SSC21-I-07

An AI-based Goal-Oriented Agent for advanced on-board automation

Christian Cardenio, Lorenzo Feruglio, Alessandro Benetton, Mattia Varile, Temenuzhka Valentinova Avramova
AIKO S.r.l. Autonomous Space Missions
Corso Castelfidardo 30/A, 10129, Torino, Italy; +39 3757065595
papers@aikospace.com

ABSTRACT

In the context of fierce competition arising in the space economy, the number of satellites and constellations that will be placed in orbit is set to increase considerably in the upcoming years. In such a dynamic environment, raising the autonomy level of the next space missions is key to maintaining a competitive edge in terms of the scientific, technological, and commercial outcome.

We propose the adoption of an AI-based autonomous agent aiming to fully enable spacecraft's goal-oriented autonomy. The implemented cognitive architecture collects input starting from the sensing of the surrounding operating environment and defines a low-level schedule of tasks that will be carried out throughout the specified horizon. Furthermore, the agent provides a planner module designed to find optimal solutions that maximize the outcome of the pursued objective goal. The autonomous loop is closed by comparing the expected outcome of these scheduled tasks against the real environment measurements.

The entire algorithmic pipeline was tested in a simulated operational environment, specifically developed for replicating inputs and resources relative to Earth Observation missions. The autonomous reasoning agent was evaluated against the classical, non-autonomous, mission control approach, considering both the quantity and the quality of collected observation data in addition to the quantity of the observation opportunities exploited throughout the simulation time. The preliminary simulation results point out that the adoption of our software agent enhances dramatically the effectiveness of the entire mission, increasing and optimizing in-orbit activities, on the one hand, reducing events' response latency (opportunities, failures, malfunctioning, etc.) on the other.

In the presentation, we will cover the description of the high-level algorithmic structure of the proposed goal-oriented reasoning model, as well as a brief explanation of each internal module's contribution to the overall agent's architecture. Besides, an overview of the parameters processed as input and the expected algorithms' output will be provided, to contextualize the placement of the proposed solution. Finally, an Earth Observation use case will be used as the benchmark to test the performances of the proposed approach against the classical one, highlighting promising conclusions regarding our autonomous agent's adoption.

.

CONTEXT AND PROBLEM DEFINITION

In the fierce competition of the space economy, the number of satellites and constellations placed in orbit might increase considerably in the upcoming years. A higher autonomy of space missions is critical to maintaining a competitive edge of scientific, technological, and commercial outcomes in such a dynamic environment. The large-scale adoption of autonomous robotics is one of the fundamental pillars at the base of the evolution of humanity into an interplanetary species.

AI applications are already a reality in a wide range of industrial and research fields. Soon, space will become the next frontier for AI. Economic growth and social interest in space domain activities are drivers of the technological push whose autonomous platforms allow for overcoming operational constraints.

Space represents a challenging environment for robotic system operations occurring in low Earth orbit up to deep space. A wide variety of satellites, spacecraft, constellations, rovers, and landers have historically faced uncertain mission environments and unexpected events. Even now, their capabilities to efficiently react, adjust and freely explore are limited.

In low Earth orbit, latency, scattered, and limited communication windows represent a significant bottleneck, a concern for the overall outcome of both scientific and commercial spacecraft missions. Such a constraint may affect the ability of an orbiting platform to react to unexpected internal or external events efficiently. Also, the delay of real-time information from the satellite limits the exploitation of unforeseen operational opportunities.

This problem is a priority for the Jet Propulsion Laboratory, as mentioned in their most recent strategic intent document[1]: *"Some future missions will have limited communication with Earth for extended periods of time, such as drilling through kilometers of icy crust on Europa, requiring the systems to be able to assess their own environment and make decisions independently. Other missions will require reacting on a timeframe that is shorter than the communication time with Earth such as sampling from short-lived plumes. Missions that cannot receive commands from Earth quickly and reliably will need the autonomous capability to explore with reduced or no human intervention. Autonomy can increase spacecraft productivity and, when the spacecraft cannot wait for ground commands, enable rapid reactions."*

In the upcoming years, technological growth and an increased commercial interest in the space economy will lead to developing more complex satellite platforms with more payloads and advanced subsystems. Such development shall be supported by optimizing the real-time management of onboard resources. As a result, the workload of satellite operators will rapidly increase when accounting for the simultaneous expansion of satellite constellations in low Earth orbit. This scenario has consequences for both the mission planning, operability, and constellations themselves.

The expansion of space exploration leads engineers and platform operators to face higher complexity issues. Hence, large-scale integration of AI-based architectures is significant to increase the operational capabilities of robotics systems in the space domain while containing the workload on the ground.

THE SOLUTION

AIKO's solution to the problems is MiRAGE, an onboard automation software that enables advanced autonomous operations.

Considering a satellite platform, three major high-level features that can benefit from adopting AI-based solutions are abstracted: (i) onboard data processing, (ii) operations planning, and (iii) autonomous control. Increasing mission and system autonomy rely on integrating cutting-edge AI technologies into these three main functional modules.

A simple cognitive architecture is abstracted from the complexity of a space system. In this regard, our approach aims to provide the satellite with the ability: to sense the environment and its status (through onboard data processing), to plan tasks according to acquired, gathered, or inferred knowledge (operations planning), to self-maneuver in the orbital environment (autonomous control), and to operate in a coordinated system to accomplish more complex distributed tasks such as those requiring a combination of the individual abilities mentioned above.

Sensing capabilities are implemented through the integration of Deep Learning applications, related to payload data processing or telemetry data processing. This approach enhances satellite self-awareness and predicts or prevents possible detrimental behaviors during the mission. In general, perceiving the surrounding orbital environment and infer structured information provides the agent with full autonomy for discovering unexpected events, unforeseen mission opportunities, or cooperation opportunities supporting other active or passive agents.

Although sensing the environment is crucial for efficiency, it is not enough to enable satellite autonomy. The onboard software presented aims to maximize the whole mission outcomes by setting up various ability levels required by specific needs of goal reasoning and scheduling capabilities to optimize limited resources. In particular:

- optimally scheduled tasks run even in an uncertain environment by enhancing a high level of flexibility while guaranteeing adaptivity to faults and events.
- autonomous control capabilities could be enabled, mixing accurate sensing and specific adaptive scheduling features.
- proximity operations, docking, optimal maneuvering, and in-orbit servicing can be satisfied by an autonomous agent equipped with an extended version of this autonomy software solution.

Future development steps will enhance library scalability features as a distributed system. Enabling autonomous constellation management, cooperative operations, goal negotiation, intent prediction, and collective knowledge. Those are closely linked to a mission's architecture level and are entirely compatible and integrable with the single-agent autonomy capabilities. In addition, such an autonomy ecosystem informs ground or mission control centers too.

The proposed solution for onboard autonomy can be seen as the satellite's brain in which the OBC confers the platform the ability to operate in the mission environment autonomously. On a large scale, this approach will enable satellites cooperation even in the framework of complex operations, such as in-orbit servicing and proximity operations, avoiding the exponential increase of operators' workload and facilitating the adoption of more complex and more extensive constellations.

Use Case – Earth Observation Benchmark Scenario

An Earth Observation (EO) satellite is a space system designed and developed to perform several observations of the Earth's surface and atmosphere from the orbital environment, acquired data is usually in the form of digital imagery. Several platforms could be considered to perform EO tasks (drones, aerial platforms, etc.), the advantage of using satellites is to be able to rely on reliable and global coverage, even in areas that are normally inaccessible.

While considering an Earth Observation product, a set of parameters must be considered to properly evaluate the quality of the product itself; one of the most relevant characteristics of such imagery data is undoubtedly the cloud coverage level of the images acquired and downlinked. The cloud coverage concept refers to the perturbation due to the presence of clouds above the area of interest. The entire process of analyzing and buying an archive imagery dataset is extremely complex, slow, inefficient, and not so beneficial from a business point of view; normally the customer will ask for a specific maximum cloud coverage threshold and receives an estimated time window for the acquisition, which maximizes the probability of obtaining a cloud-free image. Unsuccessful observations will lead to an iterative process until the customer accepts the dataset.

Nominally, satellite operators will tailor mission operations to perform a predefined number of image acquisitions at a certain acquisition frequency; this approach results in generating a significant amount of data to be stored and downloaded during visibility windows. At present, because at any time, on average, the Earth is 67% covered by clouds, and considering the buying/selling pipeline, only an amount close to 1% of the downloaded data is profitable for EO companies. In this context, it was decided to apply MiRAGE to optimize the outcome of a common Earth Observation mission, limiting acquisitions to cases of lower cloud coverage and raising the percentage of profitable downlinked images.

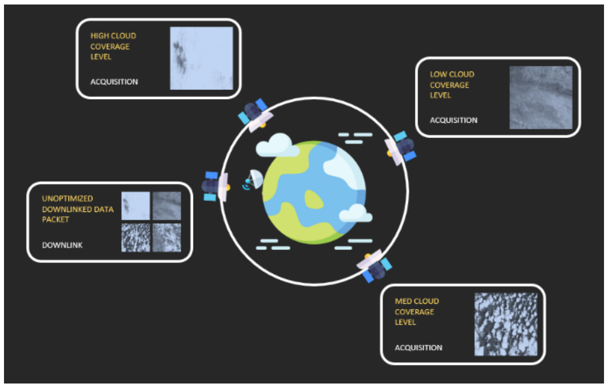

Figure 1: Common EO Scenario: acquisition modes are not affected by the quality of observations.

To test MiRAGE autonomy functionalities, applied to this EO scenario, several assumptions have been made about the mission, the satellite platform, and the mission-specific tasks to be accomplished.

Earth Observation platform definition

A smallsat equipped with two payload cameras has been considered. The two sensors work at different spatial resolutions; the high-performance payload will be used for the acquisition at lower cloud coverage levels, on the other hand, the low-res camera will monitor the cloud coverage level to detect when it is better to switch equipment. The acronyms adopted for the acquisition payloads are:

- High-Resolution Camera → HRC
- Low-Resolution Camera → LRC

Both cameras have the same field of view.

To simulate HRC acquisition, the scenario generates images that represent an uncompressed RAW format, weighing 72MB. Images are then compressed and fed to the Deep Learning algorithms as 256x256 RGB images. LRC images are captured in an uncompressed RAW format weighing 1MB and are also compressed and fed to the Deep Learning algorithms as 128x128 RBG images.

Concerning acquisition modes, the HRC will be able to operate at two acquisition frequencies.

A simple onboard memory is modeled to simulate the memory filling by storing images acquired by both HRC and LRC. Memory dimension is tailored according to the number of images nominally generated during an EO mission and the data rate that characterize a common downlink window. The onboard memory is assumed to be 8GB, which represents an average value for a smallsat onboard storage capacity.

Scenario Definition

The mission scenario considered is quite simple. The simulated satellite will switch its optical payloads to maximize the amount of lower cloud coverage acquisitions to be downlinked. This will increase the number of relevant images stored on the onboard memory, optimizing the data packets that must be downlinked. Several visibility windows are simulated so that MiRAGE can be aware of the moments in which the downlink can be started and how much of its memory can be freed.

Mission Goals and Tasks considered

MiRAGE will generate different high-level mission goals and tasks to be fulfilled. Specifically, three observation tasks (operative modes) are foreseen: HRC acquisition at high-frequency, HRC acquisition at low-frequency, and a monitoring task performed using LRC. In addition, a downlink high-level task is injected to simulate onboard data downlink to a ground station.

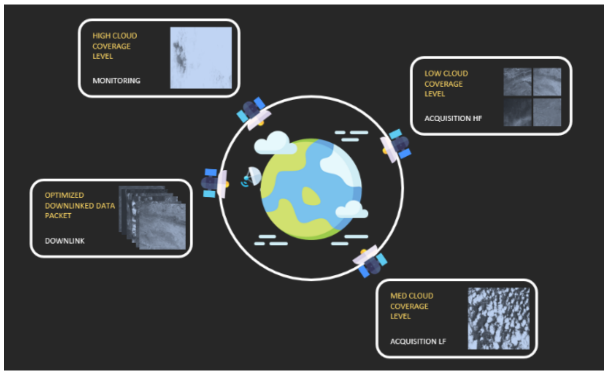

Figure 2: MiRAGE applied to EO Scenario.

AGENT ARCHITECTURE OVERVIEW

MiRAGE is based on a modular microservices architecture: the main process is responsible for orchestrating all the modules that form the software core and for the concurrent launch of the sub-process that simulates the two optical payloads.

The choice of a microservices structure was dictated by the need for agility in the software tailoring process with respect to the mission design, meaning that the number of modules is closely related to the mission requirements, which, instead, will not drive any modifications to the software core.

A simplified representation of MiRAGE architecture is shown below.

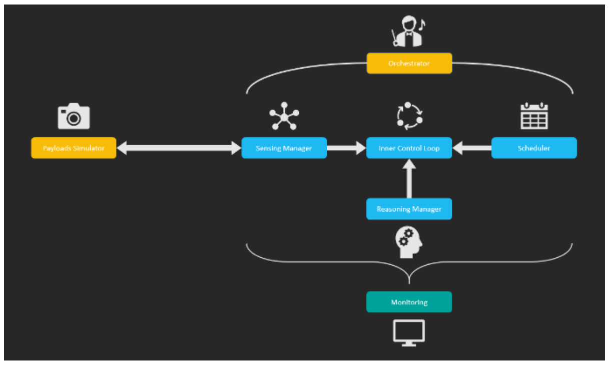

Figure 3: MiRAGE Simplified Architecture.

With reference to the previously described scenario, the only software item which is external to MiRAGE is the Payloads Simulator: in particular, it is deputed to read and edit specific files which act as placeholders for the simulated payloads settings, changed by the commands MiRAGE sends to the simulator itself.

The communication between each module and sub-module launched and managed by the Orchestrator is achieved through uniquely defined message channels. The totality of the software items constituting MiRAGE constantly sends updates to the Monitoring module, which collects the information about their functioning and the resources in use, to collect anomalies and to build the telemetry packets that will be eventually downlinked.

Finally, the MiRAGE core is defined by four main modules: the Sensing Manager, the Reasoning Manager, the Scheduler, and the Inner Control Loop. The last oversees executing each one of the tasks that are commanded by the Scheduler, other than embedding the MiRAGE time manager.

The Sensing Manager is where Deep Learning is applied to process the images made available by the Payloads Simulator. Flowing into the Inner Control Loop, the response of the Sensing Manager expands the context of the Reasoning Manager, allowing the formalization of a goal that is directly sent to the Scheduler, which analyses the onboard available resources to list in chronological order the tasks that shall be executed by the Inner Control Loop.

The main components of MiRAGE and the related technologies are further detailed in the following chapters.

SENSING MANAGER – PAYLOAD DATA PROCESSING

Deep Learning for Optical Paylod Data Processing

The increased complexity in space mission operations requires the adoption of innovative, high-performing solutions, for feature extraction. Recent advancements in Artificial Intelligence and Deep Learning enabled outstanding achievement in computer vision[2,3] and time-series analysis[4,5]. The reversed data-centric approach introduced by this innovative approach enables the definition of an implicit model, trained using supervision and capable of autonomously extract relevant features from data by backpropagating the neural network's parameters. Thus, the adoption of a Deep Learning based model to an innovative system like MiRAGE enables complex pipeline handling for processing data captured from satellite sensors.

The concept is applicable for example to spacecraft cameras, in which a Deep Convolutional Neural Network is used to firstly extract significant features from the images acquired and then passed to the Reasoning Manager for Event generation.

This also applies to a vast series of possible scenarios: from cloud detection to object tracking, super-resolution, and so on.

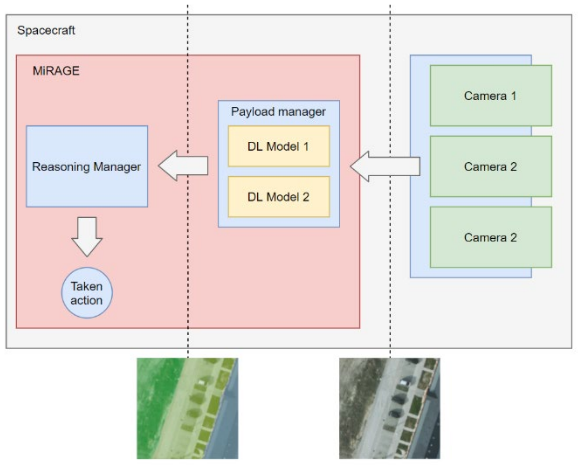

Figure 4: On-board Deep Learning Algorithms Execution Process.

Use-Case: Cloud Segmentation

As previously explained, the use of DL onboard poses new advantages in the Concept of Operations. Hereby a specific use case related to segmentation models applied to cloud detection is presented.

Many of the images taken onboard a satellite are cloudy and mostly useless for the purpose of the mission. Thus, the capability of onboard decision-making could decrease the workload on operators in Mission Control Centers.

The first application related to this specific use case is Clarity[6], a submodule of MiRAGE capable of detecting cloud coverage onboard by exploiting the state-of-the-art capabilities of a Deep Learning-based segmentation model.

A segmentation model has an encoder-decoder structure, which resembles U-Net[7], in which the input information (the image) is progressively processed, and the deeper the network the higher-level features are extracted.

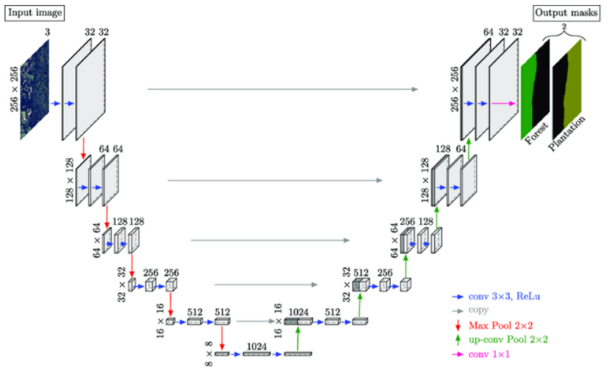

Figure 5: A U-Net-style neural network. The input image is compressed and progressively processed until the most informative layer is reached (the bottom layer of the U-shape), also called latent-space. The information is then re-expanded towards an output mask/s.

The output layer is trained to predict a binary mask in which zeros are representing no-cloud conditions and ones are representing cloudy conditions. The summation of the ones with respect to the original resolution of the image gives the cloud coverage of the image.

REASONING MANAGER

MiRAGE integrates an Expert System to infer structured information resulting from the processing of input parameters collected from telemetry, sensor data, payload data, mission data, and environmental data. Specifically, the depicted architecture integrates a reasoning manager, which embeds the core Expert System, to produce high-level mission goals and tasks to be fulfilled by the satellite according to an optimized mission schedule.

MiRAGE is built according to the knowledge-enabled programming[8] paradigm. The idea behind this paradigm is to make the algorithm or the program completely agnostic about the application scenario by separating the knowledge from the code itself; furthermore, the knowledge base is then modularized into small, broadly applicable, and reusable chunks. Concerning the reasoning manager, the knowledge base is composed of a propositional logic rule set, organized according to modular areas of pertinence describing the external environment, the mission, and the platform from the system level to the single component level.

MiRAGE Approach to Expert System Implementation

An Expert System is an AI software that 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 large 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.

Fuzzy Logic, the Input Layer

Fuzzy logic[9] is a form of propositional calculus in which the truth value of a variable lies in the real number domain, between 0 and 1 (inclusive). In contrast with the Boolean logic, where truth values of a variable may be only true or false (0 and 1), it is employed to handle the concept of partial truth. The concept of fuzzy logic is extremely important to represent the fact that people make decisions based on imprecise information; in this sense, these kinds of models are mathematical means of representing vagueness and imprecise information.

Mamdani fuzzy inference was first introduced as a method to create a control system by synthesizing a set of linguistic control rules obtained from experienced human operators[10]. In a Mamdani system, the output of each rule is a fuzzy set. MiRAGE reasoning manager embeds a Mamdani fuzzy inference system to process numerical data collected from different sources across the system platform (telemetry and sensor data) and other numerical parameters describing mission scenario and environmental aspects. Those kinds of systems are well-suited to expert system applications where the rules are created from human expert knowledge, such as the proposed application.

The diagram below illustrates how the Mamdani fuzzy inference, embedded in the reasoning manager's expert system, has been adopted in the context of the EO scenario.

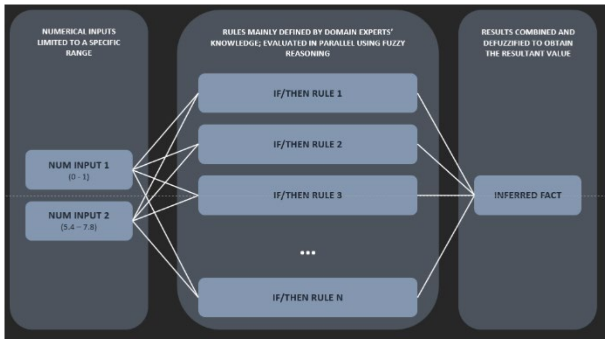
NUMERICAL INPUTS LIMITED TO A SPECIFIC RANGE
RULES MAINLY DEFINED BY DOMAIN EXPERTS' KNOWLEDGE; EVALUATED IN PARALLEL USING FUZZY REASONING
RESULTS COMBINED AND DEFUZZIFIED TO OBTAIN THE RESULTANT VALUE
NUM INPUT 1 (0 - 1)
NUM INPUT 2 (5.4 – 7.8)
IF/THEN RULE 1
IF/THEN RULE 2
IF/THEN RULE 3
...
IF/THEN RULE N
INFERRED FACT

Figure 6: Fuzzy inference process applied to the lowest level (input data) of the MiRAGE expert system's knowledge base.

Certainty Factors, Reasoning with Uncertainties

Certainty factors are applied to the rules stored in the knowledge base to deal with measurements and environment-related uncertainties. By doing so, the outcome of each rule could be true or false depending on a certain threshold. As an example, if a conclusion derived from different rule premises has a certainty factor of 0.6 and if a TRUE threshold value has been set to 0.8, the conclusion will be evaluated as FALSE or UNDEFINED. Certainty factor values associated with each derived or collected known fact are combined in parallel across the backward inference process.

Considering the presented benchmark EO scenario, the only sources of uncertainty considered are the outcome of the DL module used to process the payload data and the output variable of the fuzzified inference. Specifically, the certainty factor applied to the measured level of cloud coverage will be set to a value equal to the accuracy of the model itself. For the sake of simplicity, all the other certainty factors are set equal to 1 (always TRUE).

Use-Case Declination

The table below depicts the high-level structure of the knowledge base adopted for the benchmark EO scenario; input parameters and inferred facts are highlighted.

Table 1: Expert System Knowledge Base adopted for the EO Scenario.

Parameter	Description	Context	Level
CLOUD COVERAGE	Cloud coverage measured from the last acquired image	DETECTION	INPUT
DOWNLINK SCHEDULED	Flag to detect if a downlink task has already been scheduled	MISSION	MEDIUM
DOWNLINK STATUS	High-level downlink condition	MISSION	MEDIUM
GOAL	High-level goal to be pursued	RESPONSE	TARGET
MEMORY STATUS	High-level onboard memory condition	COMPONENT	MEDIUM
ONBOARD MEMORY	Onboard storage occupied	COMPONENT	INPUT
TASKS	Tasks to be fulfilled	RESPONSE	TARGET
WINDOW DISTANCE	Distance from next ground station visibility window	ORBIT	INPUT

AUTONOMOUS SCHEDULING

Once the Reasoning Manager has defined the goal that contains the set T of tasks needed to be performed by the satellite, what remains is to define the exact starting time of each task, that is the schedule of all tasks according to their duration, priority, resources usage and predefined precedence. At this point, an external module of the onboard real-time scheduler is invoked, and it is based on linear integer programming.

What the scheduler does more in detail is to optimize the starting time of each task and to try to schedule the greatest number of tasks using an appropriate fitness function. It works with discrete time intervals over a finite horizon H, so each starting time is an integer value in the interval [0, H].

Three parameters are considered: duration, schedule_cost, and delay_cost. The last two parameters are two expressions of the concept of priority. The schedule_cost is an expression of the task scheduling priority. The delay cost is an indicator of the importance of scheduling a task as soon as possible.

In addition, three categories of resources, that each task can occupy or consume, are considered:

- binary resources: cannot be used for more than one task at the same time and lead to non-overlapping constraints among activities.
- multiple resources: this category contains resources that are not consumed but have a maximum capacity that limits simultaneous use.
- consumable resources: can be consumed or generated in time.

It is necessary to consider also that some tasks can be scheduled only in limited intervals of the horizon (for example, the downlink activity must be performed during the window visibility with the ground station).

BENCHMARK SCENARIO RESULTS

With the premises described in the previous chapters, testing and validation were performed on the benchmark scenario, achieving important results.

After the scheduler initialization, which is defined through operations that are commonly performed after a satellite deployment (detumbling, orbit maneuver, point to target), the schedule produced is shown below.

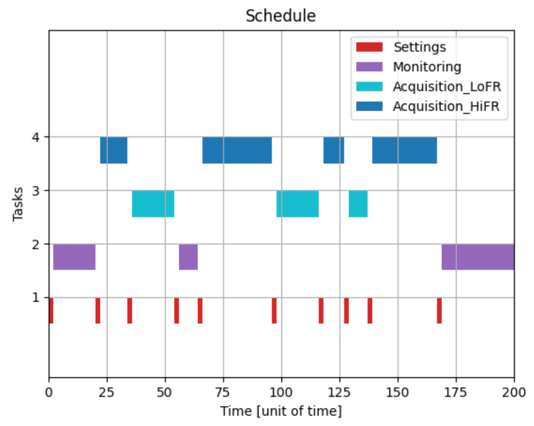

Figure 7: Schedule example: different observation tasks.

The granularity of the schedule in this validation environment is defined by the tasks' typology and by the scheduler horizon. In the benchmark scenario, the time units adhere to minutes, to have greater responsiveness during goal changes.

From Figure 7 it can be seen how the order of the tasks is tracing the definition of the goals: in fact, the payload configuration is always scheduled before the observation (this due to the tight relation between these two tasks). Clearly, the three observation modes (Monitoring, High and Low-Frequency Acquisition) alternate with respect to specific events triggered by the Sensing Manager, based on the calculated cloud coverage, remaining memory, and visibility windows. In Monitoring mode, only the LRC is activated aiming to the sole cloud coverage evaluation (no pictures are saved); in High-Frequency Acquisition mode, the HRC takes one image per second, while in Low-Frequency Acquisition mode the shooting rate is ten times slower.

In this regard, during development and during tests, the correct behavior of MiRAGE is assessed through the MiRAGE Monitoring Dashboard seen in the following figure.

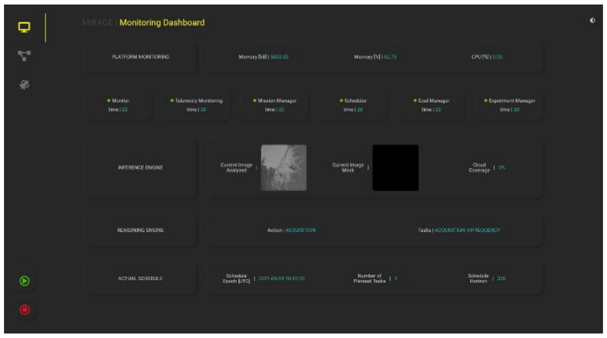

Figure 8: MiRAGE Monitoring Dashboard: HiFrequency Acquisition due to low values of cloud coverage.

Finally, the downlink scenario, in which the concurrency of some tasks can be observed, is shown in Figure 9. Data preparation, payload configuration, and attitude maneuvers can be performed simultaneously due to the different types of binary resources needed to achieve the tasks.

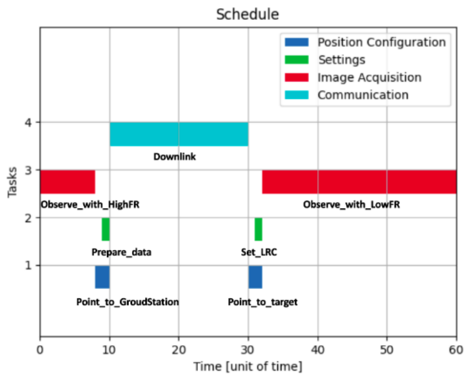

Figure 9: Schedule example: Downlink goal case.

CONCLUSIONS

Recent trends in the design and implementation of space missions are showing increasing capabilities in satellite platforms and payloads. These increased capabilities come with enormous potential to envision satellite operations that are more responsive, adaptable, and that generate data for the final user that is highly monetizable. To reach these results, it is mandatory to reconsider drastically the way operations are carried out on current space missions, and especially on Earth Observation ones. The technology presented in this paper introduces a revolutionary autonomy agent able to satisfy the increased demands of the new missions that are currently under design in these years. The use of such technology will result in satellites being more autonomous, less dependent on the mission control center, and laser-focused on prioritizing the best data available and providing them to the final user.

The use case presented in the paper demonstrates how an onboard AI-enabled goal-oriented operations manager can help in delivering responsive and adaptive operations in Earth Observation scenarios. The technology has been demonstrated in a Software-in-the-Loop simulation, presenting significant improvements on how images are acquired during the mission, and how the onboard memory usage is optimized, also considering external parameters such as the Ground Control Station distance. The software presented in the paper is currently at TRL 6 and is scheduled to be delivered for acceptance by Q3 2021, thus reaching TRL 8.

REFERENCES

1. Jet Propulsion Laboratory. 2019. Strategic Technologies 2019. Jet Propulsion Laboratory, California Institute of Technology (Pasadena, CA)
2. Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey E., (2012) ImageNet Classification with Deep Convolutional, Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, pp. 1097–1105.
3. Christian Szegedy and Wei Liu and Yangqing Jia and Pierre Sermanet and Scott E. Reed and Dragomir Anguelov and Dumitru Erhan and Vincent Vanhoucke and Andrew Rabinovich (2014). Going Deeper with Convolutions. arXiv, 1409.4842.
4. Sepp Hochreiter and Jürgen Schmidhuber (1997). Long Short-Term Memory. Neural Computation - Volume 9, pp. 1735-1780.
5. Bryan Lim and Sercan O. Arik and Nicolas Loeff and Tomas Pfister (2020). Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. arXiv, stat.ML, 1912.09363.
6. Website: https://www.aikospace.com/#clarity
7. Olaf Ronneberger and Philipp Fischer and Thomas Brox (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv, 1505.04597.
8. Beetz, Michael, Dominik Jain, et al. (2012). "Cognition-Enabled Autonomous Robot Control for the Realization of Home Chore Task Intelligence". In: Proceedings of the IEEE 100.8, pp. 2454–2471.
9. C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller" in IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 2, pp. 419-435, March-April 1990, doi: 10.1109/21.52552.
10. E.H. Mamdani, S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, International Journal of Man-Machine Studies, Volume 7, Issue 1, 1975, Pages 1-13, ISSN 0020-7373, doi: 10.1016/S0020-7373(75)80002-2.