

Oracle7-DeepSpace: Self-Maintaining, Evidence-Gated Fault Response for Deep-Space Spacecraft on a Fraction of a CPU

Prometheus7 Research Institute

September 2026

Abstract

When onboard fault protection cannot resolve an anomaly, a deep-space spacecraft falls back to safe mode and waits for the ground, at a cost in science and operations time that grows with light time. We present Oracle7-DeepSpace, a compact autonomy core that sits between pre-scripted fault responses and safe mode. It converts telemetry into confirmed facts, facts into fault hypotheses, and hypotheses into admissible responses whose preconditions are checked; it chooses among them by simulating each on an onboard digital twin built from telemetry alone and taking the best worst case; it records a receipt for every decision; it learns fast-path rules only from episodes it has verified itself, and admits them only through a gate on a reserved validation suite; and it answers operator questions with citations to its own records, refusing questions no record supports. It is about 1,100 lines of standard-library Python, with no learned weights and no GPU. In simulation of four mission profiles modelled on Voyager 1, New Horizons, Juno and Europa Clipper, on 1,700 fresh held-out single-fault scenarios, it chose the correct response in 98.9% of cases (95% CI 98.3–99.3) with no harmful action and no damage, against 29.4% for a simplified limit-to-safe-mode baseline, which caused 218 damage events. From its own verified missions it admitted nine rules that reproduce an expert-authored fault table (100%) and rejected four of four planted harmful rules. It remedied both faults in 99.2% of fresh two-fault cases. On simulator physics its twin does not model, its outcomes matched the expert table (97.7% vs. 98.0% unharmed), and under the most extreme out-of-range faults it caused fewer damage events than the table (72 vs. 100). On three spacecraft profiles added after the code was frozen it was correct in all 1,300 scenarios; a fourth, in the hot environment near the Sun, exposed a cold-environment assumption in the facts layer that the expert table shares. Of five fault types it has no hypothesis for, it handled three without harm and was defeated by a frozen sensor and a biased current sensor. Replays are bit-identical, and 60 simulated days per spacecraft show flat compute. It runs in 0.2–0.4 ms per 60-second control tick with about 106 KiB of working memory. On real NASA SMAP and MSL telemetry its simple detection front end is not competitive with a published LSTM detector (event-level $F_{0.5}$ 0.42 vs. 0.71); the decision, self-maintenance and explanation layers are complementary to learned detectors. Every reported number is tied to a SHA-256 of the code and to machine-readable result files.

1 Introduction

A spacecraft far from Earth cannot wait for instructions. One-way light time is minutes at Mars, tens of minutes at Jupiter and nearly a day to Voyager 1; in 2017 the result of a thruster test on Voyager 1 took 19 hours and 35 minutes to reach the ground [6]. Onboard fault protection is therefore essential, and by design it is conservative. For anticipated faults, engineers write monitors and fault-specific responses before launch; when those cannot resolve a condition, the last line of

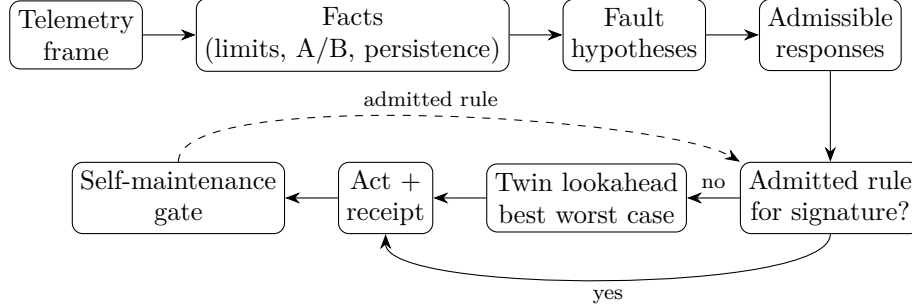


Figure 1: The decision loop of Oracle7-DeepSpace. Each 60-second tick turns telemetry into confirmed facts and hypotheses; a response is chosen by an admitted rule or by worst-case lookahead on a telemetry-built twin, and every decision leaves a receipt. The gate admits new rules only from self-verified episodes.

defence is *safe mode*: stop science, point to the Sun, stay warm and powered, and wait. Safe mode is a pause, not a fix. Recovery requires diagnosis on the ground, a tested command sequence and an uplink, which often costs days of science, and more the further out the spacecraft is.

Onboard autonomy has flown before: Remote Agent planned, executed and recovered from faults on Deep Space 1 [1]; the Autonomous Sciencecraft replanned observations on EO-1 [2]; AEGIS selects science targets on Mars rovers [3]. Still rare in flight is autonomy that *improves itself* under a discipline operators can audit, explains each decision in plain language, and does so within the compute budget of radiation-hardened processors.

This paper contributes:

1. A decision core that selects fault responses by worst-case simulation over a telemetry-built digital twin, with checked preconditions, explicit evidence collection on transients, and a receipt per decision (Section 2).
2. A self-maintenance gate that proposes rules only from self-verified episodes and admits them only if a reserved validation suite loses no correct response, adds no harmful one and costs less compute, with versioning, rollback and ground veto.
3. A self-report layer that answers operator questions from telemetry frames and receipts, with citations, and refuses unsupported questions.
4. An evaluation across four mission profiles (Section 3) with fixed seed splits, confidence intervals, an expert-authored fault table as an upper bound, compound faults, ablations, physics the twin does not model, faults outside the design range, and a test of the detection front end on real SMAP/MSL telemetry [4]; spacecraft added after the code was frozen, faults it has no hypothesis for, deterministic replay and a 60-day endurance run; and a harness with which an independent evaluator can fly scenarios of their own against the frozen code.

2 System

Facts. Each tick, telemetry is reduced to facts: operating-band violations, disagreement between redundant A/B temperature sensors, heating or cooling against the commanded heater state, reaction-wheel overcurrent, thruster over-use (firing duty), loss of star-tracker attitude, power-source shortfall against the spacecraft’s own design value, and bus load in excess of every known consumer.

A fact is *confirmed* only after it persists for P ticks ($P = 3$); a single-tick anomaly yields only `collect_evidence`.

Hypotheses and admissible responses. Each confirmed fact maps to one or more fault hypotheses (e.g. a power shortfall to solar degradation *or* a short circuit). Each hypothesis admits responses whose preconditions are checked against echoed state: switch a heater to its backup circuit only if it is still on primary, switch to the spare wheel only if it is unused, mark a sensor invalid only if its partner is valid, and so on.

Digital-twin lookahead. For every candidate response and every hypothesis still consistent with the facts, a twin is built *from telemetry only* (estimated temperatures, currents and configuration) with thermal and source constants perturbed by $\pm 10\%$, the hypothesised fault injected, and the twin calibrated to the observed source and load currents before predicting. It is simulated for 90 minutes and scored on band and survival violations, pointing error, battery depletion, the horizon-averaged power deficit, and a latent-hazard term for a confirmed fault left unremedied (weighted as survival-class for an unisolated electrical short). The response with the lowest worst-case score is chosen. A decision to wait expires after 30 minutes, so the fact is re-examined as the context changes.

Self-maintenance gate. From its own lookahead receipts, the spacecraft proposes a rule “fact signature $\rightarrow$ response” when that response resolved the signature at least three times (the facts cleared within 30 minutes with no damage) and never failed. A rule is admitted only if flying the reserved validation suite (three seeds per spacecraft per fault class) with the rule loses no correct response, adds no harmful response and uses fewer twin simulations. Every admission creates a new rulebook version; the ground can veto a rule by its identifier.

Self-report. Operator questions are classified (status, anomalies, why an action was taken, what was learned, confidence, a measured value) and answered only from telemetry frames and receipts, with citations such as `[receipt #0]`; questions without a supporting record are refused.

3 Evaluation

3.1 Simulator and mission profiles

A deterministic simulator (one tick = 60 s) models power (an RTG with a small capacitor buffer and no battery, or a solar array with a battery), thermal state (three zones, each with primary and backup heaters and redundant sensors), attitude (reaction wheels, or thrusters on A/B branches), and star-tracker/sun-sensor knowledge. Four profiles set the power source, actuators, thermal environment and power budget in the manner of Voyager 1 and New Horizons (RTG, thrusters), Juno (solar, thrusters) and Europa Clipper (solar, reaction wheels). The profiles select which subsystems and faults exist; they are not engineering-fidelity models of the real vehicles. Faults: heater stuck on, heater stuck off, biased sensor, star-tracker dropout, reaction-wheel friction, thruster-branch degradation (as on Voyager 1, which switched to its backup thrusters in 2017 [6]), solar degradation, RTG decline, battery short, and a single-tick radiation upset. A biased sensor, a radiation upset and a nominal run are *traps*: any physical response is wrong.

System	Correct (95% CI)	Harmful	Safe-mode	Damage
Oracle7-DeepSpace + self-learned rules	100.0% (99.8–100)	0	100	0
Oracle7-DeepSpace	98.9% (98.3–99.3)	0	100	0
Expert fault table (upper bound)	100.0% (99.8–100)	0	100	0
Random admissible response	43.5% (41.2–45.9)	117	403	133
Limit $\rightarrow$ safe mode (simplified)	29.4% (27.3–31.6)	200	1300	218
No action	23.5% (21.6–25.6)	0	0	163

Table 1: Fresh held-out single faults: 1,700 scenarios (4 spacecraft $\times$ their fault classes $\times$ 50 seeds). The safe-mode counts of the correct systems are battery shorts, for which safe mode is the designed response.

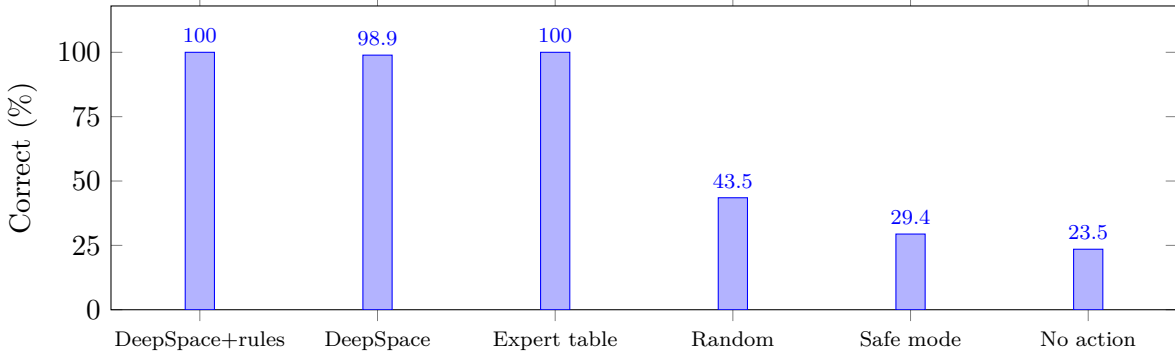


Figure 2: Correct first response on 1,700 fresh held-out scenarios.

3.2 Protocol

Seed ranges were fixed in code: development 0–7; gate validation suite 200–202; a first held-out set 1000–1049; compound faults 3000–3019; and, after two post-hoc fixes (Section 3.13), a *fresh* held-out set 5000–5049 and a fresh compound set 6000–6019, each run once. A response is correct if the first physical action is the designed remedy on the right target or, for traps, if no physical action is taken. Baselines: (i) an *expert fault table* mapping each fact to its designed response, written with full knowledge of the fault classes and using the same facts module, which makes it an upper bound for anticipated faults; (ii) a *simplified limit-to-safe-mode baseline*, which enters safe mode on any confirmed fact and has no fault-specific responses; (iii) a random admissible response; (iv) no action. Baseline (ii) is not a model of real flight fault protection, which combines fault-specific responses (closer to (i)) with safe mode as the fallback; it measures what safe mode alone achieves on these faults. Rates carry Wilson 95% intervals.

3.3 Results: single faults

Table 1 and Figure 2 give the main result. Without any fault table, Oracle7-DeepSpace comes within about one percentage point of an expert table authored with full knowledge of the fault classes, with no harmful action and no damage in 1,700 scenarios. All of its misses are one case: a small solar deficit on the Juno profile that the twin cannot resolve over its horizon. The simplified safe-mode baseline fails 70.6% of the time, enters safe mode 1,300 times and records 218 damage events, because safe mode does not repair a stuck heater, a failed wheel or a degraded thruster branch. The median time from fault onset to the correct response was 3 minutes (90th percentile 93

Planted rule	Gate verdict
sensor disagreement $\rightarrow$ switch heater	REJECT: loses correct responses 101 $\rightarrow$ 89; adds harmful 0 $\rightarrow$ 12
thruster over-use $\rightarrow$ safe mode	REJECT: loses correct responses 101 $\rightarrow$ 92
load excess $\rightarrow$ shed loads	REJECT: loses correct responses 101 $\rightarrow$ 95; adds harmful 0 $\rightarrow$ 6
attitude lost $\rightarrow$ do nothing	REJECT: loses correct responses 101 $\rightarrow$ 89

Table 2: The gate rejects planted harmful rules on the reserved validation suite (4 spacecraft, 3 seeds per fault class, 102 missions per flight).

minutes, dominated by slowly developing thermal and power faults).

3.4 Results: self-maintenance

Starting from an empty rulebook, the spacecraft proposed rules from its development missions. The gate admitted nine, one per fact signature, each keeping validation-suite correctness equal or higher (101/102 rising to 102/102) and together reducing twin simulations from 406 to zero. The admitted rules coincide with the expert table, and with them the system scored 100% on the fresh set. Four deliberately harmful rules were rejected with explicit reasons (Table 2).

3.5 Results: compound faults

Twelve fault pairs (three per spacecraft), the second fault 30 minutes after the first, 20 seeds each. On the fresh set, Oracle7-DeepSpace remedied both faults in 238/240 cases (99.2%, CI 97.0–99.8) with no harmful action; the expert table in 240/240; and the simplified safe-mode baseline in 0/240, with 102 damage events.

3.6 Ablations

On fresh seeds 5000–5019 (680 scenarios), widening the twin’s constant mismatch from 0% to $\pm 30\%$ reduced correctness only from 99.3% to 98.1%, with no harmful action at any level. Persistence matters: acting on a single tick ($P=1$) gave 87.2% correct, with 80 harmful actions and a wrong action on 33% of trap scenarios, while $P \in \{2, 3, 5\}$ produced none (98.8–99.0% correct), at a latency cost that grows with P (median 2, 3 and 5 minutes).

3.7 Self-report

On 14 operator questions across the four spacecraft, including four that must be refused, the self-report layer answered 14/14 correctly. Three fakes scored: echoing the question 0, a canned “all systems nominal” 0, always refusing 4. An example, verbatim: “At $t=98$ I switched the battery heater to its backup circuit because the battery kept heating while its heater was commanded off. I chose it by simulating each option on my onboard model (worst-case scores, lower is better: switch heater backup scored 2.0, collect evidence scored 64.0, safe mode scored 104.0). [receipt #0]”

3.8 Compute

Per 60-second tick the full stack took 0.22–0.43ms uncapped, the worst single decision (with lookahead) 11–31 ms, and peak working memory was 106–109 KiB. An earlier build of the same design, run under a hard cap of 5% of one CPU core and 64 MB of memory, made the same decisions

Condition	Oracle7-DeepSpace		Expert table		Safe-mode baseline	
	strict	outcome	strict	outcome	strict	outcome
Base physics	99.0	100.0	100.0	100.0	29.4	68.2
Thermal mismatch	92.5	99.4	–	–	–	–
Power mismatch	99.4	99.9	–	–	–	–
Sensor mismatch	98.9	100.0	–	–	–	–
Attitude mismatch	99.0	100.0	–	–	–	–
All terms + dropout	93.3	97.7	94.7	98.0	29.2	72.3
Weak faults, base	83.1	100.0	100.0	100.0	14.3	71.4
Strong faults, base	85.7	100.0	100.0	100.0	14.3	71.4
Weak faults, all terms	93.4	98.7	99.7	98.7	14.3	71.4
Strong faults, all terms	81.7	88.7	100.0	84.7	14.3	71.4

Table 3: Physics the twin does not model (1,700 scenarios per row) and out-of-range fault strengths (700 per row), seeds 7000–7049, flight code unchanged. *Strict*: the designed response on the right target (%). *Outcome*: the spacecraft came to no harm (%).

as uncapped at about 1 ms per tick, with a worst decision of about 100 ms and 16 MB resident; even that worst case is under 0.2% of the tick budget.

3.9 Physics the twin does not model, and faults outside the design range

The twin shares the simulator’s equations, so the mismatch ablation above varies only constants. To test structure, the spacecraft was flown on simulator physics the twin does not model, with the flight code frozen (same SHA-256) and on seeds 7000–7049, never used before. The added terms were: inter-zone conduction, a radiative (T^4) share of heat loss and heater power $\times 0.8$ – 1.25 per zone (*thermal*); 95% coulombic efficiency, an unmodelled load drifting within $\pm 4\%$ of the source, and current noise (*power*); 0.25°C quantisation and heavy-tailed outliers (*sensors*); $3\times$ pointing jitter (*attitude*); and all of these plus 2% telemetry-frame dropout (*all*). One test parameter, the load drift, was resized on development seeds before the run, because $\pm 0.6\text{ A}$ exceeded an RTG spacecraft’s entire power margin and browned out healthy spacecraft. Separately, six fault classes were injected *weak* (just above the detection thresholds) and *strong* (far beyond the design range). A designed response is not necessarily right outside the design range, so every run is also judged by its *outcome*: no damage, no harmful action, battery above 30%, and no zone out of band for more than 10 minutes. The outcome measure was added after the first run; the runs are deterministic, and the strict numbers reproduced exactly (Table 3).

Three findings follow. First, the twin is not brittle to physics it does not model: with every term active, Oracle7-DeepSpace and the expert table lose about the same (strict 93.3% vs. 94.7%, outcome 97.7% vs. 98.0%), and each records six harmful actions, which locates the loss in the facts layer both share rather than in the twin. The dominant case is diagnostic: a healthy but under-powered heater, or a stuck-on heater whose extra heat is absorbed by the additional losses, can be indistinguishable in telemetry from a failed one; 124 of the 128 thermal-mismatch misses left the spacecraft unharmed. Second, outside the design range the strict score understates the system: every strict miss on weak or strong faults under base physics was harmless, because the twin declined to shed science for deficits it predicted the spacecraft could carry (science was off 58 minutes on average, against 110 for the table), and chose safe mode rather than load shedding for a 55% array loss. Third, at the extreme, with strong faults under every mismatch term, no system avoided harm in every run; Oracle7-DeepSpace recorded 72 damage events against the table’s 100

Unknown fault (400 runs)	Oracle7-DeepSpace strict / outcome	Expert table strict / outcome	Safe-mode baseline strict / outcome
Heater weakens to 35%	66.8 / 100.0	66.8 / 100.0	0.0 / 83.3
Star tracker drops out 3 ticks in 10	100.0 / 100.0	100.0 / 100.0	0.0 / 0.0
Power source decays slowly	100.0 / 100.0	100.0 / 100.0	0.0 / 100.0
Temperature sensor freezes	0.0 / 79.3	0.0 / 79.3	0.0 / 33.3
Load-current sensor reads 3.2 A high	0.0 / 0.0	0.0 / 0.0	0.0 / 0.0

Table 4: Faults the flight code has no hypothesis for, on all eight spacecraft (%). *Strict*: the designed remedy where one exists, otherwise no harmful action. *Outcome*: no harmful action and no damage.

(outcome 88.7% vs. 84.7%).

3.10 A harness for independent scenarios

Every scenario above was written by the system’s developers. To make an independent test cheap, the script `run_scenarios.py` (in `eval/`) flies a JSON file of scenarios (spacecraft, physics variant, and faults with onset, target and optional strength outside the design range) against the frozen flight code and, optionally, both baselines. It reports, per fault, the first response, its correctness and latency, harmful actions and damage, stamped with the SHA-256 of both the scenario file and the flight code. An evaluator can therefore design a test without seeing the code, and it can be run without changing anything.

3.11 New spacecraft, unknown faults, determinism and endurance

Four further tests were written and run once on seeds 8000–8049, with the flight code frozen (same SHA-256).

Spacecraft added after the freeze. Four profiles were registered from outside the flight package: Voyager 2 (RTG, thrusters), OSIRIS-APEX and the Mars Reconnaissance Orbiter (solar, reaction wheels, different power budgets), and Parker Solar Probe (solar, reaction wheels, in a *hot* environment; every original profile runs cold). On the first three, Oracle7-DeepSpace was correct in all 1,300 scenarios (100%, CI 99.7–100) with no harmful action, and the rules the gate had admitted from the original four spacecraft transferred without loss; the simplified safe-mode baseline scored 25.0–33.3%. On Parker Solar Probe it was correct in 302 of 450 (67.1%), with 198 harmful actions and no damage, and the expert table scored exactly the same. The cause is in the shared facts layer: “a zone above its thermostat, warming while its heater is commanded off” is read as a stuck-on heater, which is true where the only heat source is the heater but false where the environment itself is hot.

Faults with no hypothesis. Five faults the code has no hypothesis for were injected on all eight spacecraft (400 runs each; Table 4). Three were handled without harm: a heater that weakens to 35% (the designed remedy in two thirds of runs, and every miss harmless), an intermittently dropping star tracker, and a slow decay of the power source. Two defeat it: a temperature sensor that freezes is never recognised, because no fact tests for a reading that stops moving, and the frozen value keeps corrupting thermal control (damage in 45 of 400 runs); and a load-current sensor that reads 3.2 A high is taken for a short circuit, sending a healthy spacecraft to safe mode every time, which is the same conservative failure as the baseline. The expert table behaves identically in every case, so both gaps lie in the facts layer.

Detector	SMAP $F_{0.5}$	MSL $F_{0.5}$	Combined $F_{0.5}$
EWMA + persistence, $P=1$	0.465	0.341	0.416
EWMA + persistence, $P=3$	0.534	0.089	0.229
EWMA + persistence, $P=5$	0.566	0.104	0.268
LSTM + nonparametric thresholds [4]	0.71	0.69	0.71

Table 5: Event-level results on real SMAP/MSL telemetry. The simple detector costs $0.21 \mu\text{s}$ per sample but is clearly weaker than the published LSTM.

Determinism. Every telemetry frame of 690 missions (all eight spacecraft) was recorded and replayed through fresh flight code: all 690 replays reproduced every receipt exactly. The same missions run in three separate processes with different hash seeds produced one identical digest of all receipts.

Endurance. Each of the eight spacecraft flew 60 simulated days continuously (86,400 ticks), with a fault every 36–60 hours until its redundancy was spent, a radiation upset every 6–10 hours, and Earth isolating a short and restoring the pre-safe-mode configuration 72 hours after any safe-mode entry. Of 53 faults, 52 were remedied, with no harmful action and no damage on any spacecraft; the one miss was a stuck-on heater during a safe-mode period on Juno. The median time per tick was flat from the first day to the last (about 0.2 ms), the worst tick was 27 ms, and the flight code’s memory grew only with its log of receipts (about 1 KiB per decision, at most 140 KiB after 60 days). The soak harness was corrected twice before the reported run (its crediting of actions to faults, and its model of Earth’s recovery); the flight code was not changed.

3.12 Real telemetry: SMAP and MSL

We tested the evidence layer (confirm by persistence; never act on a single sample) on NASA’s labelled SMAP (55 channels) and MSL Curiosity (27 channels) telemetry [4], verified against the published label file (82 channels, 105 anomaly sequences). The detector (an EWMA residual, a threshold from the *training* split only, and persistence P) was fixed before any test data or label was read, and all values of P are reported. Event-level results (Table 5) are not competitive with the published LSTM detector. Persistence raised precision on SMAP but hurt on MSL, where it fragmented long flagged stretches into many short false events. The simple front end built for the simulator is therefore inadequate for real telemetry; the decision, self-maintenance and explanation layers do not depend on it and can consume facts from any validated detector.

3.13 Post-hoc changes and threats to validity

Two changes were made after the first held-out and compound runs, both motivated by failure analysis: (i) a fact is marked handled only by an action that is one of its own admissible responses, so a second simultaneous fault is not dropped; and (ii) a decision to wait expires. The first held-out result (96.5%, CI 95.6–97.3) and the first compound result (92.5%) are therefore the pre-registered numbers; the fresh sets report the fixed system, run once. The pre-fix run’s result file, including its gate log (validation-suite baseline 97/102), is released alongside the others. Further threats: physics the twin does not model was tested only with the terms of Section 3.9; the facts layer assumes a cold spacecraft, cannot recognise a frozen sensor, and trusts current sensors (Section 3.11); a fact once handled is not re-armed if the ground later reverses the response; the fault classes, scenarios

and scoring were designed by the system’s developers; the expert table is an idealised upper bound; and no flight-like hardware, flight-software integration or independent test design has yet been used.

4 Related work

Remote Agent combined planning, execution and model-based mode identification and recovery on Deep Space 1 [1]. The Autonomous Sciencecraft on EO-1 detected events and replanned observations onboard [2]. AEGIS performs autonomous science targeting on Mars rovers [3]. LSTM-based detection with nonparametric thresholds set the reference result on SMAP/MSL telemetry [4]. F Prime, an open-source component-based framework for flight software [5], is a natural integration target. Oracle7-DeepSpace differs in combining worst-case twin selection with receipts, a self-maintenance gate that admits only rules that preserve correctness and reduce compute, explicit refusal, and a compute budget compatible with existing flight processors.

5 Conclusion and path to flight

In simulation, Oracle7-DeepSpace turns most of the faults that a safe-mode-only baseline answers with safe mode into correct, explained repairs; derives an expert-level fault table from its own verified experience without being given it; rejects harmful self-modifications; holds its outcomes on physics its twin does not model; and does so in milliseconds on a fraction of one CPU core, deterministically and without drift over 60 simulated days. Its detection front end is not yet adequate for real telemetry, and its facts layer has three identified gaps. Next steps: a facts layer that predicts passive heating from the environment, detects a sensor that stops moving, cross-checks current sensors against the power balance and can tell a weak actuator from a failed one; a hybrid with a validated learned detector; scenarios designed by an independent evaluator, for which the harness exists; evaluation on additional real telemetry and in a hardware-in-the-loop testbed; an F Prime component and a C port of the decision core; and graduated autonomy, in which rules are first proposed onboard and approved on the ground.

Reproducibility. Every number in this paper except the two disclosed pre-registered results comes from one code version, identified by the SHA-256 of the `deepspace` package sources:

b936ec011372fd9860c748b39b5117c7b8b1296f85d169ccf8a9b6f8e394408

Commands (Python 3, standard library only) and the result files they write:

- `eval/eval_fresh.py`: Table 1 and compound faults (`results/fresh_eval.json`);
- `eval/eval_final.py`: gate, ablations, self-report and compute (`results/final_supporting.json`);
- `eval/eval_structural.py`: Table 3 (`results/structural_eval.json`);
- `eval/eval_generalization.py`: Section 3.11 (`results/generalization_eval.json`);
- `eval/eval_smap_msl.py`: Table 5 (`results/smap_msl_eval.json`);
- `eval/run_scenarios.py` *file.json*: any independently written scenario set.

The result files and an interactive simulation running the same code are at <https://prometheus7.com/DeepSpace7/evidence>. Source code is available from the Prometheus7 Research Institute on request.

Use of AI tools. The system, the experiments and this manuscript were developed with substantial assistance from AI coding and writing tools, under the direction of the Prometheus7 Research Institute, which takes responsibility for all content.

References

- [1] N. Muscettola, P. P. Nayak, B. Pell, B. C. Williams. Remote Agent: to boldly go where no AI system has gone before. *Artificial Intelligence* 103(1–2):5–48, 1998.
- [2] S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, et al. Using autonomy flight software to improve science return on Earth Observing One. *Journal of Aerospace Computing, Information, and Communication* 2:196–216, 2005.
- [3] T. A. Estlin, B. J. Bornstein, D. M. Gaines, R. C. Anderson, D. R. Thompson, M. Burl, R. Castaño, M. Judd. AEGIS automated science targeting for the MER Opportunity rover. *ACM Transactions on Intelligent Systems and Technology* 3(3):50, 2012.
- [4] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. *Proc. KDD*, 2018. arXiv:1802.04431.
- [5] R. Bocchino, T. Canham, G. Watney, L. Reder, J. Levison. F Prime: an open-source framework for small-scale flight software systems. *AIAA/USU Conference on Small Satellites*, 2018.
- [6] NASA Science. Voyager 1 fires up thrusters after 37 years. News release, December 2017. <https://science.nasa.gov/missions/voyager-program/voyager-1/voyager-1-fires-up-thrusters-after-37-years/>