OnAIR: Applications of The NASA On-Board Artificial Intelligence Research Platform

Evana Gizzi[1], Timothy Chase Jr[1], Christian Cassamajor-Paul[2], Rachael Chertok[3], Lily Clough[4], Connor Firth[4], Alan Gibson[1], Ibrahim Haroon[5], James Marshall[1], Patrick Maynard[1], Michael Monaghan[1], Hayley Owens[6], Daniel Rogers[1], Mahmooda Sultana[1], Jivko Sinapov[6], Bethany Theiling[1]

[1]NASA Goddard Space Flight Center
[2]Massachusetts Institute of Technology
[3]University of Vermont
[4]Aurora Engineering, Inc.
[5]University of Massachusetts Lowell
[6]Tufts University
evana.gizzi@nasa.gov

Abstract

Infusing artificial intelligence algorithms into production aerospace systems can be challenging due to costs, timelines, and a risk-averse industry. We introduce the Onboard Artificial Intelligence Research (OnAIR) platform, an open-source software pipeline and cognitive architecture tool that enables full life cycle AI research for on-board intelligent systems. We begin with a description and user walk-through of the OnAIR tool. Next, we describe four use cases of OnAIR for both research and deployed onboard applications, detailing their use of OnAIR and the benefits it provided to the development and function of each respective scenario. We conclude with remarks on future work, future planned deployments, and goals for the forward progression of OnAIR as a tool to enable a larger AI and aerospace research community.

Code — https://github.com/nasa/OnAIR

1 Introduction

The aerospace sector contains diverse research problems that provide fruitful domains for artificial intelligence (AI) research applications. For example, traversing remote planetary environments in the search for life necessitates autonomy due to the challenges associated with human-in-the-loop ground control stations, which introduce latency, proximity, and bandwidth limitations (National Academies of Sciences et al. 2018). Similarly, reacting to space weather events on Mars in the quest for building remote habitable worlds will require on-board data processing and modeling, which is challenging to enable in space due to its unique computational constraints (National Research Council et al. 2013). Enabling the integration of AI into aerospace systems will drastically transform what is possible in space, and the need for this intelligence has existed for quite some time. Applications such as intelligent distributed systems missions (e.g., swarms and constellations), onboard data mining, and autonomous guidance navigation and control in space systems all require contribution from AI research (for example, multi-agent systems, machine learning/computer vision, and reinforcement learning, respectively) (Dahl et al. 2023).

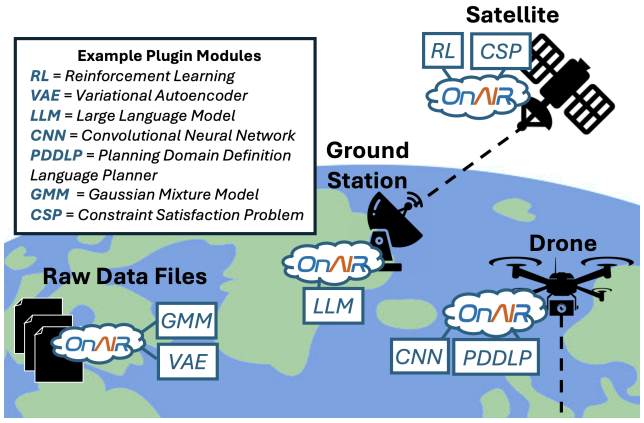


Figure 1: OnAIR can interoperate with a range of systems of varying fidelity and complexity, thus supporting heterogeneity and generalizability. OnAIR has been successfully integrated with flight software on-board satellites, drones, robots, data from ground stations, and using raw data files.

Despite the exciting possibilities for AI in space, the aerospace sector is traditionally conservative and risk-averse. Space research contains a high barrier to entry due to mission sensitivity, inaccessibility, and cost of flight missions, making the infusion of current low-maturation, low-assurance, and low-trust AI algorithms a risky endeavor. To this end, we present the On-Board Artificial Intelligence Research Platform (OnAIR), a Python-based software pipeline and cognitive architecture designed to unify and standardize AI research and development for space systems. OnAIR offers a robust and easy-to-use framework that reduces timelines/barriers of entry for AI researchers, expediting the integration of AI algorithms into space applications. OnAIR has been used to facilitate numerous diverse research and mission applications at various stages of maturation, includ-

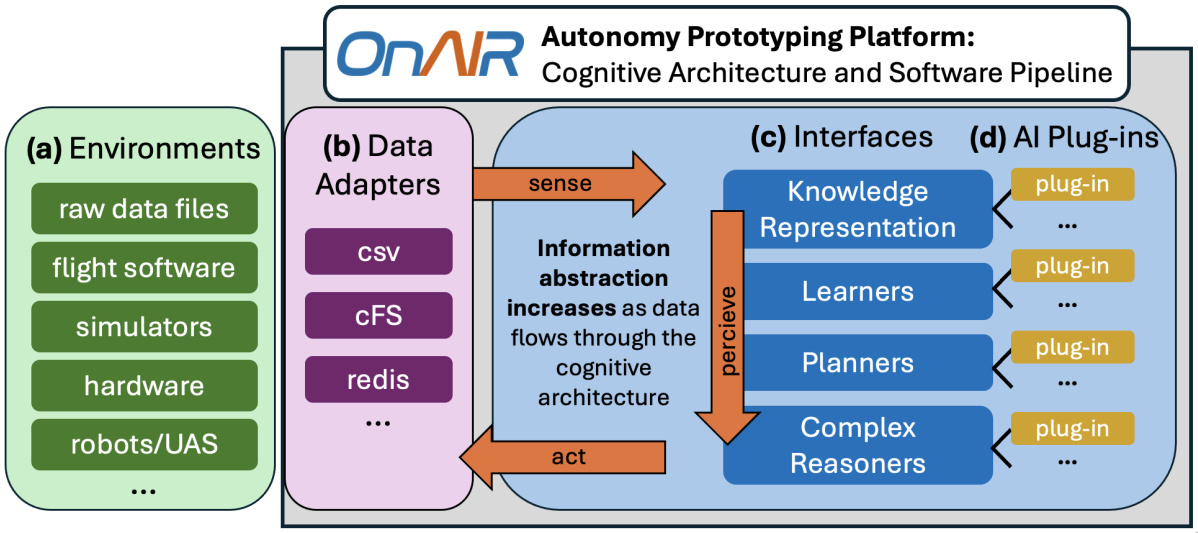


Figure 2: System diagram of OnAIR and its interaction with external environments (a). Data from the environment is ingested using a user-chosen data adapter (b) to provide agent "sensing". Data flows across four plugin interfaces of increasing knowledge abstraction to emulate the neurological premise of cognition, providing the structure for agent "perception". Finally, outputs from the complex reasoners interface, meant to provide the structure of executive control reasoning, are sent into the environment in the form of agent "action".

ing two successfully deployed NASA missions as a tool to enable AI reasoning on board.

In this paper, we provide background on the OnAIR tool, a walk-through of its use, and a review of its successful deployments to a range of research projects and missions, all at varying stages of maturation and across disparate domains. Additionally, we describe how OnAIR has enabled diverse researchers with varying backgrounds to work together under one architecture to develop AI algorithms and tools for space systems. OnAIR is actively maintained and developed as part of an internal NASA project; community feedback and contributions are welcome.

2 Related Work

The infusion of artificial intelligence algorithms into space has been ongoing since the 1990s, with its early use in "Spike", Deep Space 1, and Earth Observing 1 (Dahl et al. 2023). However, each of these missions employed their custom architectures for AI infusion. More recently, there have been efforts to create standard flight software frameworks like NASA's Core Flight System (cFS) and F-Prime (McComas, Wilmot, and Cudmore 2016; McComas 2012; Bocchino et al. 2018a,b), both of which have supported major missions, and both of which are open source. Although these architectures offer publish-subscribe capabilities, they are not explicitly engineered for AI research. Alternatively, the SpaceROS environment offers similar features to the Robot Operating System (ROS) (Macenski et al. 2022), but has been repurposed for use in space, with more emphasis on memory and timeliness safety (Probe et al. 2023). Implementing autonomous functions within these related architectures is challenging due to the complexity of connecting numerous disparate components, where it would be more beneficial to have an all-inclusive cognitive architecture in place for autonomous functions. To the best of our knowledge, OnAIR is the first ever cognitive architecture for space, flown in space (see Section 5 for details of flight test). As AI deployment in space increases, AI architectures will be essential to streamline processing and decision-making, collectively emitting intelligence as opposed to hard-wiring many individual components which may become burdensome or even unfeasible in sufficiently large autonomous functions for spacecraft.

3 The OnAIR Cognitive Architecture

OnAIR was developed at NASA Goddard Space Flight Center (GSFC) to support the interoperation of diverse researchers to collectively enable autonomy for fleet mission types, under the Distributed Systems Mission (DSM) project (Gramling et al. 2022). The OnAIR tool was successful in combining the efforts of multi-disciplinary researchers with varying software engineering/AI experience across multiple development environments at various stages of maturation, discussed later in this paper. The primary objective of deploying OnAIR is to fuse independent systems at low cost to development teams, to allow for parallel development of those systems, and to provide the structure of dataflow and timing to standardize operations across a scenario.

OnAIR is a plugin-based software pipeline and cognitive architecture that provides modular, interchangeable, user-designed components with an implementation that follows the Russell and Norvig (Russell and Norvig 2010) *sense-perceive-act* structure of a rational agent. Sensing happens

Interface Plugin	Description	Example
`Knowledge Representation` ($\mathcal{P}_{KR}$)	Knowledge Representation plugins synthesize high-level knowledge from the raw environmental dataframes in $\mathcal{F}$ according to user-defined algorithms.	Planning Domain Definition Language (PDDL) symbolic predicate descriptors (`arm_lifted()`) are synthesized from sub-symbolic robot end effector trajectory information.
`Learners` ($\mathcal{P}_L$)	Learner plugins receive both the high-level information generated by $\mathcal{P}_{KR}$, and low-level dataframes in $\mathcal{F}$	Sub-symbolic feature data from $\mathcal{F}$ and symbolic labels generated from $\mathcal{P}_{KR}$ train a vision transformer neural network
`Planners` ($\mathcal{P}_P$)	Planner plugins receive high-level information from $\mathcal{P}_{KR}$ and $\mathcal{P}_L$ to be used by high-level planners, and can additionally append these recommendations to the f.	A Task and Motion Planner (TAMP) uses state information and weather predictions to deduce science instrumentation decisions.
`Complex Reasoners` ($\mathcal{P}_{CR}$)	Complex Reasoner plugins receive high-level outputs from all plugins, and combine them into an "executive function" decision-making process.	An ensemble decision-making approach is employed, where both path planning in $\mathcal{P}_P$ and robot battery-level prediction in $\mathcal{P}_L$ are combined to make optimized search and rescue locomotive action decisions.

Table 1: An overview of OnAIR's interface plugin types, with specific implementation examples. Each user-designed plugin falls into one of these categories.

through a user-defined data adapter (Figure 2b). Perception takes place across four levels of abstraction meant to emulate the neurobiological premise of cognition (Figure 2c). Decision-making and actuation happen at the highest level of cognition, meant to emulate executive control reasoning of the human brain, and to support ensemble-based algorithms. Each abstraction layer is represented by a plugin interface. Please see Table 1 for descriptions and examples of each abstraction layer.

Benefit of Using OnAIR

OnAIR's use in both research and deployment scenarios is to enable integration when development requires bringing parallel systems together to accomplish holistic tasks. In the following research and deployment sections, the systems discussed are comprised of independent systems attached to OnAIR as plugins. OnAIR provides the structured, regulated data flow required for sequential processing and inter-plugin communication, mirroring cognitive flow, and enabling the independent components to create a full functionality loop. This binding of disparate systems reduces development, deployment, and maintenance costs and generalizes to a range of tasks, scenario fidelity, and problem complexity. OnAIR's ease of use is proven by its use in teams to accomplish research, production, and implementation tasks across a range of environments and timescales, including applications in opportunistic science discovery (Theiling et al. 2022, 2024), mission resilience (Gizzi et al. 2022b; Staley et al. 2023), creative problem-solving in robots (Gizzi et al. 2022a), and intelligent distributed systems. In Section 4 and Section 5, we review four use cases where OnAIR facilitated research and development for projects at varying maturation levels in different domains.

How to Use OnAIR

In this section, we describe how to use the OnAIR platform, shown in Figure 2.

- **Configurations:** The user begins by specifying which data adapter $\mathcal{DA}$ will be used for the target environment $\mathcal{E}$. For example, to load raw data from a `.csv` file, the user must specify the data file path and the `csv adapter` within the provided `config.ini` file. The data adapter is the main way of integrating the OnAIR pipeline with external tools and workflows, and the open-source repository provides three example adapters for use at the time of writing: `csv`, `Redis`, and `cFS`.
- **Data Pipeline:** At runtime, data from the environment $\mathcal{E}$ begins piping into the software pipeline of OnAIR through specified data adapter $\mathcal{DA}$ (Figure 2a and Figure 2b). $\mathcal{DA}$ process data from E into a `DataSource` object $\mathcal{DS}$, which buffers this processed data into $\mathcal{F}$, which is a list of frames f, where f is a fixed array of heterogeneous data of size n.
- **Cognitive Architecture:** The cognitive architecture of OnAIR *senses* information about the environment through the `DataSource` object, which regularly renders `frames` from its buffer $\mathcal{F}$. OnAIR contains four main plug-in interfaces, shown in Figure 2c and described in Table 1. Each plugin interface can house a non-fixed amount of user-define python plugin modules. OnAIR requires all plugins to implement an `update` function (which receives a frame f) and a `render_reasoning` function.

The full OnAIR stack is open source, allowing for the construction of DataSources and additional plugins as required. Documentation is extensive, including high-level architecture design, component descriptions, plugin development, unit testing, and full end-to-end examples. To date,

Figure 3: OnAIR has been used for AI prototyping in a range of extremely diverse use cases. **(top, left)** NASA GSFC/Mach 33 custom-built small unmanned aerial vehicle (sUAS) which used OnAIR as an autonomy/software system prototyping architecture through all stages of sUAS development. **(top, right)** OnAIR for image and spectroscopy processing on ModalAI Starling drones and Turtlebots. **(bottom, left)** NASA drone mission simulation using PyBullet. **(bottom, middle)** NASA drone mission simulation using AirSim. **(bottom, right)** multi-agent system of Turtlebots performing distributed tasks.

OnAIR has reached roughly 30 users and open-source contributors spanning both internal and external domains. Internal and community collaborators have configured OnAIR to run in concert with various simulators (Gazebo, PyBullet, JMAVSim, AirSim), technologies (Robot Operating System, PX4, I2C, MAVLink, R, Docker), and embodied agents (Turtlebots, ModalAI Starlings, custom drones), demonstrated in Figure 3.

4 Research Applications of OnAIR

OnAIR has been used as a prototyping research tool for numerous early-stage research applications with NASA and in collaboration with university and industry partners. This section contains examples of OnAIR's uses in autonomy research across multiple institutions and disciplines to enable and streamline the development process.

Crop Health and Contaminant Detection

Researchers from NASA Goddard Space Flight Center, Tufts University, University of Vermont, and Massachusetts Institute of Technology used OnAIR to develop a framework for identifying and characterizing saliency in agricultural data, detecting unhealthy soil using a twin network and classifying the type of deviation; see Figure 4 for an example case. The system was built from the ground up within OnAIR. Researchers built out unique features to enable cross-network communication between agents (and their OnAIR instances) geolocated near their three campuses. These systems were built around OnAIR's data ingestion and were easy to integrate using the compartmentalized data source component. For algorithmic development, OnAIR was able to switch easily between data fidelity levels, accepting data from stock images, simulations, or local Raspberry Pi sensors, with no changes required to follow-on plugins. Agents in heterogeneous environments could additionally communicate seamlessly (e.g. sim agent and Pi sensor agent). OnAIR facilitated distributed agent communication, and data source configuration to accept knowledge synthesized by other agents. Algorithms for data processing, saliency detection, and cross-communication were developed and integrated in parallel by different teams as plugins, and the structured sequential dataflow forced accounting for each translation and reasoning step in the detection pipeline.

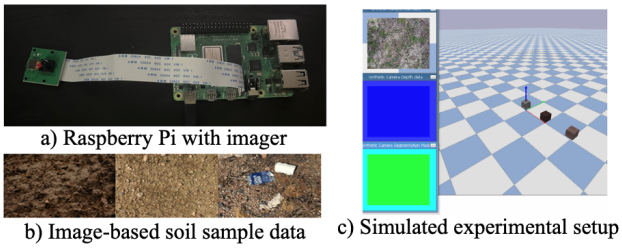


Figure 4: Intelligent detection and isolation of contaminants in soil. **(a)** A Raspberry Pi and imager are used to perform a real-world experiment on the contaminant search problem. **(b)** Images of soil used in experimentation. **(c)** A drone-based contaminant search problem is simulated in PyBullet using images of soil, which are fed into OnAIR and used to detect salient samples.

Mission Operations and Biosignature Detection

A collaboration between Aurora Engineering, the University of Tulsa, and NASA Goddard Space Flight Center (Williams et al. 2024) is currently making use of OnAIR as a mission operations research tool, with OnAIR as the focal point connecting several otherwise disparate and difficult-to-coordinate systems. The team is working on a proposed use case around Saturn's moon Enceladus, involving a constellation of orbiters studying the plume ejecta above the surface using mass spectrometers (Figure 5). Several instances of OnAIR run on Raspberry Pis, each simulating an orbiter using prepackaged navigational and science telemetry of the simulated mission run, are fed to OnAIR instances through static data sources and processed by onboard algorithms. OnAIR handles each agent's datastream and integrates the backend of all simulated systems (simulated orbiter trajectories and telemetry, mass spectra collection) with the onboard autonomy processes, which classify mass spectra, detect possible biosignatures, and communicate with other agents. The classification code developed in R was integrated with OnAIR through a single plugin wrapping the operations in Python. Agents communicate through a Redis server, which mocks the more complex *in situ* protocols to speed development and iteration and is supported natively in OnAIR. OnAIR unites the disparately developed systems for telemetry processing, machine learning classification methods, and communications that were developed separately with minimal modification by each user group, enabling rapid integration and structuring of data flow across agents.

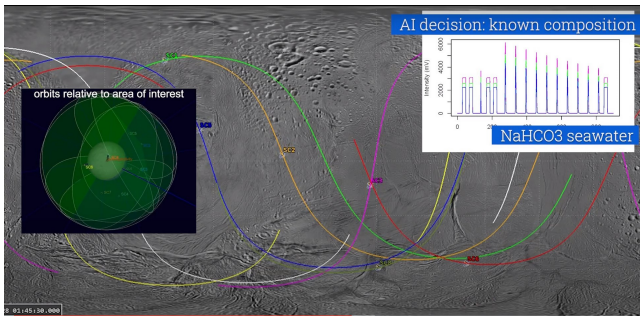


Figure 5: Screen capture of simulated Enceladus constellation mission. Eight SmallSats orbit Enceladus, shown in the inset (left) and as groundtracks in a 2D representation of Enceladus (background). All data analysis and computations (right) are performed in real-time during the simulated mission (Williams et al. 2024).

5 Deployed Applications of OnAIR

OnAIR has been used as a prototyping research tool and deployment architecture for on-board artificial intelligence algorithms for two NASA research missions, described below. This section describes each mission, its instantiation of OnAIR, and how OnAIR streamlined the development process and enabled onboard AI.

Network for Assessment of Methane Activity in Space and Terrestrial Environments (NAMASTE)

The Network for Assessment of Methane Activity in Space and Terrestrial Environments (NAMASTE) is a three-year (2022-2025) research campaign led by Dr. Mahmooda Sultana (NASA GSFC) for characterizing the methane distributions in the permafrost sites of Alaska (funded by Planetary Science and Technology Analog Research (PSTAR)). Due to the challenging terrain, autonomous drones are deployed to navigate the landscape and perform methane measurements (see Figure 6a,b,c). Intelligent decision-making is required among a fleet of drones to identify areas of high scientific interest while minimizing redundant or otherwise uninteresting measurements, and maximizing science-data throughput and understanding in a limited timeline. OnAIR aided research, development, and deployment by providing a standard for data ingestion and interaction between the many subsystems involved in the NAMASTE software architecture, including the autonomy planning and control stack.

OnAIR Setup OnAIR plugins were developed for multiple stages, most notably for science data processing, drone commanding and telemetry, and path planning, to constitute the main intelligent sense-and-seek functionality of the mission. The science data processing plugin (Figure 6c, "Science") ingested the complex raw sensor readings embedded in the OnAIR dataframe f using a Redis adapter, and outputted a single methane concentration value. The drone commanding and telemetry plugins use MAVLink, a standard protocol for communicating with various unmanned vehicles, to control and add GPS information from the UAV to f (Figure 6c, "GPS to loc"). The path planning plugin was developed to reason about the methane concentration value, GPS information, and current mission status within f, outputting the next target location for the drone leveraging standard grid or gradient-based searching algorithms that consider the extent and direction of maximum methane concentration (Figure 6c, "EDP" and "Actuation").

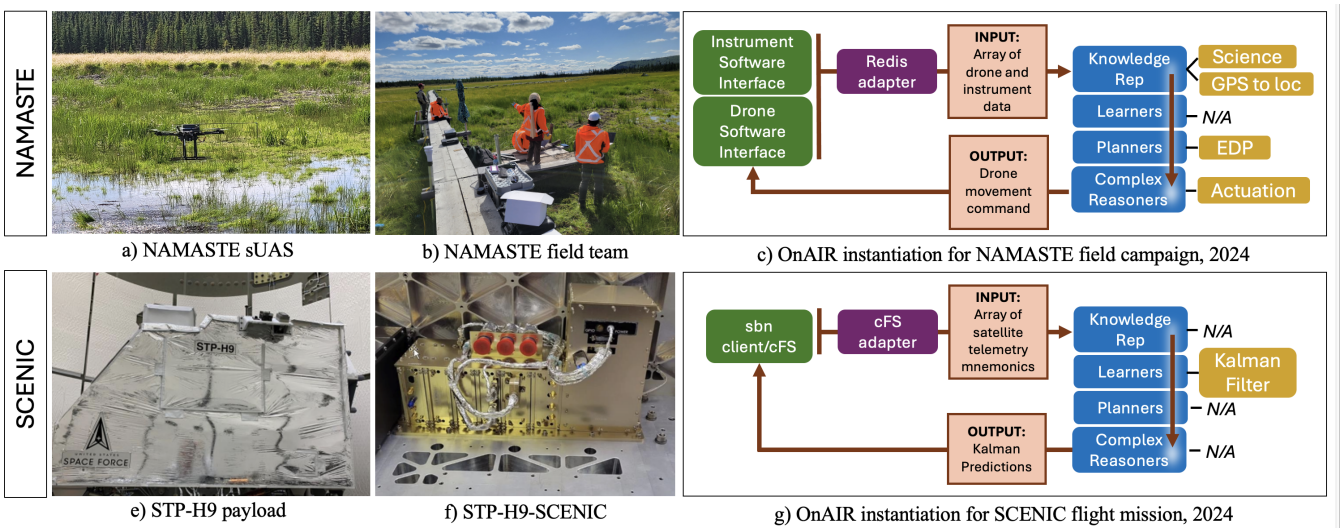
Results The NAMASTE mission is currently ongoing, with the second field campaign having been completed in the Summer of 2024. OnAIR was an integral factor in accomplishing the research goals accomplished in the recent field campaign. OnAIR allowed the diverse, distributed, and multi-experienced team to accomplish interoperability and software/hardware maturation on an aggressive agile timeline. OnAIR enabled interoperability in just two months of development (in a three-year project). OnAIR also allowed separate teams to continue parallelized development as various portions of the campaign matured at different times due to the diverse nature of project contribution. Additionally, OnAIR enabled the development of all software interfaces, including the custom methane sensor interface, the sUAS interface, the radio communication interface, and all AI module interfaces.

SpaceCube Edge-Node Intelligent Collaboration (SCENIC)

The SpaceCube Edge-Node Intelligent Collaboration (SCENIC) (Geist et al. 2023) payload was deployed on Space Test Program - Houston 9, a satellite tethered to the International Space Station (ISS) that launched in 2023 (led by Dr. Christopher Wilson, funded by the Space Force) (Figure 6e,f,g). In conclusion of the primary mission, the satellite hardware was made available for additional experiments, where OnAIR was chosen for an onboard AI demonstration. This posed several challenges, however, including I.) a six-month timeline with a small team, II.) a resource-constrained processor with a specialized instruction set architecture (FPGA-based platform running at 100 MHz), and III.) the need to integrate with the existing publish-subscribe flight software system (cFS) operating in the C language. For brevity, we refer interested readers to (McComas 2012; McComas, Wilmot, and Cudmore 2016) for a full account and description of the cFS architecture.

OnAIR Setup The short timeline forced a minimal experiment that would demonstrate OnAIR's functionality in a low-level autonomous function. Two plugins were developed to support this goal: a Kalman filter to operate over all incoming telemetry and a plugin to write the dataframe and Kalman filter residuals to a file. The limited computing power of the 100 MHz processor also limited what algorithms could be implemented, but it proved sufficient for the performance overheads introduced by OnAIR and the Kalman filter implementation.

Data was ingested by OnAIR through the cFS Software Bus Network (SBN), a client library to connect to the SBN, and the SBN data adapter available in the open-source OnAIR release. The SBN data adapter subscribed to an existing telemetry packet that consisted of forty telemetry points



a) NAMASTE sUAS b) NAMASTE field team c) OnAIR instantiation for NAMASTE field campaign, 2024

e) STP-H9 payload f) STP-H9-SCENIC g) OnAIR instantiation for SCENIC flight mission, 2024

Figure 6: OnAIR was used to enable autonomy research, development, and deployment on-board two NASA missions. **(a, b)** Images from the Summer 2024 field campaign of NAMASTE in Fairbanks, AK. **(c)** Instantiation of OnAIR to support research and deployment of NAMASTE. **(e, f)** Images from the Spring 2024 SCENIC Flight mission. **(g)** Instantiation of OnAIR to support research and deployment of the SCENIC.

and was sent at a rate of 0.25 Hz. The cFS to OnAIR data pipeline had been developed as part of a previous demonstration but required updates to work with the cFS version Caelum and the latest version of OnAIR.

Results OnAIR and the required flight software updates were successfully uploaded to SCENIC. The onboard processor was rebooted and OnAIR successfully launched. OnAIR periodically wrote the results to a csv file in a directory that was automatically downlinked and later reconstructed on the ground. OnAIR ran without error for 4 days alongside cFS, at which point the experiment was stopped in favor of other mission priorities. The SCENIC/OnAIR experiment was a success: a small team was able to implement plugins that process live spacecraft telemetry in a short amount of time. OnAIR plugin development was the easiest part of the project: more time was spent adapting to the FPGA instruction set and integrating with the SCENIC hardware. Using OnAIR made it possible to write, integrate, and test these plugins rapidly and independently, first using historical data through the `csv` data source and eventually with live data from cFS running on a SCENIC laboratory testing environment. OnAIR made it possible to experiment in such a short timeframe, facilitating rapid reuse of components and easy prototyping capabilities. OnAIR proved instrumental in the rapid development of an autonomous function, enabling accelerated development time compared with traditional pipelines that leverage the C coding language.

6 Future Work

The OnAIR development team has continued feature development and maintenance of OnAIR, with expanded contributions from the public research community. The next planned use of OnAIR will be to support the three-year NASA GSFC "Autonomous Science and Technology for Responsive Adaptability" internally funded research and development effort in collaboration with academic and industry partners. OnAIR will serve as the standard cognitive architecture for the heterogeneous swarm assets in this research. Additionally, OnAIR will be used to support single and multi-node mission resilience at NASA GSFC. OnAIR is currently in the research plans for upcoming missions in snow hydrology, space weather, satellite resilience, remote sensing (spectroscopy), exoplanet detection, and more (all at various stages of maturation in their research) to support reasoning in drones, rovers, ground controller stations, and satellites.

Due to its open-source availability, OnAIR has been organically used by the greater research community (industry and academic), and we plan on continuing to encourage this use and contribution. In future work, we will host regular hackathons and information sessions to enable the general research community to engage, use, and contribute to the code base. We are excited to see that OnAIR has served as a "bridge" tool, to ease AI researchers into the aerospace domain, and to increase AI literacy for aerospace researchers.

7 Acknowledgments

We would like to acknowledge the following interns who have contributed to OnAIR (and its earlier iterations): Ibrahim Haroon, Jeffrey St. Jean, Nicholas Pellegrino, Gabriel Rasskin, James Staley, William Zhang, and Charles Zhao. Thank you to the contributors to the OnAIR open source repository github. Thank you to the Goddard Space Flight Center Internal Research and Development (IRAD) program and the Distributed Systems Missions project for supporting this work. Thank you to the NAMASTE and SCENIC teams for their contributions to this work.

References

Bocchino, R.; Canham, T.; Watney, G.; Reder, L.; and Levison, J. 2018a. F Prime: an open-source framework for small-scale flight software systems. In *The 2018 Small Satellite Conference*.

Bocchino, R.; Canham, T.; Watney, G.; Reder, L.; and Levison, J. 2018b. F Prime: an open-source framework for small-scale flight software systems.

Dahl, M.; Page, C.; Cahoy, K.; and Gizzi, E. 2023. CDeveloping Intelligent Space Systems: A Survey and Rubric for Future Missions. In *2023 Small Satellite Conference*.

Geist, A.; Crum, G.; Brewer, C.; Afanasev, D.; Sabogal, S.; Wilson, D.; Goodwill, J.; Marshall, J.; Perryman, N.; Franconi, N.; et al. 2023. Nasa spacecube next-generation artificial-intelligence computing for stp-h9-scenic on iss. In *Proceedings of the Small Satellite Conference*, SSC23-P1-32. AIAA/USU.

Gizzi, E.; Nair, L.; Chernova, S.; and Sinapov, J. 2022a. Creative problem solving in artificially intelligent agents: A survey and framework. *Journal of Artificial Intelligence Research*, 75: 857–911.

Gizzi, E.; Owens, H.; Pellegrino, N.; Trombley, C.; Marshall, J.; and Sinapov, J. 2022b. Autonomous System-Level Fault Diagnosis in Satellites using Housekeeping Telemetry. In *Small Satellite Conference*.

Gramling, C.; Crum, G.; Dosberg, M.; Gizzi, E.; Green, C.; Hill, J.; Johnson, M.; Mauldin, K.; Morgentsern, R.; Roberts, C.; and Schiff, C. 2022. NASA's Goddard Space Flight Center's Distributed Systems Missions Architecture. In *International Astronautical Congress 2022*.

Macenski, S.; Foote, T.; Gerkey, B.; Lalancette, C.; and Woodall, W. 2022. Robot operating system 2: Design, architecture, and uses in the wild. *Science robotics*, 7(66): eabm6074.

McComas, D. 2012. NASA/GSFC's Flight Software Core Flight System. In *Flight Software WorkshopFlight Workshop*, GSFC. CPR. 7525.2013.

McComas, D.; Wilmot, J.; and Cudmore, A. 2016. The core flight system (cFS) community: Providing low cost solutions for small spacecraft. In *Annual AIAA/USU Conference on Small Satellites*, GSFC-E-DAA-TN33786.

National Academies of Sciences; Division on Engineering and Physical Sciences; Space Studies Board; and Committee on the Review of Progress Toward Implementing the Decadal Survey Vision and Voyages for Planetary Sciences. 2018. *Visions into Voyages for Planetary Science in the Decade 2013-2022: A Midterm Review*. National Academies Press.

National Research Council; Division on Engineering and Physical Sciences; Aeronautics and Space Engineering Board; Space Studies Board; and Committee on a Decadal Strategy for Solar and Space Physics (Heliophysics). 2013. *Solar and space physics: A science for a technological society*. National Academies Press.

Probe, A.; Oyake, A.; Chambers, S. W.; Deans, M.; Brat, G.; Cramer, N. B.; Kempa, B.; Roberts, B.; and Hambuchen, K. 2023. Space ros: An open-source framework for space robotics and flight software. In *AIAA SCITECH 2023 Forum*, 2709.

Russell, S.; and Norvig, P. 2010. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition.

Staley, J.; Lu, K.; Short, E. S.; and Gizzi, E. 2023. A Framework for Multi-Agent Fault Reasoning in Swarm Satellite Systems.

Theiling, B. P.; Brandt, M. A.; Clough, L. A.; Crum, G.; Gizzi, E.; Gramling, C. J.; Green, C.; Johnson, M.; Korde-Patel, A.; Mackinnon, J. P.; et al. 2022. Using Coordinated, Multi-Agent Platforms for Dynamic Ocean Worlds Science. In *AGU Fall Meeting Abstracts*, volume 2022, P23A–05.

Theiling, B. P.; Yu, W.; Clough, L. A.; Galchenko, P.; Naikal, F.; Da Poian, V.; McKinney, B. A.; and Gizzi, E. 2024. A Science-Focused Artificial Intelligence (AI) Responding in Real-Time to New Information: Capability Demonstration for Ocean World Missions. In *Astrobiology Science Conference (AbSciCon)*.

Williams, C.; McKinney, L.; Clough, L.; Theiling, B.; and McKinney, B. 2024. Autonomous Science: Simulated SOlar System Mission Enceladus, Icy Ocean Moon of Saturn. In *NASA TURC*. NASA.