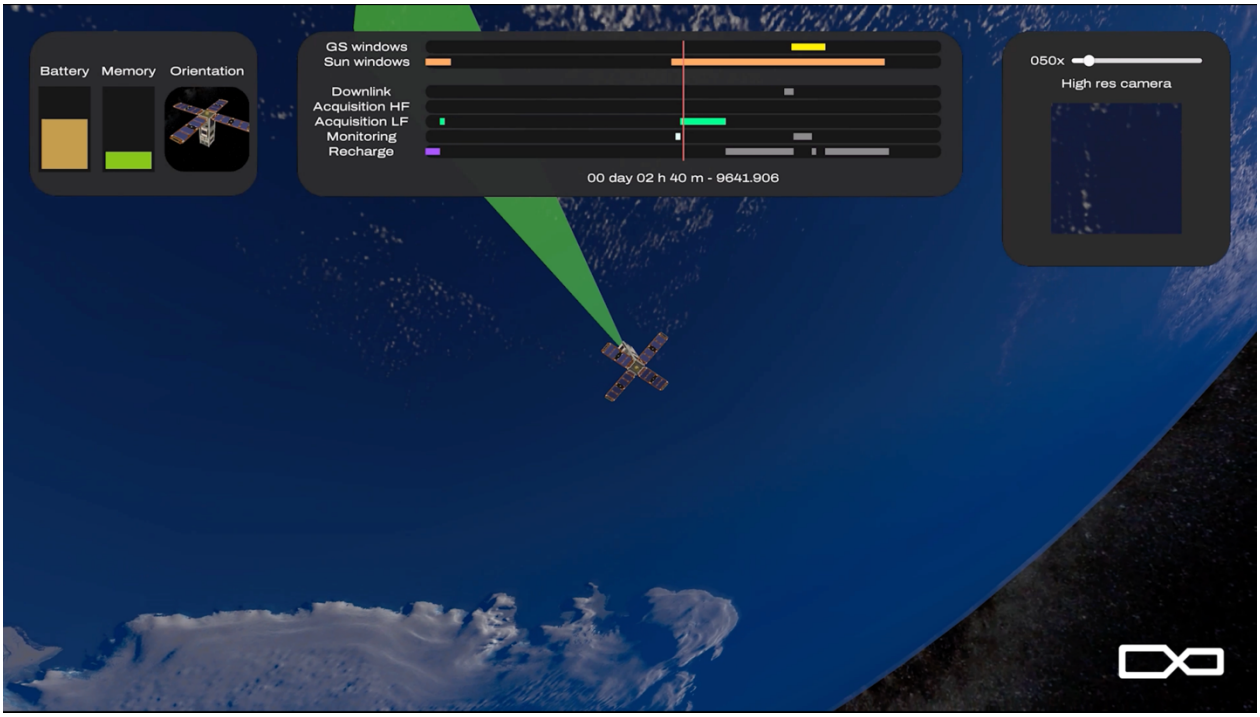


**Figure 4 -** Earth Observation scenario for validation, initial acquisition sequence and first transition.

If the cloud coverage conditions improve (coverage below 25%, in this case), orbital_OLIVER engages the third payload acquisition mode (high-resolution/high-frequency), as visible in Figure 5. The same steps (in the opposite way) are taken whenever the onboard processing tool detects a degradation in the data quality due to excessive presence of clouds. While the insights from payload data (i.e. the cloud coverage information) are crucial in this concept of operations, orbital_OLIVER bases its replanning decisions also on the battery and storage levels, in order to ensure that demanding acquisitions such as the high-resolution/high-frequency ones are not engaged whenever the power levels are too low or there is not enough space to store the payload data. In this case, the knowledge about Sun visibility and Ground Station visibility windows allows orbital_OLIVER to ensure that the satellite has enough power to go through an eclipse window (about 40 minutes per orbit) and to engage in downlink tasks whenever the Ground Station is visible.

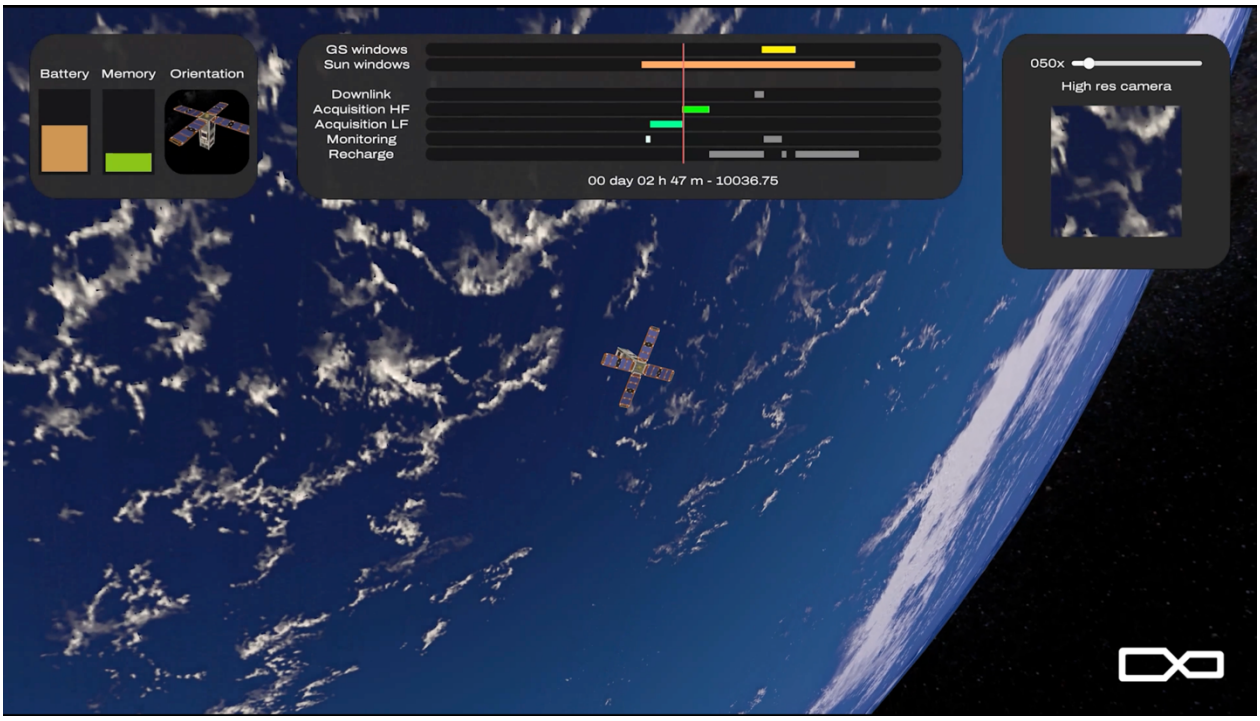


**Figure 5 -** Earth Observation scenario for validation, replanned acquisition schedule

This simulation has been then compared with a benchmark scenario, in which the classical approach to operations is used. In this scenario neither platform autonomy nor onboard data processing is enabled by software tools. In this case, the satellite is equipped just with the *primary* camera, and all the acquisitions that are scheduled from ground are executed in the high-resolution/high-frequency mode.

Figure 6 shows the results of a preliminary analysis carried over 6 hours of acquisitions. The data has been categorized according to the cloud coverage measured in each frame (good quality – below 25%, mid quality – between 25% and 50%, poor quality – above 50%). While the absolute number of frames labeled as good is lower in the autonomy-enhanced scenario, the relative fraction of good data over the total number of acquired frames increases from 60% to about 80%. On the other hand, the use of onboard autonomy allowed for more efficient usage also of onboard consumables, providing a 35% decrease of the time spent in recharging mode (1.9h vs 1.22h).

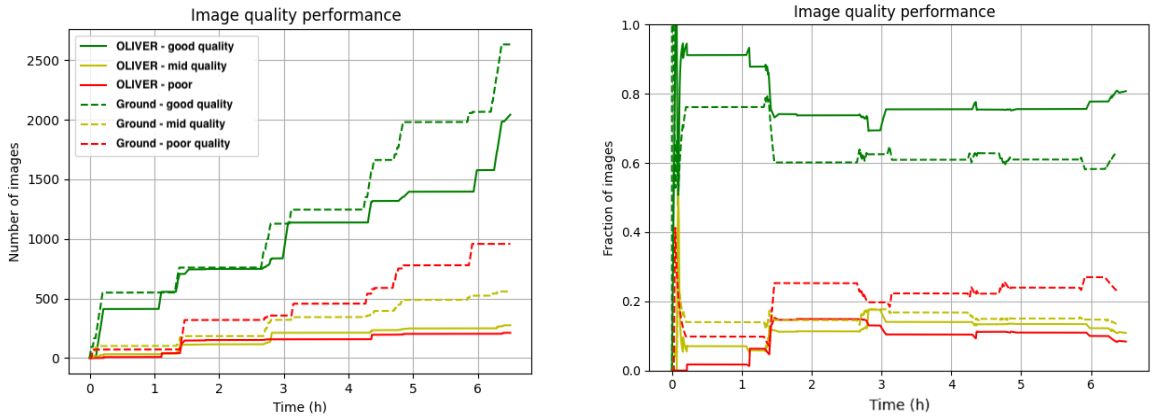


**Figure 6 -** Time series of image acquisitions divided according to image quality: absolute counts (left), relative fraction (right).

## 5. Conclusions and next steps

This work provided a brief overview of how spacecraft autonomy will improve spacecraft performance, allowing for faster response, better platform health monitoring, and exploiting mission opportunities that would go unnoticed in the ground-based mission operation framework. As a byproduct of the enhanced independence from the ground, spacecraft autonomy will reduce the workload on the mission control centres, providing a scalable solution that can be applied to handle the tens of thousands of satellites that will orbit Earth by the end of this decade. A specific software solution to implement onboard autonomy -orbital_OLIVER- has been presented. The AI-based, modular architecture of the software ensures robustness and flexibility against several mission scenarios and platform sizes.

At the time of writing, the major milestone ahead in orbital_OLIVER's roadmap is the completion of the in-orbit validation currently in progress through the InCubed+ programme. This is expected to occur between Q2/Q3-2023. The preliminary results from a simulated application of onboard autonomy to an EO scenario have been discussed. In the scenario enhanced by orbital_OLIVER, the fraction of high-quality data (i.e. cloud coverage below 25%) has shown an increase from about 60% to 80% over the total amount of frames acquired, with respect to the benchmark scenario based on ground-generated schedules.

Once the in-orbit validation of the software's features is achieved, orbital_OLIVER is planned to enter a commercialization phase, as the natural continuation of the *Early Adopters Program*. From the technical standpoint, a new version of the planning module will be implemented over the next months, providing the software with proactive and reactive planning capabilities [10]. Concerning Hardware compatibility, support will be added to additional CPU architectures (e.g. RISC-V) and HW accelerators.

Eventually, the capabilities of this tool will be upgraded to act on distributed systems. This will include implementing features regarding the autonomous management of constellations, enabling cooperative operations, goal negotiation, intent prediction, and the sharing of collective knowledge. These features are closely related to the architecture of the mission and shall be seamlessly integrated with single-agent autonomy.

## Acknowledgements

The authors acknowledge the support received by the EU Commission's Horizon 2020 programme (Grant ID: 816442) and by ESA's InCubed+ programme (orbital_OLIVER ex MiRAGE project).

## References

[1] NASA Autonomous Systems – Systems Capability Leadership Team, "Autonomous systems taxonomy" Report, (2018) https://ntrs. nasa.gov/citations/20180003082, pp.1, (accessed 25.01.2023).
[2] NASA Jet Propulsion Laboratory, Strategic Technologies Report (2019), pp.11-13.
[3] J. Frank, NASA Autonomous Mission Operations Roadmap, https://www.nasa.gov/sites/default/files/files/J_Frank-Autonomous_Mission_Operations.pdf (accessed 25.01.2023)
[4] Nesnas, I.A., Fesq, L.M. & Volpe, R.A. Autonomy for Space Robots: Past, Present, and Future. Curr Robot Rep 2, 251–263 (2021). https://doi.org/10.1007/s43154-021-00057-2 (accessed 25.01.2023)
[5] CCSDS 520.0-G-3 Informational Report, Mission Operation Service Concept, December 2010
[6] CCSDS 521.0-B-2 Recommended Standard, Mission Operations Message Abstraction Layer, March 2013
[7] ECSS-E-ST-40C Space Engineering, Software, 6 March 2009
[8] European Commission – Horizon 2020, CORDIS EU research results, "Enhanced mission autonomy through onboard Artificial Intelligence for next-generation spacecraft", Project "MiRAGE", Grant agreement ID 816442, https://cordis.europa.eu/project/id/816442 (accessed 25.01.2023)
[9] ESA InCubed Activity Portfolio, orbital_OLIVER, https://incubed.esa.int/portfolio/orbital_oliver (accessed 25.01.2023)
[10] R. Maderna, G. Giordana, Proactive-reactive on-board planning for complete goal-oriented automation, SpaceOps 2023 #246, 17th Conference on Space Operations, Dubai, United Arab Emirates, 2023, 6-10 March (this issue)