AERCam Autonomy: Intelligent Software Architecture for Robotic Free Flying Nanosatellite Inspection Vehicles

Steven E. Fredrickson[*], Steve G. Duran[†], Angela N. Braun[‡], Timothy M. Straube[§], and Jennifer D. Mitchell[**]
NASA Johnson Space Center, Houston, Texas 77058

The NASA Johnson Space Center has developed a nanosatellite-class Free Flyer intended for future external inspection and remote viewing of human spacecraft. The Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam) technology demonstration unit has been integrated into the approximate form and function of a flight system. The spherical Mini AERCam Free Flyer is 7.5 inches in diameter and weighs approximately 10 pounds, yet it incorporates significant additional capabilities compared to the 35-pound, 14-inch diameter AERCam Sprint that flew as a Shuttle flight experiment in 1997. Mini AERCam hosts a full suite of miniaturized avionics, instrumentation, communications, navigation, power, propulsion, and imaging subsystems, including digital video cameras and a high resolution still image camera. The vehicle is designed for either remotely piloted operations or supervised autonomous operations, including automatic stationkeeping, point-to-point maneuvering, and waypoint tracking. The Mini AERCam Free Flyer is accompanied by a sophisticated control station for command and control, as well as a docking system for automated deployment, docking, and recharge at a parent spacecraft. Free Flyer functional testing has been conducted successfully on both an airbearing table and in a six-degree-of-freedom closed-loop orbital simulation with avionics hardware in the loop. Mini AERCam aims to provide beneficial on-orbit views that cannot be obtained from fixed cameras, cameras on robotic manipulators, or cameras carried by crewmembers during extravehicular activities (EVA's). On Shuttle or International Space Station (ISS), for example, Mini AERCam could support external robotic operations by supplying orthogonal views to the intravehicular activity (IVA) robotic operator, supply views of EVA operations to IVA and/or ground crews monitoring the EVA, and carry out independent visual inspections of areas of interest around the spacecraft. To enable these future benefits with minimal impact on IVA operators and ground controllers, the Mini AERCam system architecture incorporates intelligent systems attributes that support various autonomous capabilities. 1) A robust command sequencer enables task-level command scripting. Command scripting is employed for operations such as automatic inspection scans over a region of interest, and operator-hands-off automated docking. 2) A system manager built on the same expert-system software as the command sequencer provides detection and smart-response capability for potential system-level anomalies, like loss of communications between the Free Flyer and control station. 3) An AERCam dynamics manager provides nominal and off-nominal management of guidance, navigation, and control (GN&C) functions. It is employed for safe trajectory monitoring, contingency maneuvering, and related roles. This paper will describe these architectural components of Mini AERCam autonomy, as well as the interaction of these elements with a human operator during supervised autonomous control.

---

[*] Branch Chief, Intelligent Systems Branch, Automation, Robotics and Simulation Division, NASA JSC ER2, AIAA Senior Member

[†] Computer Engineer, Special Projects Office, Automation, Robotics and Simulation Division, NASA JSC ER6

[‡] Ascent Analyst, Flight Design and Dynamics Division, NASA JSC DM4, AIAA Member

[§] Flight Control Engineer, Aeroscience and Flight Mechanics Division, NASA JSC EG6

[**] Crew Exploration Vehicle Deputy for Flight Dynamics Functional Area Lead, Aeroscience and Flight Mechanics Division, NASA JSC EG6

I. Introduction

The Johnson Space Center Engineering Directorate has developed a free-flying, robotic inspection vehicle called the Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam). At 7.5 inches in diameter and approximately 10 pounds in weight, the Mini AERCam technology demonstration unit is a nanosatellite-class spacecraft that is designed for external inspection and viewing for human or robotic spacecraft. The Mini AERCam Free Flyer can be operated via remote piloting or as a supervised autonomous system, with functions such as automatic stationkeeping, point-to-point maneuvering, and waypoint tracking. The Mini AERCam Free Flyer is accompanied by a sophisticated control station for command and control, as well as a docking system for automated deployment, docking, and recharge at a parent spacecraft.

Mini AERCam aims to provide beneficial on-orbit views that cannot be obtained from fixed cameras, cameras on robotic manipulators, or cameras carried by crewmembers during extravehicular activities (EVA's). On Shuttle or International Space Station (ISS), for example, Mini AERCam could support external robotic operations by supplying orthogonal views to the intravehicular activity (IVA) robotic operator, supply views of EVA operations to IVA and/or ground crews monitoring the EVA, and carry out independent visual inspections of areas of interest around the spacecraft.

II. Mini AERCam Overview

A. Mini AERCam System Description

The nanosatellite-class spherical Mini AERCam[1,2,3] Free Flyer incorporates significant additional capabilities compared to the 35-pound, 14-inch AERCam Sprint Free Flyer that flew as a remotely piloted Shuttle flight experiment in 1997 (Figure 1). Mini AERCam, shown in Figure 2, hosts a full suite of miniaturized avionics, instrumentation, digital imagers, communications, navigation, video, power, and propulsion subsystems. Technology innovations include a rechargeable xenon gas propulsion system, rechargeable lithium ion battery, custom avionics based on the PowerPC 740/750 microprocessor, "camera-on-a-chip" imagers with wavelet video compression, micro electromechanical system (MEMS) gyros, Global Positioning System (GPS) relative navigation, digital radio frequency communications, micropatch antennas, digital instrumentation network, and compact mechanical packaging. An expert-system based command script capability enables hands-off execution of complex Free Flyer operations, including automated inspection scans.

The Mini AERCam docking system[4] consists of sensor and mechanical components: The Free Flyer video system uses a target based system to provide docking-approach navigation for maneuvering the Free Flyer into close proximity with a magnetic docking mechanism. Then the magnetic docking system performs the final alignment and capture of the Free Flyer, culminating in a precision hard-dock suitable for connecting propulsion and electrical recharge elements.

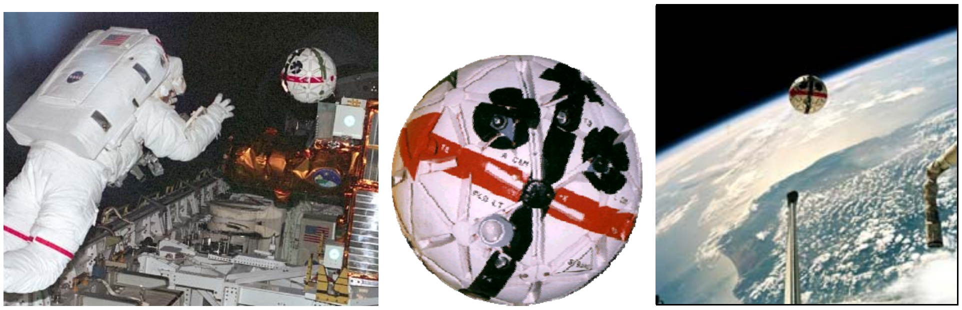


Figure 1: AERCam Sprint on STS-87

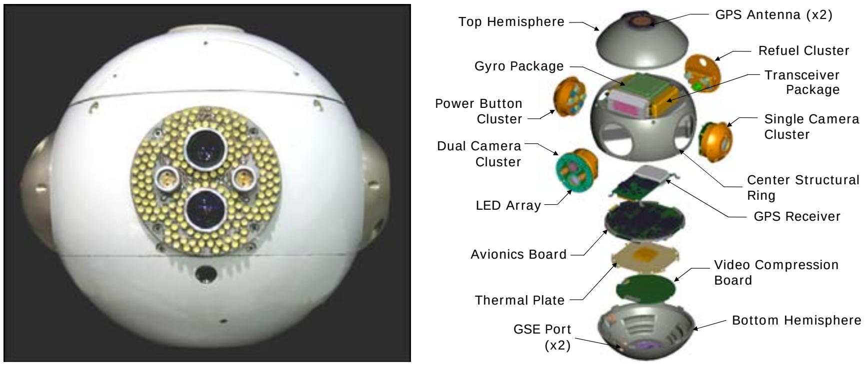


Figure 2: Mini AERCam Free Flyer

B. Mini AERCam System Testing

Free Flyer functional testing has been conducted on an airbearing table and in a six-degree-of-freedom closed-loop orbital simulation. In the airbearing table environment (Figure 3), the Free Flyer hardware is placed in the cradle of a tetherless airbearing sled, which produces a nearly frictionless environment. The Free Flyer fires its thrusters to maneuver along the airbearing surface while it transmits video and telemetry to the pilot control and display station. The orbital simulation models the three-dimensional dynamics of the Free Flyer in proximity to a parent vehicle, such as the Shuttle or ISS, and produces corresponding God's-eye views and simulated Free Flyer camera views. A high-fidelity simulation is achieved by using Mini AERCam avionics and by directly interfacing to Free Flyer thruster driver signals, emulating the MEMS gyro responses in hardware, and using the "truth" state to drive a GPS signal generator connected to the Free Flyer GPS receiver.

In addition to functional testing, the Mini AERCam technology demonstrator has been subjected to space-environment analyses and testing to ensure it is a suitable basis for a flight system design. Environmental testing with the technology demonstrator has included thermal vacuum, radiation, communications link margin, and solar lighting.

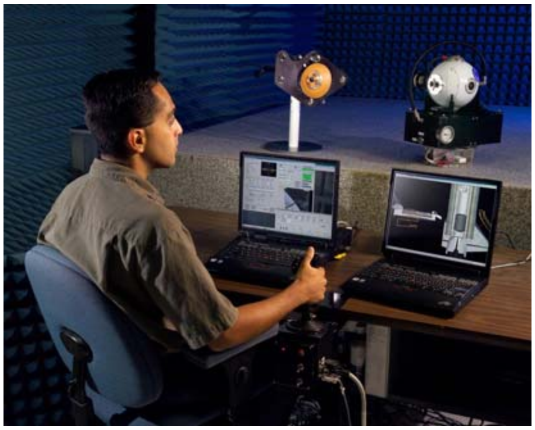


Figure 3: Mini AERCam Airbearing Table Testing

III. Mini AERCam Intelligent Software Architecture

The Mini AERCam system architecture incorporates intelligent systems attributes that support autonomous capabilities. 1) A robust command sequencer enables task-level command scripting. Command scripting is employed for operations such as automatic inspection scans over a region of interest, and operator-hands-off automated docking. 2) A system manager built on the same expert-system software as the command sequencer provides detection and smart-response capability for potential system-level anomalies like loss of communications between the Free Flyer and Control Station. 3) An AERCam dynamics manager provides nominal and off-nominal management of guidance, navigation, and control (GN&C)[5] functions. It is employed for safe trajectory monitoring, contingency maneuvering, and related roles. The following sections will describe these architectural components of Mini AERCam autonomy, as well as the interaction of these elements with a human operator during supervised autonomous control.

A. Command Sequencer

The Command Script Processor (CSP) is a robust command executor based on the C Language Integrated Production System (CLIPS[6]) that automates Free Flyer trajectory waypoint guidance commands. The command sequencing and execution is based on rules that model the way that an operator manually executes a paper procedure. Figure 4 illustrates the parallel between operator functions and CSP functions: Both agents monitor system feedback through telemetry, follow a procedure, decide which command to issue next based on the current system state (reflected in telemetry), then issue the commands.

The CSP resides both on the Control Station and on board the Free Flyer. On the Control Station it allows the operator is develop, load, and run command scripts that automate Free Flyer tasks in a closed-loop fashion, as if the operator is directly controlling the Free Flyer. From the Free Flyer's perspective it does not know or care if a command is issued directly from the Control Station graphical user interface (GUI) by the operator or from the CSP.

The sequencing and execution rules implement the following logic:

1. Monitor specified telemetry (system state) to determine readiness to issue a command to the system
    - verify good communications between Free Flyer and Control Station
    - verify valid Free Flyer navigation state before executing motion control commands
    - verify successful completion of the previous command
2. If all of the checks are true for a given command, then the command is issued

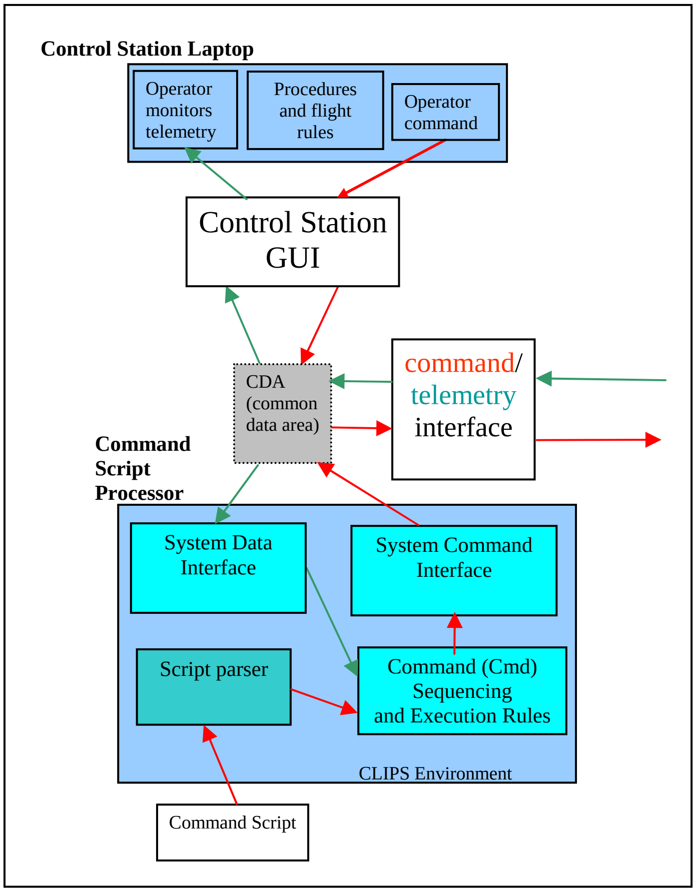
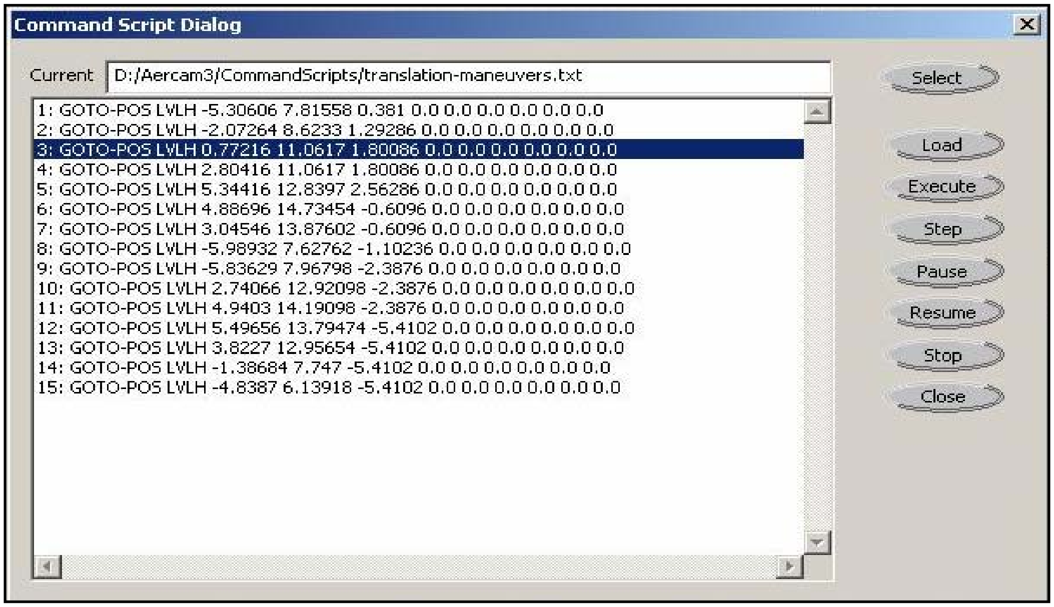


Figure 4: Mini AERCam Command Scripting Architecture

The user interface is designed for supervised autonomy where the operator can monitor autonomous operation and quickly take over control at any point. Also, if monitored checks fail during autonomous execution (i.e. communication between Control Station and Free Flyer is lost or a command fails to execute onboard), command script processing stops, an operator alert is issued, and control is handed back over to the operator. If the communications link is severed, then control reverts to the CSP on board the Free Flyer which follows a pre-defined set of contingency scripts executed by the System Manager, as described in the next section.

As shown in Figure 5, the user interface allows the operator to load, execute, pause, and/or resume a script, which is stored as a text file. Scripts can be created for configuring the Free Flyer, performing automated scanning operations, performing automated docking (which involves both maneuvering the Free Flyer as well as reconfiguring onboard navigation sensors), and virtually any sequence of commands that the operator might issue regularly. Script entries consist of a command keyword set followed by a set of numerical indicators. The operator can also manually select the starting point in the script. The script shown in Figure 5 consists of a set of waypoints.

The simple hand over capability between autonomous operation and manual operation is particularly useful during a scanning mission. The operator can stop the autonomous scan, take over manual control for a detailed inspection at a point of interest, then resume the scan where it left off.



Current D:/Aercam3/CommandScripts/translation-maneuvers.txt

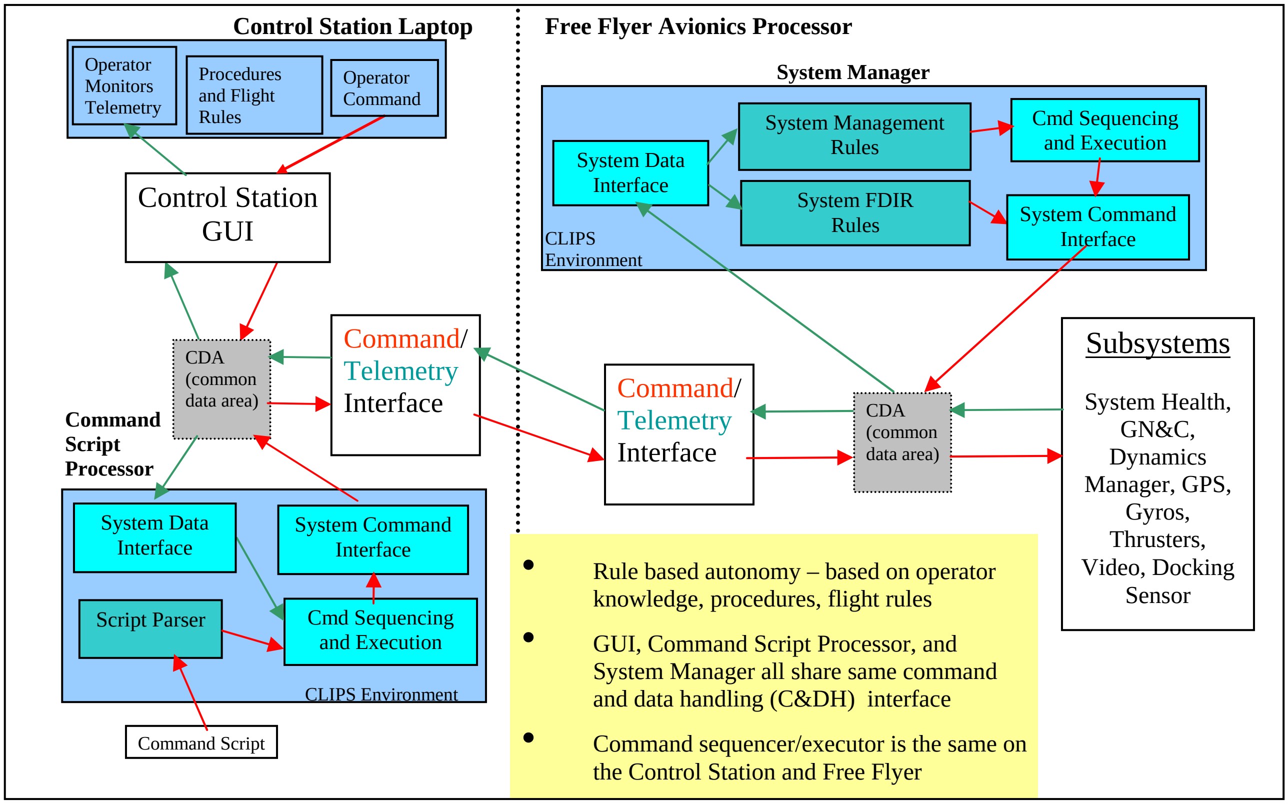
```
1: GOTO-POS LVLH -5.30606 7.81558 0.381 0.0 0.0 0.0 0.0 0.0 0.0
2: GOTO-POS LVLH -2.07264 8.6233 1.29286 0.0 0.0 0.0 0.0 0.0 0.0
3: GOTO-POS LVLH 0.77216 11.0617 1.80086 0.0 0.0 0.0 0.0 0.0 0.0
4: GOTO-POS LVLH 2.80416 11.0617 1.80086 0.0 0.0 0.0 0.0 0.0 0.0
5: GOTO-POS LVLH 5.34416 12.8397 2.56286 0.0 0.0 0.0 0.0 0.0 0.0
6: GOTO-POS LVLH 4.88696 14.73454 -0.6096 0.0 0.0 0.0 0.0 0.0 0.0
7: GOTO-POS LVLH 3.04546 13.87602 -0.6096 0.0 0.0 0.0 0.0 0.0 0.0
8: GOTO-POS LVLH -5.98932 7.62762 -1.10236 0.0 0.0 0.0 0.0 0.0 0.0
9: GOTO-POS LVLH -5.83629 7.96798 -2.3876 0.0 0.0 0.0 0.0 0.0 0.0
10: GOTO-POS LVLH 2.74066 12.92098 -2.3876 0.0 0.0 0.0 0.0 0.0 0.0
11: GOTO-POS LVLH 4.9403 14.19098 -2.3876 0.0 0.0 0.0 0.0 0.0 0.0
12: GOTO-POS LVLH 5.49656 13.79474 -5.4102 0.0 0.0 0.0 0.0 0.0 0.0
13: GOTO-POS LVLH 3.8227 12.95654 -5.4102 0.0 0.0 0.0 0.0 0.0 0.0
14: GOTO-POS LVLH -1.38684 7.747 -5.4102 0.0 0.0 0.0 0.0 0.0 0.0
15: GOTO-POS LVLH -4.8387 6.13918 -5.4102 0.0 0.0 0.0 0.0 0.0 0.0
```

Figure 5: Command Script Processor User Interface

B. System Manager

The primary components of the System Manager are Fault Detection Isolation and Recovery (FDIR) and resource management. The System Manager employs a copy of the same CSP that runs on the Control Station for handling the execution of onboard closed-loop recovery scripts as well as resource management scripts.

The System Manager is built from a rule-based, expert system in which the rules “look for” predefined data signatures to detect when an event of interest (possible failure, anomaly, or safety condition violation) has occurred, then automatically loads and executes a recovery script. An important consideration for a teleoperated Free Flyer is what to do if a loss of communications occurs between the Control Station and the Free Flyer. The specific action to take depends on several factors, corresponding to specific data signatures. Another common consideration involves close-in proximity operations for which several rules are implemented to protect the Free Flyer and parent vehicle. Figure 6 highlights the end-to-end control architecture. This figure depicts the parallels between operator control and CSP control on the Control Station. It also depicts the parallels between Control Station control and onboard control using the System Manager.



- Rule based autonomy – based on operator knowledge, procedures, flight rules
- GUI, Command Script Processor, and System Manager all share same command and data handling (C&DH) interface
- Command sequencer/executor is the same on the Control Station and Free Flyer

Figure 6: End to End Control Architecture

C. Dynamics Manager

The AERCam Dynamics Manager (ADM) software is a set of logic[7] tied in to the System Manager and tightly coupled with the core GN&C software that commands Free Flyer trajectory maneuvers in response to system failures. The ADM manages Free Flyer maneuvering primarily during periods of communication loss, but also in situations where automatic safe maneuvering is required (i.e. when minimum standoff distance or speed limit violations occur). All of the ADM commands are aimed at either recovering the mission or safing the Free Flyer in the event of a hardware or software failure. The System Manager has responsibility for choosing the appropriate ADM command based on the current vehicle state. Command requests are then sent through the Common Data Area (CDA) to the ADM**,** which calculates the appropriate parameters, including trajectory targets, and performs direct commanding of the GN&C system. Figure 7 shows the commanding relationship between these different components for different situations.

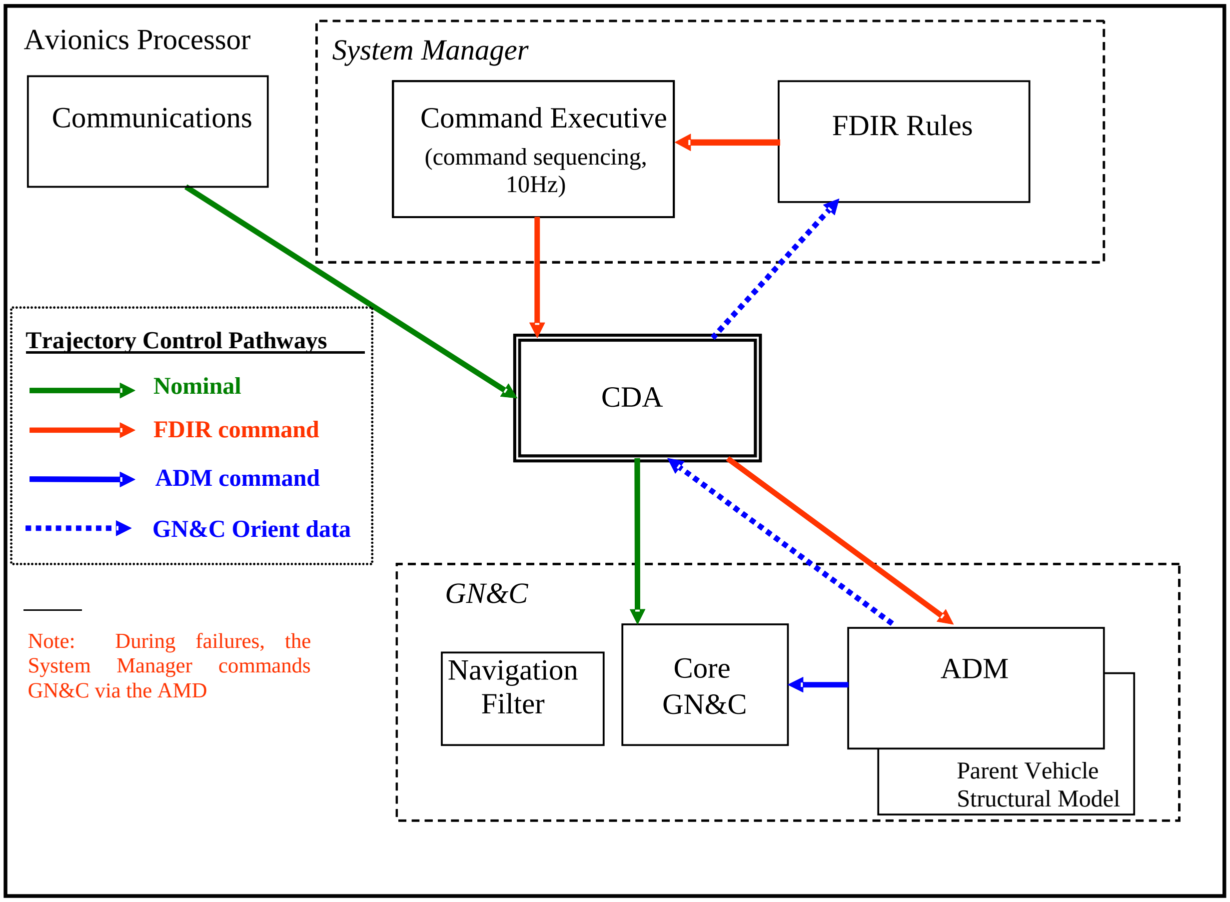


Figure 7: ADM/System Manager Commanding Interface

The commands that may be passed from the System Manager to the ADM are:

CONTINUE: Free Flyer continues executing the current path/plan.

REACQ_COMM: Free Flyer backtracks through a user-defined number of breadcrumbs (position and attitude markers), pausing for a user-defined number of seconds at each to reacquire communications.

RETURN: Free Flyer follows the breadcrumb history, without pausing, until reaching the specially marked “Point of Interest” breadcrumb.

SAFE: Free Flyer is commanded to the closest predetermined location relative to the parent vehicle where the Free Flyer can be safe for an extended period of time without chance of recontact with the parent vehicle.

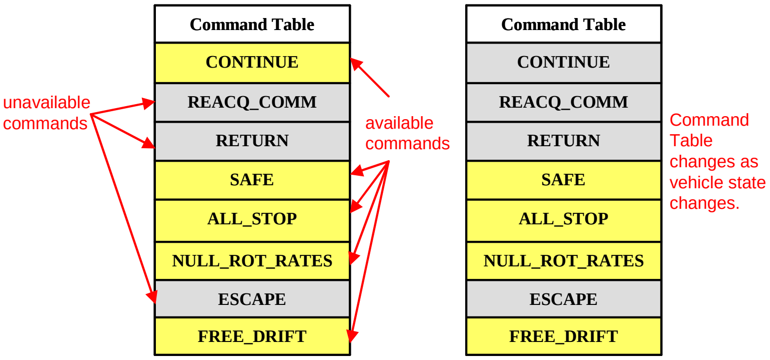
ALL_STOP: Null all Free Flyer rates (translation and rotation), and hold the current relative position and attitude.

NULL_ROT_RATES: Null all Free Flyer rotation rates and hold the current relative attitude.

ESCAPE: Free Flyer is commanded on an escape trajectory that assures no return contact with the parent vehicle for multiple orbits.

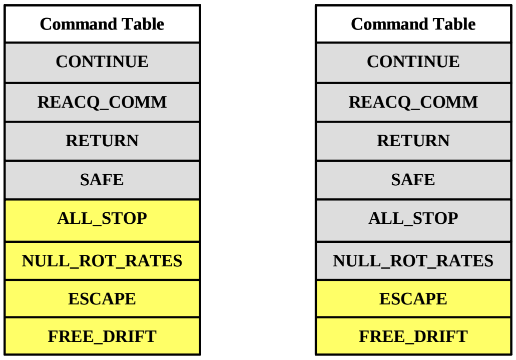
FREE_DRIFT: Inhibit all Free Flyer jet firings.

Each command has a list of flags and necessary conditions associated with it which must be satisfied for that command to execute. For example, communication must be lost for a REACQ_COMM command to make sense, and the Free Flyer must have good attitude state information to perform any type of translation maneuver. The System Manager is constantly monitoring system state information and will issue to the ADM only the command available for a given system state. The ADM commands have a natural priority order from most likely to save the mission to least likely to recover the Free Flyer. This is the same order that the commands were listed above and is subject to the specifics of a particular mission. Using the combination of command availability based on vehicle state and priority order, the System Manager can choose the best ADM command in any situation by selecting the highest priority command that is available at any given time. This logic is most easily understood by an example.



a) ADM Command Availability Table, Nominal Operations.

b) Hand Controller failure prompts a state change. SAFE becomes available because the FF is in automatic mode and inside the standoff distance.



c) The FF reaching the nearest safe location prompts another state change, and the FF holds position and attitude.

d) Once propellant and/or power reach critical levels, the FF is commanded to ESCAPE.

Figure 8: ADM Command Sequencing for a Hypothetical Failure Scenario

Consider the following situation where the Free Flyer is conducting an automatic inspection scan of a section of the parent vehicle. Based on vehicle state information, the available ADM commands are shown in Figure 8a. The System Manager chooses the highest priority available command and issues the CONTINUE command to ADM. As long as there are no failures or vehicle state changes precluding normal operations, the ADM will remain in this state indefinitely. Now, suppose the crew takes over and manually maneuvers the Free Flyer inside of the nominal standoff distance to take a better look at some feature. The crew is allowed to do this because once they take manual control of the Free Flyer, its automatic flight control functions are disabled, and constraints, such as acceptable standoff distance, can be overridden. Now assume that just the Hand Controller (HC) heartbeat is lost by the Free Flyer. Note that this is not a total loss of communication, since Control Station and Relative Navigation data flows to the Free Flyer are still active. With the loss of crew input, the System Manager will detect the loss of hand controller input and take action. Based on the Free Flyer state change (loss of HC heartbeat and current position inside of the standoff distance) and the ADM command availability rules, the System Manager would issue the SAFE command. Once the Free Flyer has reached the nearest safe location, the SAFE option is no longer valid (Figure 8c), and the System Manager will issue the ALL_STOP command. Relative navigation should still be acceptable, so the Free Flyer will maintain its relative position and attitude with respect to the parent vehicle while the crew works malfunction procedures (e.g. enabling alternative commanding from the Control Station). If the recovery procedures do not work and Control Station commanding is unavailable, the System Manager will command an ESCAPE maneuver once power or propellant levels reach a critical threshold, meaning the Free Flyer is just able to complete the escape maneuver (Figure 8d).

The ADM uses “waypoints” and “breadcrumbs” as steering targets. Waypoints are predefined position and attitude values, relative to the parent vehicle, that are used to perform routine functions such as scanning the exterior of the parent spacecraft, or traveling from the Free Flyer hangar to a region of interest around the parent vehicle.

Between waypoints, or when the Free Flyer is in manual mode, breadcrumbs are defined at user-defined distance intervals that allow the Free Flyer to retrace its steps. Breadcrumbs are a record of position, attitude, and creation time. The time stamp is recorded to ensure that breadcrumbs are not “stale” when being used to retrace the Free Flyer’s path around the parent vehicle. A breadcrumb can be marked as a “Point of Interest” by the crew. Once a waypoint is marked as a point of interest, the crew is able to use the RETURN command to automatically revisit that location.

IV. Discussion and Conclusion

The autonomy architecture developed for Mini AERCam was influenced by several practical goals for embedded flight software design. All components of the system described in the previous section are needed to operate within the available memory and processor utilization constraints of an embedded single-processor running the rest of the flight software under VxWorks[8] real-time operating system. In addition to run-time performance considerations, the project strongly desired an autonomy architecture and implementation that would facilitate verification for software quality assurance. For example, the CSP employs a simple script format (not a complex language) coupled with extensive script parsing to ensure script correctness prior to loading and execution.

The software development team implemented a rule-based expert system (using CLIPS) for conditional sequencing and system management because of advantages in representation, flexibility and ease of maintenance compared with traditional hand coded methods. Representation and flexibility are especially important since FDIR details (what and how to detect, and how to respond) are operations not fully defined until the system has been designed and its failure modes are understood. Also, as the system undergoes testing and failure modes are better understood and additional failure modes are often discovered, the ability to modify or add detection rules and recovery scripts with a simple but powerful representation avoids more costly and error-prone changes to low level software late in the development cycle.

The main issues with pursuing the expert system approach involved understanding the scalability, response time, and determinism of the system. Response time was considered with respect to the task to be performed (monitored data update rates, critically and consequences). The Free Flyer produces data at 25Hz, 10Hz, and 1Hz. System data (including navigation data) is published at 10Hz and 1Hz, with GN&C control data at 25Hz. The Free Flyer typically maneuvers at only a couple inches per second and operates nominally (except for undock/docking) 15ft or more from the parent vehicle. With 1Hz navigation updates and nominal translational rates, response times of even a few seconds would be acceptable, because the Free Flyer can only move inches before an action is effected. Through lab tests on prototype hardware, it was shown that the implementation chosen produced response times of 10s of milliseconds. Since this is less than the fastest monitored data update rate of 10Hz (100ms frame), it could be considered real-time for this application. Any variability (or lack of determinism) within the update interval for the data monitored could be tolerated because the system would not detect the difference.

The inclusion of the ADM allows for FDIR responses and related actions that involve only GN&C elements to be commanded directly by the ADM without going through the command data area and command executive used by the system manager. The primary run-time advantage of this second tier of task-level control is better scalability for autonomous motion control functions. The primary system advantage is greater encapsulation of the GN&C subsystem for verification and validation.

References

---

1 Mini AERCam website, http://aercam.nasa.gov, Accessed August 1, 2006

2 S. Fredrickson, L Abbott, S. Duran, D. Jochim, W. Studak, J. Wagenknecht, N. Williams, “Mini AERCam: Development of a Free-Flying Nanosatellite Inspection Robot,” Proceedings of SPIE Vol. 5088, p. 97-111, Space Systems Technology and Operations; Peter Tchoryk, Jr., James Shoemaker; Eds. Aug 2003.

3 S. Fredrickson, S. Duran, and J. Mitchell, “Mini AERCam Inspection Robot for Human Space Missions,” Proceedings of AIAA Space 2004, San Diego, California, September 28-30, 2004.

---

4 S. Fredrickson, S. Duran, N. Howard, and J. Wagenknecht, “Application of the Mini AERCam Free Flyer for Orbital Inspection,” Proceedings of SPIE Vol. 5419, Spacecraft Platforms and Infrastructure, Peter Tchoryk, Jr., Melissa Wright, Eds., April 2004.

5 J. Wagenknecht, S. Fredrickson, T. Manning, and B. Jones, “Design, Development and Testing of the Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam) Guidance, Navigation, and Control Systems,” 26th Annual American Astronautical Society Guidance and Control, 5-9 February 2003

6 CLIPS website, http://www.ghg.net/clips/CLIPS.html, Accessed August 1, 2006

7 R. Proud, J. Hart, "FLOAAT, A Tool for Determining Levels of Autonomy and Automation, Applied to Human-Rated Space Systems", Proceedings of Infotech @ Aerospace, Arlington, Virginia, September 26-29, 2005.

8 Wind River Systems VxWorks website, http://www.windriver.com, Accessed August 1, 2006