

Multi-Agent Reinforcement Learning for Balancing SDA Task Execution and On-Orbit Constellation Management

Ryan Herchig, Ph.D.

IERUS Technologies, Inc.

Ted E. Green

Lockheed Martin Corporation

John Burger, Mary Stewart

IERUS Technologies, Inc.

ABSTRACT

Operating a space-based sensing constellation requires balancing two competing imperatives: executing the observation taskings the mission exists to serve, and safekeeping the on-orbit assets execution depends on. With the rapid growth in launches, orbital debris, and on-orbit activity, the management of space-based sensing assets is becoming increasingly complex, compounding the difficulty of sustaining continuous tasking performance. Every maintenance period spent preserving a satellite's sensor is observation capacity taken from the tasking queue; every period of aggressive tasking degrades the fleet that must execute tomorrow's queue. These decisions are coupled across satellites and across time, with the task side itself being temporally structured: taskings arrive dynamically, carry priority grades and perishable collection windows, and include custody-class requirements paying off only when a multi-observation sequence is completed. Because the queue persistently exceeds fleet capacity, every planning period is an exercise in triage: which taskings to execute now, which to defer, and which to let expire. Current operational practice typically addresses scheduling with combinatorial optimizers that plan each period independently. This single-period optimization forfeits exactly the multi-period reasoning the balance requires: it cannot anticipate stochastic sensor degradation, adapt to dynamic tasking arrivals, coordinate fleet-level maintenance timing, or weigh present execution against future capacity.

Reinforcement Learning (RL) has shown promise for satellite scheduling, with single-agent approaches demonstrating fast policy execution suitable for on-board autonomy. However, applications to multi-satellite coordination that integrate task execution with fleet-level operational control (maintenance, energy, and dynamic tasking together) remain limited. The problem is the space-based counterpart of the sensor-network orchestration challenge that has motivated recent operational programs with the added complication that collecting assets themselves require ongoing management in an increasingly congested and contested operating environment. Hybrid RL-optimizer architectures have emerged as a promising paradigm where machine learning guides classical solvers rather than replacing them. However, existing approaches lack the strategic, multi-period coordination needed for fleet-level operations under uncertainty, and also lack multi-objective formulations that would allow operators to move along the execution-versus-fleet-health trade which remain largely unexplored.

Our work addresses three gaps: (1) extending strategic-guidance architectures to multi-agent settings with centralized training and decentralized execution (CTDE), enabling fleet-level coordination while respecting operational communication constraints; (2) identifying the credit-assignment structure this balance requires, showing that under naive shared credit, agents learn to refuse maintenance precisely when fleet health is lowest, and that counterfactual (marginal-contribution) credit corrects this failure, a mechanism we predicted from offline analysis and confirmed in training; and (3) enabling multi-objective doctrine discovery, where operators select policies along the execution-versus-stewardship front based on evolving mission priorities rather than fixed optimization weights.

We present a two-layer autonomous operations framework pairing learned strategic reasoning with classical combinatorial optimization. A multi-agent RL system (one agent per satellite) outputs strategic parameters each planning period: maintenance decisions, collection priority shaping across the tasking queue, and per-satellite energy envelopes. These parameters are consumed by a downstream combinatorial solver. During training we use integer linear programming (ILP) for provably optimal, deterministic schedules, eliminating solver noise so the reward signal reflects only the quality of the agents' strategic choices; the guidance object is solver-agnostic by design for deployment flexibility. The architecture functions as a serial shield: the solver guarantees constraint satisfaction regardless of the agents' proposals, so the learning layer can be enabled, disabled, or overridden without affecting safety. Task execution proceeds at the tempo of the planning cycle. Each period, the feasibility stage generates hundreds to over a thousand candidate sensor-target-window opportunities. The policy prices this menu, and the solver

selects a joint fleet plan satisfying access, energy, and health constraints, replanning every period as new taskings arrive and conditions evolve.

Training follows CTDE with multi-agent proximal policy optimization (MAPPO) on simulation rollouts. The training environment, COSMO (Constellation Operations Simulation for Multi-Objective Optimization), is a medium-fidelity simulator. It models stochastic sensor degradation via Ornstein-Uhlenbeck processes with health-to-quality coupling, weather fields from ERA5 reanalysis, and coupled onboard energy dynamics with battery carryover, with per-satellite resource envelopes enforced by the solver. Tasking arrivals follow Poisson processes and include custody-class multi-observation requirements with completion-threshold credit. The constellation is heterogeneous (optical and SAR), and a pluggable solver is embedded in the environment dynamics with parallelized rollout collection. We additionally report findings from investigating model-based gradient augmentation through a physics-informed learned dynamics model in solver-in-the-loop training.

From reward alone, the policy acquires fleet-management judgment that single-period optimization cannot represent. It pulls satellites offline for maintenance at a sharp learned health threshold closely tracking the dynamic-programming optimum. It operates the battery through deep charge cycles matched to eclipse and demand rather than holding a static reserve. It triages a dynamic, oversubscribed tasking queue by priority, realizing a large, provably-bounded fraction of the theoretically achievable observation value. The policy also makes the central balance of constellation operations explicit: near-term task execution trades against fleet longevity through an operator-controllable dial. Evaluated against a deliberately strong baseline ladder, the learned policy approaches the task execution of an expertly hand-tuned composite policy while surpassing its fleet-health outcomes. While approaches that are task-execution-only are near-optimal when decisions are decoupled, they fail outright once degradation and energy couple those decisions across satellites and time. Multi-objective extensions maintain competing goals as separate reward dimensions. Doctrine-conditioned training then discovers a front of operationally distinct doctrines, from collection-maximizing to fleet-preserving, selectable by operators as mission priorities evolve.

1. INTRODUCTION

Operating a space-based sensing constellation requires jointly satisfying two competing objectives: executing observation taskings and preserving the on-orbit assets that execute them. The mission exists to execute observation taskings: to point sensors at the objects an operator needs seen, inside the windows when they can be seen. But assets that execute those taskings are themselves subject to degradation and resource constraints. Sensors degrade with use. Batteries charge and discharge against an eclipse cycle. A satellite undergoes maintenance and is unavailable for collection. Every maintenance period is *observation capacity spent on future readiness*, and every period of aggressive tasking depletes the fleet that must execute tomorrow's queue. That is the genuine balance of concern here: the two imperatives pull in opposite directions, and the pull is coupled across every satellite in the fleet and every period of a campaign. This tension is not specific to any one sensing modality: a ground-based imager waiting on a weather window and a space-based sensor waiting on illumination and viewing geometry face the same underlying decision problem, an uncertain, perishable chance to collect, set against a fleet that must remain healthy enough to keep collecting (Section 3.3).

The task side is not a static list. Taskings arrive continuously, carry priority grades, and expire if windows pass unserved. Many are custody-class requirements, meaning they demand several observations of the same object and pay nothing until the full sequence completes. And the queue is persistently larger than the fleet can serve. At the operating point we evaluate, a fleet of five satellites over a 32-period campaign, feasibility screening surfaces a median of 724 candidate sensor-target windows in a planning period. Of those candidates, the fleet executes a median of only 14. That gap between what is possible and what gets done reflects a measurement taken at this specific fleet size and campaign length. It is not a fixed property of the task model itself. Every period is therefore an exercise in triage: which taskings to execute now, which to defer, and which to let expire. In that same measurement, the selection fraction falls from about 12 percent early in a campaign to about 5 percent late. This decline happens because supply thins as the campaign progresses, while the standing queue of unserved taskings keeps growing. (Section 3.2 walks through this funnel in full detail, including how it was measured.)

Current operational practice typically meets this with combinatorial optimizers planning each period on its own. Single-period optimization cannot represent the balance between present collection and future fleet capacity. Optimizing the current period in isolation has no way to anticipate a sensor's decline, no way to weigh a collection taken today against the fleet capacity it costs tomorrow, and no way to coordinate maintenance timing across satellites. It optimizes for the current period alone, which is not the question fleet operations possess.

This paper presents a system that learns to address these challenges, and characterizes what it learns in operator terms. Our contributions are the following.

First, a two-layer autonomous operations framework that is safe by construction. A multi-agent reinforcement learning (RL) system, one policy per satellite, sets strategic parameters each planning period, and an exact combinatorial solver turns those parameters into the fleet’s schedule. The solver enforces every operational constraint regardless of what the learned layer proposes, so the learning can be enabled, disabled, or overridden without any risk to feasibility.

Second, a characterization of learned fleet-management behavior capturing the operational judgement of expert practice. From reward alone, the policy pulls satellites offline for maintenance at a sharp learned health threshold, cycles onboard energy against eclipse and demand rather than hoarding it’s, and triages the oversubscribed queue. Measured against a deliberately strong baseline ladder, it reaches statistical parity with an expertly hand-tuned composite policy on task execution while ending campaigns with visibly healthier fleets.

Third, the credit-assignment story for joint tasking and stewardship: a real failure mode, its fix, and the limits of both. Under naive shared credit, agents learn to refuse maintenance exactly when fleet health is lowest. Counterfactual credit corrects this, a result we predicted offline and then confirmed in training. We also show the requirement is architecture- and scale-dependent: the production architecture secures stewardship structurally, and at fleet scale the counterfactual machinery becomes redundant for task execution at a compounding compute cost.

A fourth contribution is doctrine variants selectable at launch. The framework carries multi-objective machinery that lets an operator express mission priority as a behavioral posture, from collection-maximizing to fleet-preserving, selected by a launch flag rather than a retrain. We report the state of that machinery at our training budget.

Fifth, the system generalizes without retraining. The learned behaviors endure across campaign lengths and transfer across fleet sizes: a policy trained on one horizon and one fleet size holds up, zero-shot, at longer campaigns and larger fleets, each condition measured against its own bar.

Sixth, a measurement apparatus. Every score is reported against a certified per-episode bound on what an omniscient scheduler could achieve, the baselines are built to be strong, and evaluations pre-registered. This apparatus matters operationally: autonomy that cannot be certified will not be fielded, and certified bounds, strong baselines, and these controls allow a learned tasker to be measured against a provable standard.

The rest of the paper follows this order. Section 2 places the work among reinforcement learning for satellite scheduling, sensor-tasking orchestration, hybrid learning-and-solver architectures, and multi-objective reinforcement learning. Section 3 states the problem and environment it resides in. Section 4 describes the framework and its training. Section 5 sets out the evaluation methodology. Section 6 reports results at the operating point of 5 satellites and 32 periods. Section 7 examines the multi-objective doctrine machinery. Section 8 covers generalization to longer campaigns and larger fleets. Sections 9, 10, and 11 give the limitations, future work, and conclusions, respectively.

2. RELATED WORK

Reinforcement learning (RL) for satellite scheduling has matured fastest in the single-satellite setting, where policies for agile Earth-observing scheduling execute in milliseconds and suit onboard autonomy [1]. The recipe extends to space-to-space surveillance of resident space objects [2] and single parameter-shared policy across an Earth-observing constellation [3]. These systems show learned schedulers can carry realistic constraint structure and run fast enough to matter, but they optimize task execution while treating the collecting assets as fixed; fleet-level management of the collectors themselves, their degradation, energy, and maintenance, sits outside their scope.

Sensor-network orchestration for space domain awareness is an active ground-side area: RL for sensor tasking runs from single-sensor formulations [4] through deep RL for space-based [5] and ground-based [6] observers to multi-agent tasking of sensor networks [7], with autonomous agents now orchestrating operational telescope networks [8]. Our concern is the space-based counterpart, with one complication: collectors are satellites whose sensors degrade, batteries cycle, and maintenance removes them from the pool. Multi-satellite work integrating task execution with fleet-level control, maintenance and energy and dynamic tasking together, remains limited [9].

A second problem is hybrid architectures in which machine learning guides classical optimization rather than replacing it [10]: the solver retains exactness and constraint guarantees while the learned component supplies anticipation across periods. Existing hybrids largely target single-shot problems; ours is multi-agent and multi-period, pricing an oversubscribed menu every period with the solver being the sole feasibility authority.

Finally, multi-objective reinforcement learning (MORL) [11], [12] offers two routes to operator-selectable doctrine. The multi-agent setting has its own utility-based treatment [13]: a single policy conditioned on a preference vector at run time, or a family of specialists, each trained at a fixed preference vector and selected at launch. How each route performs is itself a result (Section 7). Together, this body of work sets up our contribution: a fleet that learns to execute taskings and steward itself at once, described in the terms an operator would use to run it.

3. PROBLEM FORMULATION: SDA TASK EXECUTION WITH A SELF-MANAGED FLEET

3.1 The task side

The mission serves a stream of observation taskings. Taskings arrive continuously rather than a daily batch: arrivals follow a Poisson process at a mean of ten new requirements per six-hour planning period, modulated by a diurnal cycle and by seeded surge events that multiply the rate three to five fold. Each tasking carries a priority grade, and urgent taskings expire three periods after arrival if not served. A large fraction are custody-class requirements: they demand multiple observations of the same object, and pay value only when the full sequence completes, mirroring the multi-observation sequences of catalog custody and object characterization. In our formulation, 70 percent of requirements need one collect, 20 percent need two, and 10 percent need three, with credit landing at the final collect. This completion-threshold structure is the custody mechanic: partial coverage of a two-collect requirement is worth nothing until the second collect lands.

Decisions run at the tempo of the planning cycle. Each six-hour period, every satellite receives a fresh menu of feasible collection opportunities, the fleet's plan is re-solved, and execution proceeds. The full replan cycle, observation assembly, policy pricing, and exact solve, completes in under a second per period at every fleet size we measured: whole-pipeline upper bounds of 0.35 seconds at 5 satellites, 0.45 to 0.49 seconds at 8, and 0.61 to 0.63 seconds at 12, taken from instrumented replays. The production solver is additionally capped at two seconds per solve by configuration. Replanning is therefore never the bottleneck; the scarce resource is the fleet itself.

3.2 Problem scale and fleet management

The tasking load is persistently oversubscribed, and the funnel behind that oversubscription is worth walking through in full (depicted on the environment loop, Fig. 2). Each period, feasibility screening finds a median of 724 candidate sensor-target windows. Ownership and presentation logic narrow that pool to about 230 solver rows, of which 202 carry learned prices into the optimizer, and the solver ultimately selects a median of only 14 collections. In short, out of hundreds of physically possible opportunities each period, only a handful are executed. This funnel is not an artifact of a few unusual episodes; it holds consistently across the 640 periods evaluated, with window count varying by a 10th-to-90th-percentile spread of 466 to 1,141 (full range 339 to 2,866) and selected count varying by an 8 to 26 spread.

The imbalance worsens as a campaign runs its course. Early on, fresh windows are plentiful (median of 875 per period) and backlog of priced-but-unserved candidates is small (median of 110). Late in the campaign, fresh windows have thinned to a median of 616, while the backlog has grown to a median of 289. The fleet is therefore choosing from a shrinking supply of new opportunities against a deepening queue of unserved candidates, exactly why the selection fraction falls from about 12 percent early to about 5 percent late. Every period is triage, hardening with campaign age.

The fleet executing this queue is itself subject to management requirements. Sensor health degrades stochastically: each satellite's health follows an Ornstein-Uhlenbeck process (a mean-reverting random walk) with strong downward drift, so an unmaintained sensor loses, in expectation, roughly six health points per hundred every period, and reaches its floor within about fourteen periods. Health has a direct functional consequence on collection value. It couples into the product: the quality stamped on every image is the sensor's intrinsic image quality scaled by the collecting satellite's health at collect time, and scoring pays quality directly. Low health also cuts imaging capacity in proportion. Maintenance restores health fully, but it takes the satellite out of the tasking pool. Energy is the second coupled resource: each satellite runs on a battery charged at 15 Watt-hours (Wh) per period, eclipse-coupled, with a 60 Wh capacity, against typed draws of 10 Wh per Synthetic Aperture Radar (SAR) image and 5 Wh per optical image, and the solver enforces each satellite's energy envelope on every plan. This structural trade-off drives the decision problem: time spent maintaining a sensor trades current collection opportunities for long-term health, while relentless collection degrades the assets required to handle future taskings.

3.3 The simulation and its scope

The training environment is a domain proxy, and we state its relationship to SDA. The task structure carries the SDA decision problem: dynamic arrivals with priority grades, perishable windows, custody-class multi-observation requirements with completion-threshold credit, and oversubscription by construction are the decision-theoretic content of space-domain-awareness tasking, and they are modeled directly. Some of the physics reflects Earth-observation and ground-sensor heritage rather than space-looking sensing. Collection is gated by an exogenous, time-varying uncertainty on whether a feasible opportunity can be taken, a condition no space sensor escapes: a space-looking tasker contends with illumination, solar phase angle, and background just as an imager contends with weather. We implement that gate with real historical fields replayed from ERA5 reanalysis [14], cloud cover for optical collection and rain rate for SAR. The ERA5-based weather gate is inherited from Earth-observation practice, but its role in the decision problem is domain-general, representing an uncertain, perishable chance to collect. We accept this modeling trade-off deliberately. The balance problem this paper studies, execution versus fleet self-management under coupled degradation and energy, does not depend on specific weather mechanics used to gate collection. Closing the gap from the simulation side is future work (Section 10).

3.4 Problem formalization

Formally, this is a cooperative multi-agent decision problem [15]. Each satellite is an agent. Each period, every agent emits strategic parameters, its maintenance decision, priority scores over its own opportunity menu, and its energy envelope, and an exact solver turns the fleet's joint proposals into one feasible plan. The reward is a four-dimensional vector: collection quality, fleet readiness, data delivery, and resource efficiency. A preference vector ω weights these dimensions into the training objective. Here ω is a launch-time mission-priority setting, a knob an operator sets when commissioning a policy, not a retraining exercise. The headline results in this paper use the balanced setting with equal weights. Section 7 examines what moving ω buys.

4. THE FRAMEWORK: LEARNED STRATEGIC GUIDANCE OVER AN EXACT SOLVER

4.1 Two-layer design

The system has two layers with a hard boundary (Fig. 1). The learning layer holds one policy per satellite. Per period, each agent outputs strategic parameters: a maintenance decision, a priority score for every opportunity on its own menu, and per-satellite energy and storage envelopes. The optimization layer is an exact integer linear program (ILP) consuming parameters and producing the fleet schedule. The solver enforces every constraint family on every solve. These include one window per requirement, pairwise slew and turnaround conflicts, per-satellite imaging-time budgets, storage and energy envelopes, and maintenance exclusivity. We call this the *serial shield*, in the spirit of shielding for safe RL [16], where a correct-by-construction layer overrides any unsafe proposal so the learned policy cannot violate constraints [17]. The solver

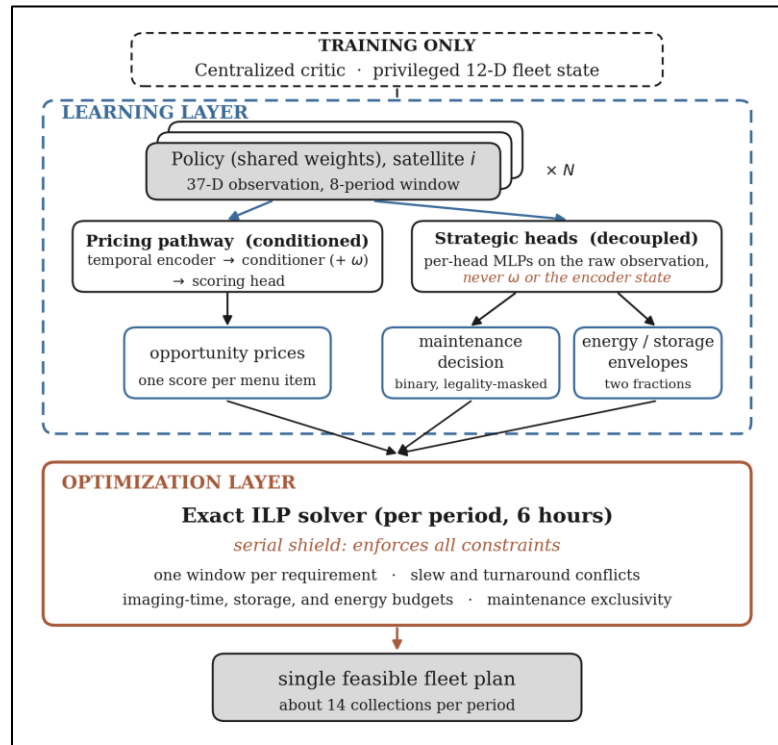


Fig. 1. The two-layer architecture. One parameter-shared policy per satellite emits strategic parameters into an exact ILP solver that assembles the single feasible fleet plan. Inside the policy, a conditioned pricing pathway (temporal encoder and preference weights) sets opportunity prices, while decoupled strategic heads read the raw observation only and set the maintenance and energy decisions; that isolation from the preference input and the encoder is what structurally protects fleet stewardship (Section 6.3).

The solver

guarantees constraint satisfaction regardless of learned layer proposals, so the RL can be enabled, disabled, or overridden without feasibility risk. This separation is what makes the learned layer safe to deploy aggressively rather than a liability to be hedged: because the solver absorbs all risk, the policy is free to specialize entirely on the judgment problem underneath it, deciding which opportunities are worth pursuing, when a satellite should stand down for maintenance, and how aggressively to spend energy and storage against an oversubscribed, constantly shifting queue. The learned layer only reprices the objective within the solver's guardrails, but that repricing is precisely the strategic layer carrying out fleet-level triage, maintenance timing, and long-horizon trade-offs; it can never enact an infeasible plan.

4.2 Centralized training, decentralized execution

Training follows the centralized-training, decentralized-execution pattern (CTDE) with multi-agent proximal policy optimization (MAPPO) [18]. Decentralized execution implies each satellite carries its own copy of the parameter-shared policy and operates on its own local observations, which include broadcast fleet-level aggregates but not the fuller privileged state used in training. Cooperation is instilled during centralized training, where a critic (value estimator used only during training) sees privileged fleet-wide state. Here, fleet-wide state is the centralized-critic idea shared by cooperative actor-critic methods [19]. At execution time no satellite needs another's decisions to act, the desired comm-outage-tolerant posture. Policies are parameter-shared: every satellite runs the same network on individual observations, an ingredient of scalable multi-agent learning [20] enabling fleet-size transfer (Section 8.1). Each agent's observation is a 37-dimensional vector covering its own state (health, storage, energy, maintenance history), summarizing opportunity menu, broadcast fleet-level aggregates, and physical-feasibility context.

4.3 The actor and credit assignment

Each agent's policy network has three parts. A temporal encoder, a two-layer gated recurrent network, reads the last eight periods of observations, so decisions condition on trends such as health decline and queue growth rather than snapshots. A scoring head prices the opportunity menu: each opportunity is a 17-dimensional descriptor (type, priority, urgency, window geometry, weather outlook, custody progress, and a completability estimate), and the head applies cross-attention from the agent's state to the menu and self-attention across the menu [21], so opportunities are priced in the context of what else is available. Attention over a variable set of candidates is the mechanism behind neural approaches to combinatorial selection and routing [22], [23]. Strategic heads (maintenance, energy and storage envelopes, downlink) are deliberately decoupled from the encoder: they read small slices of raw current-state features through their own small networks. Decoupling was a stewardship-robustness choice, its consequence for preference conditioning appears in Section 7; pathways and separation are shown in the Fig. 1. At the boundary to the solver, scores are standardized, z-scored and mapped to a fixed positive range, so the solver sees a stable price scale regardless of policy drift.

The critic is a scalar value function to estimate omega-weighted return directly from privileged global state. On top of shared team reward, the training configuration applies counterfactual, marginal-contribution credit on the collection dimension [24]: each period, the solver is re-run with each agent's opportunities removed, and the agent is credited the difference. This is one of two standard answers to multi-agent credit assignment, the other being value decomposition of team return [25], [26]. In operator terms, each satellite is credited by what the fleet would lose without its contribution. A satellite that stands down for maintenance is not punished for collections its neighbors cover; the credit is maintenance-exempt by construction. This costs one extra solve per agent per period during training and skipped entirely at execution. Why this machinery is in the configuration, when it is needed, and when it is redundant is a measured story with a failure mode at its center (Section 6.3).

4.4 Training recipe, and what does not work

The recipe has two stages. Stage one trains from scratch: 300 iterations of MAPPO across parallel simulated campaigns, with a value-informed reference schedule distilled in early and annealed away, standard clipped policy updates, and a KL-based early stop on the discrete heads. Stage two is a short fine-tune of the selected seed: 100 iterations with a pricing teacher. Distillation here means training the policy directly toward a known-good pricing rule, rather than discovering it through trial and error alone. The teacher is the completion-aware price, priority divided by remaining collects and scaled by quality, and the fine-tune regresses the policy's scores toward it while normal RL training continues. Section 6.4 shows this fine-tune is precisely attributable and is the headline's entire lift.

One negative result shaped this recipe, and we report it because it generalizes. We investigated model-based gradient augmentation: backpropagating through a physics-informed learned dynamics model [27], [28] to give the policy gradient information through the solver. The path is measurably inert. The differentiable route can only see the score

distribution's non-ranking moments; score standardization at the solver interface neutralized what remained. A controlled experiment, world-model gradients off versus on over 500 iterations through the full ramp window, moved capture by -0.002, replicated at -0.001. The complementary trial-and-error channel, score noise correlating with credited advantage through solver selections, measures near zero across four checkpoint generations, with a pooled correlation magnitude of at most 0.005. The through-line is simple: no gradient channel through an exact solver carries selection information. Pricing knowledge is therefore installed supervised, through warm initialization and distillation, while RL carries coupled discrete decisions, maintenance timing and envelope posture, where trial and error informs.

4.5 COSMO, the training environment

Training runs in COSMO (Constellation Operations Simulation for Multi-Objective Optimization), a simulator designed to support constellation-autonomy research beyond this paper's specific experiments.

The environment's physics center on Ornstein-Uhlenbeck sensor-health degradation with health-to-quality coupling: stamped image quality is the sensor's intrinsic quality scaled by collecting-satellite health. Onboard energy is a battery with eclipse-coupled charging and typed per-image draws, enforced as a per-satellite envelope in every solve. Weather is replayed real-weather fields from ERA5 reanalysis [14], spanning 2015 to 2022, gating optical collection on cloud and SAR collection on rain rate. Orbital access geometry generates per-period opportunity structure for a heterogeneous fleet, 3 optical and 2 SAR at operating point.

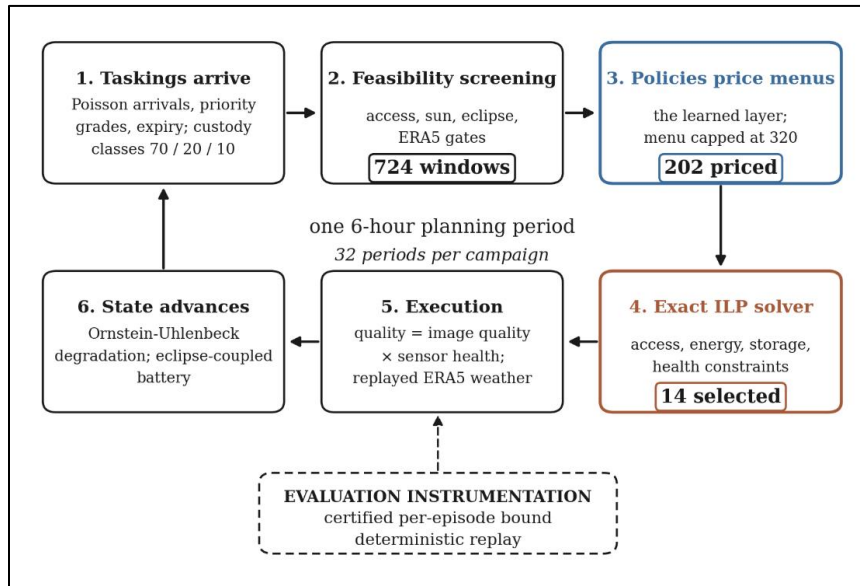


Fig. 2. The COSMO environment loop. The per-period cycle from tasking arrivals through pricing, the exact solve, execution, and state advance, annotated with the median tasking funnel (724 feasible windows to 202 priced candidates to 14 collections).

Task machinery layers Poisson arrivals with priority grades, diurnal and surge modulation, and urgent expiry onto custody-class multi-observation requirements with completion-threshold credit, producing oversubscription by construction. The funnel of Section 3.2 is COSMO's native regime, not a tuned corner case, and full per-period loop is shown in Fig. 2.

The platform's core qualities include an exact solver embedded in the environment loop with a pluggable backend, deterministic episode replay, and parallelized rollout collection; same-seed retraining after a major infrastructure change reproduced final network weights byte-for-byte, and every

episode carries a certified optimality bound (Section 5). These qualities are what made our measurement discipline possible. The environment is designed for reuse beyond this work, an intent rather than measured result: its fidelity features are individually flag-gated, so the platform composes from a fast baseline to full coupled world.

5. EVALUATION METHODOLOGY

Every capture score is reported against a certified per-episode bound: the highest observation value that could provably have been achieved on that same episode. The bound is computed per episode as a joint linear-programming relaxation presented every opportunity pool, with battery dynamics included, tightened by an exact solver under a fixed time budget; if the tightening times out, the dual outer bound is kept (never the incumbent) so the bound is certified conservative. This makes headroom visible instead of hidden. A capture ratio of 0.79 means 79 percent of what was provably achievable on those exact episodes, and the remaining 21 percent, is a measured quantity we can decompose (Section 8.1). The bound computation carries wall-clock jitter at the seventh decimal, so we quote capture ratios to at most four decimals at the operating point and at most three decimals at the longer horizons and larger fleets, where the bound additionally carries dual slack of at most 0.002, stated here once.

Baselines are deliberately strong and value-aligned: every script prices opportunities with the same priority information the policy sees. The ladder runs from a random-legal floor, through execution-only scripts (greedy optimal-price variants that never maintain), through an oracle-threshold maintenance script (maintain below health 0.7 when payback remains), up to an expertly hand-tuned composite stewardship policy and its foresight variant. The composite is the crucial bar: combining oracle maintenance rule with tuned pricing to reach 0.8201 of certified achievable value. Section 6.1 reports the full ladder (Table 1). Design intent is beating weak baselines proves nothing in this domain; the claims in this paper are made against the strongest scripted stewardship at our disposal.

All headline evaluation runs on a frozen set of 20 evaluation episodes with fixed construction seeding; every policy and every script sees identical worlds, identical arrivals, and identical weather. Ten from-scratch training seeds and analysis plans for major evaluations were pre-fixed. Each campaign length and fleet size scored only against own frozen episodes, own baseline bars, and own certified bounds; conditions never pooled. A note on reproduction: re-running training with identical seeds after major infrastructure change reproduced final actor and critic weights byte-for-byte over 300 iterations. The single evaluation-side discrepancy found, three parts in ten million in one metric, was diagnosed to a documented solver wall-clock budget rather than nondeterminism. Machine-scoped form of reproducibility caveat is stated in Section 9.

6. RESULTS AT THE OPERATING POINT: 5 SATELLITES, 32 PERIODS

6.1 The main result

The learned system is reproducible: 0.7922 ± 0.0151 of the certified achievable value, from ten seeds, with a range of 0.760 to 0.816, converging after the actor unfreezes partway through training. That is the from-scratch policy, before any fine-tune. With a short distillation fine-tune, the policy reaches statistical parity with an expertly hand-tuned composite on task execution, at 0.8169 with the composite's 0.8201 inside the confidence interval, while surpassing its fleet-health outcome, 0.7505 against 0.6294 end minimum health (Fig. 3, which places both policies and an execution-only script side by side on capture, requirement satisfaction, and end-of-campaign health). Stated operator-side, the policy captures approximately four fifths of the provably achievable observation value, with the remainder decomposed in Section 8.1 and a per-condition percent table carrying the bound caveats (Table 2). The parity holds in both of the operator's currencies, not observation value alone. In custody terms, the fraction of multi-look requirements whose full observation sequence completes, what maintaining a track actually means, the distilled policy satisfies 0.3541 against the composite's 0.3545, the composite again inside the policy's confidence interval; the near-match is therefore not an artifact of capture currency but holds equally in the metric an operator reads as tracks kept. A requirement counts only when its full custody sequence completes, and urgent taskings that expired unserved, about 1.5 percent of arrivals, sit outside the base.

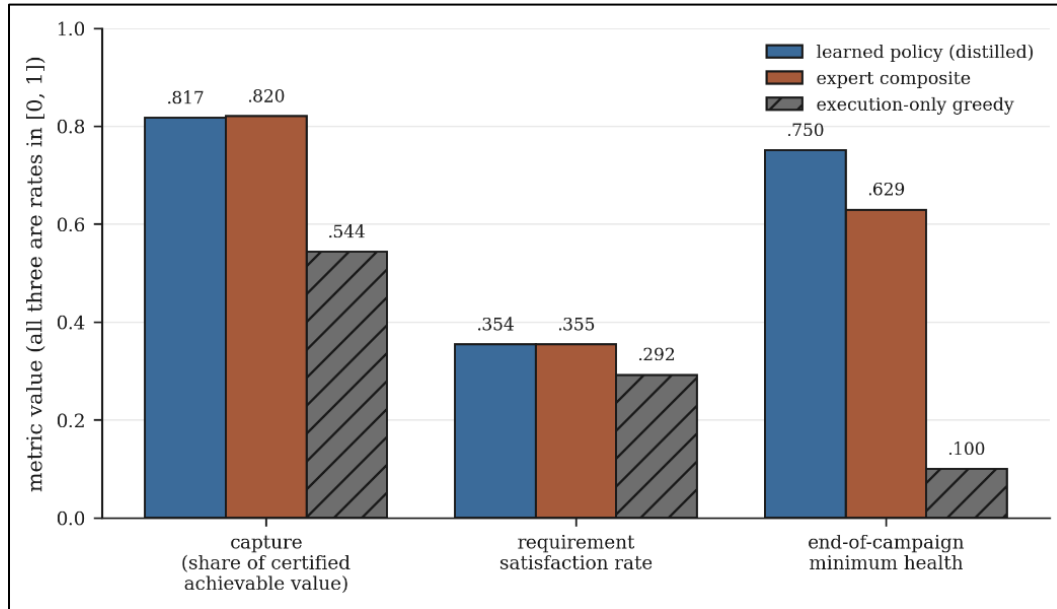


Fig. 3. Performance at operating point (5 satellites, 32 periods). Capture, requirement satisfaction, and end-of-campaign minimum health for distilled policy, expert composite, and an execution-only script.

Coupling is most pronounced at the execution-only end of the ladder, where these policies fail in all 100 of 100 paired episodes against the learned policy. The collapse is stewardship-driven, not myopia-driven. The expert composite is also myopic and does not collapse; execution-only scripts end at the 0.10 health floor, having exhausted their fleet's health reserves. The opposite failure is equally instructive: a policy maintained at random keeps fleet health (0.8251 end health) but captures almost nothing (0.2631), because stewardship without triage also fails. The two edges of the table are the two one-sided failure modes, and the policy family lies between on both axes.

Table 1. Task execution and fleet stewardship at operating point (5 satellites, 32 periods). Capture and end-of-campaign minimum health for learned policies and full baseline ladder, all on the same frozen 20 episodes.

Policy or script	Capture (share of certified achievable)	End-of-campaign minimum health
Learned policy, from scratch (10 seeds, mean $\pm$ sd)	0.7922 $\pm$ 0.0151	0.7310 $\pm$ 0.0349
Learned policy + pricing distillation (selected seed)	0.8169 (95% CI 0.8074 to 0.8275)	0.7505
Expert composite (steward)	0.8201	0.6294
Expert composite + foresight	0.8203	0.6294
Oracle threshold script (maintain below 0.7)	0.7980	0.6294
Execution-only: optimal-price greedy	0.5435	0.10 (floor)
Execution-only: completion-aware	0.5463	0.10 (floor)
Execution-only: foresight	0.5469	0.10 (floor)
Never-maintain	0.5182	0.10 (floor)
Random-legal floor	0.2631	0.8251

Notes. (a) Capture is quoted to at most four decimals per the operating-point precision rule.

(b) Policy seeds $N = 10$; the distilled row is one fine-tune of the best seed, evaluated on $N = 20$ episodes.

(c) The policy wins every paired episode against every non-maintaining script, 100 of 100, at margins of +0.308 to +0.361 (per-period capture).

(d) The three threshold-rule scripts share an end health of 0.6294: the maintenance rule, not the pricing, determines fleet condition. Random-legal ends healthy (0.8251) while capturing only 0.2631; the table's edges are the two one-sided failure modes.

6.2 Learned behaviors in operator terms

The following behaviors were acquired from reward alone rather than specified. Maintenance timing is learned: a sharp health threshold (maintenance probability of 1.00, 1.00, 0.93, 0.07, and 0.00 across descending health bands, correlation -0.84) matching the oracle threshold rule of the baseline ladder (Fig. 4, which plots maintenance probability against health band for the learned policy and the oracle rule). Health is the SDA lever here: the sensor's health scales quality stamped on every image, a decayed sensor returns observations too weak to sustain custody or tighten an orbit solution, and the learned threshold is the policy protecting its own ability to produce usable tracks. In duty-cycle terms, the policy holds each satellite in maintenance for about 15 percent of satellite-periods, a campaign mean of 15.1 percent with a seed range of 12.9 to 16.3 percent, pulling a satellite offline when

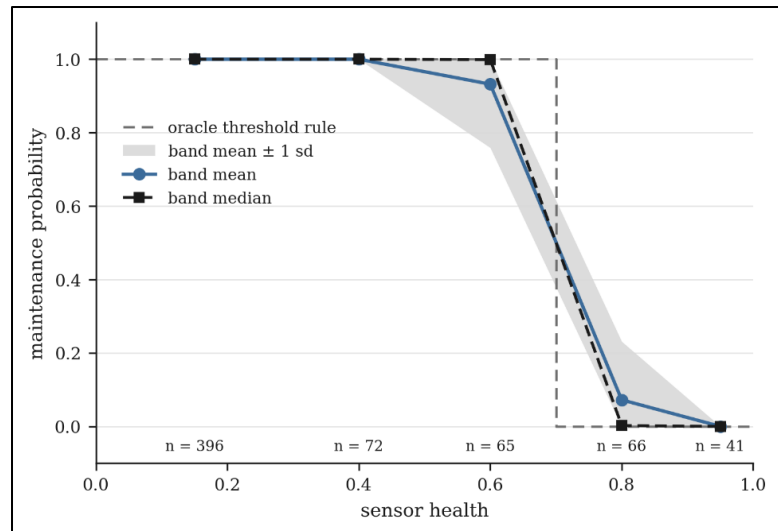


Fig. 4. Learned maintenance threshold. Probability of choosing maintenance against sensor-health band, with the learned policy's band means and medians overlaid on the oracle threshold rule, which the band mean tracks at correlation -0.84. Here, n refers to the number of policy decisions that fell into each sensor-health bin.

health falls below about 0.7 and almost never above it. The wider 13 to 17 percent span reported elsewhere is the family-wide census across sizes and horizons.

The policy actively cycles the battery rather than holding a static reserve: charge depth tracks eclipse and demand, with state-of-charge troughs of 0.07 to 0.13 and energy binding in 56 to 59 percent of periods, and no late-campaign collapse or hoarding at any horizon. This yields sustained collection throughput rather than trading future coverage for present collection: as the standing queue deepens the policy draws down charge rather than defending a reserve, collections per period rising across the campaign thirds (about 13, 14, and 16) while fleet charge falls (0.735, 0.630, 0.559). Charge distribution carries no hoarding mass pinned near a full battery. The binding fraction indicates cycling reflects active management of a scarce resource rather than slack.

Triage under oversubscription follows a known-optimal myopic price. After price distillation, the policy's triage matches known-optimal myopic price at rank agreement 0.96 overall, at least 0.89 in every priority band, up from 0.06 to 0.08 in low bands before distillation. The policy allocates limited collection opportunities to urgent and high-value tasks first, the perishable new-object windows and high-priority custody sequences, which is exactly the prioritization the myopic price encodes. These figures describe the distilled headline policy; the from-scratch policies agree less strongly. In plain terms, the fine-tune is what teaches the policy to sort the low-priority tail correctly, exactly where a price teacher should help.

6.3 The credit-assignment mechanism

The interesting question behind these behaviors is what credit structure makes stewardship learnable at all. It is not automatic, and the failure mode is instructive.

Under naive shared credit, earlier architecture converged to an inverted maintenance policy, declining maintenance when fleet health was

lowest, with $P(\text{maintain})$ of 0.032 at low health and a positive health correlation, predicted from offline analysis and confirmed in training; counterfactual credit corrects it, to 0.917 with a correlation of -0.756 (Fig. 5, left, the maintenance curve under each credit scheme). The run-level comparison tells a similar story: measured at final checkpoint, the naive-credit run holds about half the baseline's maintenance (23 against 49) and materially lower readiness (0.62 against 0.82) over frozen episodes. Intuitively, when the whole fleet shares one reward, a satellite that stands down to maintain sees its teammates' collections credited to everyone, so the gradient attributes reduced contribution to the maintaining satellite, precisely when the fleet can least afford reduced maintenance. Counterfactual credit corrects this by crediting each satellite with the fleet's marginal loss in its absence. These findings are scoped to the architecture measured, the earlier-generation heads and shaping.

The production architecture does not need that runtime correction because it is structurally protected: every modern no-counterfactual configuration holds the oracle-shaped curve at 5, 8, and 12 satellites, with correlations of -0.854 , -0.839 , and -0.894 (Fig. 5, right, the same curve across fleet sizes). Structural protection here means the decoupled maintenance head reading raw health features, maintenance-exempt shaping, and terminal salvage; a no-counterfactual configuration is not a no-credit configuration. And at scale the machinery costs more than it returns. At scale the counterfactual term is redundant for capture at compounding cost: no-cf at or above the cf family at every size (0.810, 0.819, 0.822 versus 0.792, 0.771, 0.791) with +27, +32, and +35 percent compute (the 5-satellite +27 is co-load-approximate per the bundle; +32 and +35 are clean); the per-agent marginal signal dilutes monotonically (0.046, 0.037, 0.029); the historical 12-satellite instability fingerprint tracks the counterfactual shaping.

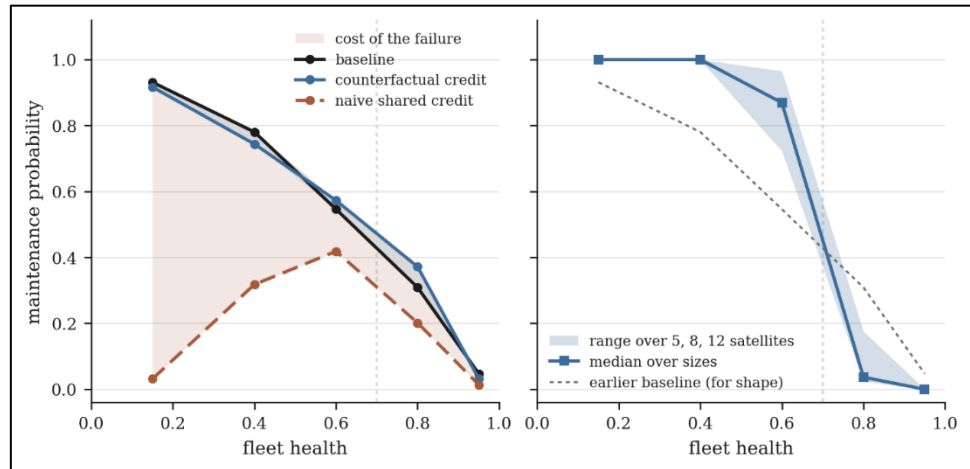


Fig. 5. Credit-assignment failure mode and structural fix. Maintenance probability against fleet health under three credit schemes in earlier architecture (left) and across three fleet sizes in final architecture (right).

The lesson generalizes: credit-assignment requirements are architecture- and scale-dependent, and structural protection can replace runtime credit machinery. For an autonomous SDA fleet the stakes are concrete: the failure mode is a constellation that learns to forgo maintenance as its sensors degrade. Designing it out is what allows the fleet to operate and keep the catalog without a human forcing maintenance.

6.4 The distillation attribution

The headline lift is precisely attributable, and the attribution is airtight because of a control. The distillation term is the entire lift: matched control regresses to seed mean (0.7924) while the distilled twin gains +0.0245 with disjoint confidence intervals and rank agreement rising in every priority band (low bands from 0.06 to 0.89). Full decomposition is donor 0.8056, control 0.7924 (a change of -0.0132), and distilled 0.8169 (a change of +0.0245), with confidence intervals of [0.782, 0.803] against [0.807, 0.828]. The control is what makes this clean: vanilla fine-tuning of a best seed is not neutral, because it re-samples toward the population mean, so the twin with everything held equal except the price teacher is the only fair comparison. The per-period metric moves harder than the episode metric, from 0.7005 to 0.7751, which signifies better selection every period, exactly what a price teacher should buy.

6.5 The trade dial, scoped

Whether the balance point itself is adjustable has one measured, scoped answer. On a checkpoint that expresses terminal-maintenance behavior, masking that maintenance buys +0.034 capture for -0.109 end health (capture 0.7125 to 0.7469, end health 0.726 to 0.617, over the same 20 episodes). This is a property of such a policy, not of the world: a second, independently trained line moved only +0.0017, having no terminal maintenance to trade. The fuller question of doctrine is Section 7.

7. MULTI-OBJECTIVE DOCTRINE: WHAT THE DIAL ACTUALLY DOES

7.1 The multi-objective machinery

The multi-objective machinery has two forms. The preference vector ω can be concatenated into policy, in the manner of preference-conditioned single-policy multi-objective learning [29], and, as a separate architectural flag, ω can also be fed to decoupled strategic heads. That flag matters. Without it, the strategic heads receive no information about doctrine by construction. Preference conditioning is mechanically live: the maintenance head's response to doctrine weights moves from exactly zero to two orders above the detection bar with one architectural flag, from a response of 0.0 to 1.142 and 2.330 across two independently trained probe runs, against a detection bar of 0.01. The reason it is exactly zero without the flag is the design choice of Section 4.3: decoupled heads read raw state only, so nothing about doctrine reaches them unless doctrine is wired to their input. The machinery is therefore genuinely live. Whether it produces coherent doctrine is a separate question.

7.2 The measured result

At the training budget evaluated, neither multi-objective route produced a measured Pareto front: mixed- ω conditioning is live but doctrine-incoherent, and the fixed- ω specialist family traces no capture-versus-health trade, since the highest-collection specialist reads best on both axes at once. Both current-generation attempts found no front at the 300-iteration budget; the earlier-generation conditioned front is that generation's

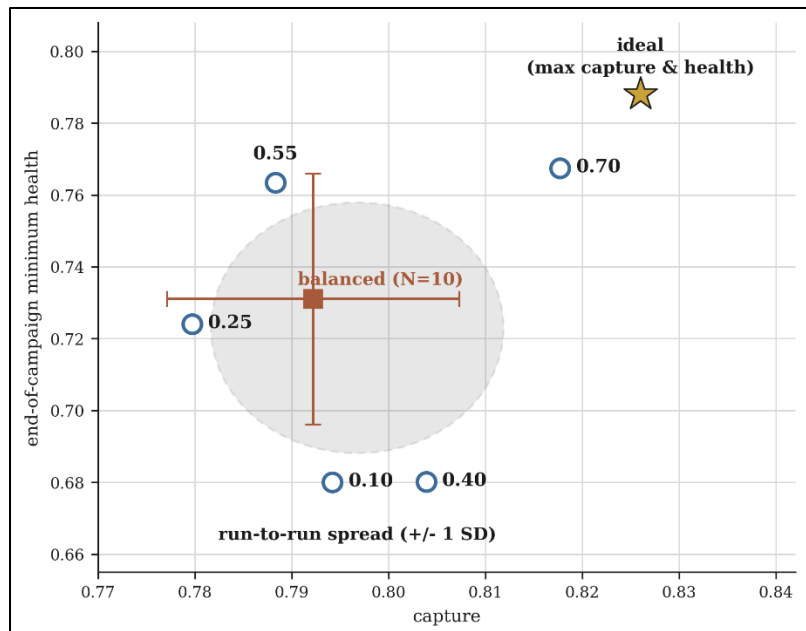


Fig. 6. Doctrine specialists in capture-versus-end-health plane. Five fixed-preference specialists (open circles, labeled by preference weight on collection, each a single run) and balanced campaign reference (filled, $N = 10$ with run-to-run error bars); shaded ellipse is balanced policy's run-to-run spread and star marks ideal corner of maximum capture and health.

result and is not claimed for the current system.

Specialist data make the non-result concrete. Five fixed-omega points along the collection-fleet edge give captures of 0.7797 to 0.8177 and end health of 0.6800 to 0.7674, both non-monotone in preference weight (Fig. 6, the specialists and balanced reference in the capture-health plane, with balanced policy's run-to-run spread shaded and ideal corner marked). Each specialist is a single run, and the five points sit within the balanced policy's run-to-run spread, so no ordered front toward the joint optimum emerges. Mixed-omega runs are flat on capture, spanning about 0.018, with maintenance-census response inverted relative to doctrine. Neither route buys a clean trade at this budget.

7.3 What survives and matters operationally

What survives operationally is selectable specialists: doctrine variants selectable by launch flags alone, each at campaign-class performance, expressing doctrine as behavioral texture; at this problem and budget a multi-policy MORL approach delivers where single-policy conditioning does not yet. At their corners, the collection-quality specialist reads 0.8177 and the fleet-readiness specialist 0.7942, both campaign-class, and the head-conditioning architecture is free at fixed omega, giving a balanced readout of 0.8196. For a practitioner weighing single-policy conditioning against a family of specialists, on this problem and at this budget, the multi-policy route delivers and the single-policy route is live but not yet doctrine-coherent. Doctrine, in this system, is something you select at launch rather than dial at run time.

8. GENERALIZATION: LONGER CAMPAIGNS AND LARGER FLEETS

8.1 Endurance and fleet-size transfer

The behaviors of Section 6.2 are not artifacts of the 32-period horizon they were trained at. We evaluated the system at 1.5x and 2x campaign lengths, 48 and 64 periods, each length carrying its own frozen evaluation episodes, its own baseline bars, and own certified bounds. Behaviors extend to 1.5x and 2x campaign lengths: the threshold rule is health-anchored not horizon-anchored (correlation -0.86 to -0.92 at every horizon), cycling is stationary, capture within 0.013 to 0.044 of each horizon's own expert bar (Fig. 7, left, capture shortfall against each length's own bar). The energy posture keeps the same gentle profile at every length, with no late-campaign hoarding or collapse.

Transfer is the operator-relevant finding. A 32-period-trained policy transfers zero-shot to longer campaigns near-perfectly on capture; training natively at the horizon buys terminal stewardship (5 of 6 runs counter the zero-shot maintenance taper). A policy trained on 32-period campaigns transfers zero-shot to 48- and 64-period campaigns with negligible loss in capture (Table 2). What native training at the longer horizon adds is end-of-campaign fleet condition. Meanwhile the execution-only comparison compounds the other way: the completion-aware script falls from 0.546 at 32 periods to 0.503 at 48 and 0.483 at 64. Coupling punishes short-sightedness harder the longer the campaign runs.

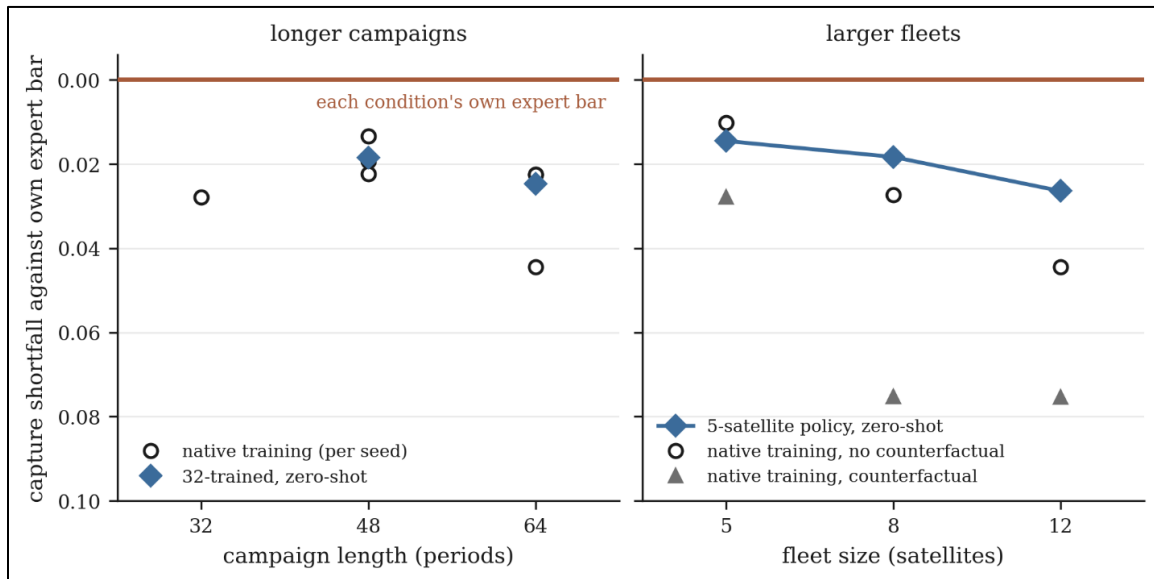


Fig. 7. Generalization across campaign length and fleet size. Capture shortfall against each condition's own expert bar for campaign lengths (left) and fleet sizes (right), comparing zero-shot transfer with native training.

Table 2. Learned capture by generalization condition. Learned capture at each campaign length and fleet size as a share of that condition’s own certified bound, with each condition’s expert bar.

Condition	Trained natively here	5-satellite, 32-period policy, zero-shot	Own expert bar
32 periods, 5 satellites (operating point)	79.2% $\pm$ 1.5 (10 seeds)	80.6% (home condition)	82.0%
48 periods	81.0 to 81.9% (3 seeds)	81.4%	83.2%
64 periods	79.8 to 82.0% (3 seeds)	81.8%	84.2%
8 satellites	81.9% (no counterfactual)	82.8%	84.6%
12 satellites	82.2% (no counterfactual)	84.0%	86.6%

Remaining headroom shifts to physics with length: energy-forced idle reaches about 78 percent of addressable losses at 2x horizon; the remaining shortfall is battery arithmetic, not judgment. Residual headroom itself falls with horizon, from 0.202 to 0.185 to 0.183 of the certified bound. The share a better scheduler could in principle recover shrinks from 31 percent to 22 percent. The system is closer to the physics wall at long horizons, not further from it.

Parameter-shared policy transfers zero-shot across fleet sizes, landing within 0.02 to 0.03 of each size’s expert bar (Fig. 7, right) and above every natively trained policy at matched budget, with better end-health than the scripts; per-condition captures and bars are in Table 2, each size evaluated in its own frozen world. Policies trained from scratch at 8 and 12 satellites fall below the zero-shot transfer at matched budget. Running one parameter-shared policy across a constellation echoes the single-agent-to-constellation scaling studied for Earth observation [3], here without any retraining at the new size. The denominator caution, stated once for both generalization sections: capture ratios at different conditions use each condition’s own certified bound, so cross-condition comparisons are made in each condition’s own currency and never pooled. We state the zero-shot dominance as measured and do not fully explain it; 300 iterations may under-train the larger worlds (Section 9).

8.2 Cooperation, measured

A fair question for any multi-agent claim is whether the satellites actually cooperate, or whether the architecture only makes it look that way. We tested this directly with a controlled perturbation at evaluation time: force one satellite’s observed health below its maintenance threshold and watch what every other satellite’s learned scoring does in that same period. Cooperation is architectural, and that is now a measured claim: perturbation probes show teammate policies do not re-price a stood-down neighbor’s targets (score shifts three orders below the bar, every size, both credit types) while the solver reallocates the same period. The channel for coordination exists, since fleet health is observed and the same perturbation flips the targeted satellite itself to maintenance, so this is a clean mechanism statement rather than a power-limited null: the policies price their own opportunities and leave reallocation to the optimization layer.

That reallocation is strong and grows with fleet size. Stand-down absorption scales with fleet size (+0.10, +0.24, +0.33 pickup at 5, 8, and 12, and the fraction of stand-down events showing any positive pickup rises with it, about 0.61, 0.78, and 0.83) and is weakest when maintenance synchronizes; the operational implication is to stagger maintenance windows, a recommendation derived from the measurement, not a measured intervention. This absorption trend is the behavioral face of the substitutability trend in the credit analysis of Section 6.3: as fleets grow, the per-agent marginal signal dilutes, and more substitutable fleets absorb stand-downs more completely. And the burden of stewardship is not concentrated on a small subset of satellites. Fleet robustness comes from interchangeability, not roles: within-type maintenance burden is near-uniform (CV at most 0.10, maximum 0.093); any satellite covers any role. For an operator, the package is a fleet with no single point of stewardship failure, and, in coverage terms, no single point of custody failure: when any sensor drops for maintenance its reachable targets are largely picked up the same period by the rest of the fleet (Fig. 8, same-period pickup by fleet size and the falling per-agent marginal signal), and that resilience strengthens as the constellation grows.

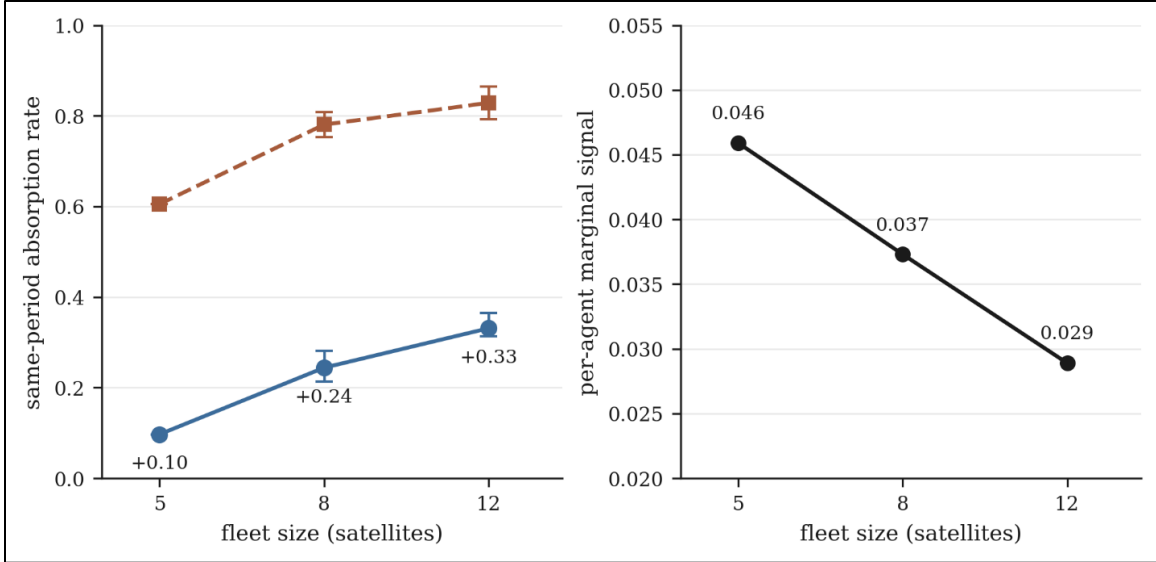


Fig. 8. Stand-down absorption and its mechanism. Left, by fleet size: increase in teammates' same-period pickup of a stood-down satellite's targets over matched controls (blue, with run-to-run range bars) and fraction of stand-down events with any positive pickup (terracotta, dashed). Right: per-agent marginal contribution signal over same sizes. Two panels are separate probes sharing only size axis.

9. LIMITATIONS

The principal limitations are as follows.

First, this is a single world. One engine-profile family, one heterogeneous fleet mix (3 optical, 2 SAR at the operating point), one battery configuration, with one scarcity variant probed once. We make no cross-world generalization claim, and Section 3.3 states the domain-proxy relationship to operational SDA.

Second, the headline operating point is a 5-satellite fleet. The 8- and 12-satellite results of Section 8.1 are evaluation-scale results: native training at those sizes used the same 300-iteration budget as the operating point, which may under-train the larger worlds, and the zero-shot dominance there is stated as measured, not explained. A 12-satellite training instability observed in an earlier configuration appears in the final architecture only as an attenuated statistical fingerprint, an elevated advantage correlation on counterfactual-credited runs of about 0.91 versus 0.87 without, with no training divergence at this budget.

Third, instance counts are stated per point throughout. Error bars at the operating point come from ten training seeds. The distilled headline is one fine-tune pair evaluated on 20 frozen episodes. The doctrine points of Section 7 are one run each. Per-size and per-horizon native results are one to three runs each. The behavioral threshold curves are single-checkpoint probes whose shape replicates across horizons and sizes.

Fourth, the doctrine result is scoped to its budget. Both current-generation front attempts, conditioned and specialist, were negative at 300 iterations. The earlier-generation conditioned front is presented strictly as that generation's result. The trade dial of Section 6.5 is one demonstrated instance on one checkpoint, and a second, independently trained policy line did not replicate it.

Fifth, compute. Training cost grows superlinearly with horizon, at 69, 127, and 243 seconds per iteration at 32, 48, and 64 periods. The counterfactual credit adds 27 to 35 percent of training wall-clock at 5 to 12 satellites, the 5-satellite figure approximate and measured under compute co-load, for a term Section 6.3 shows is redundant for task execution in the final architecture.

Sixth, metric precision. Capture ratios carry solver-budget jitter at the seventh decimal and are quoted to at most four decimals at the operating point, and to three decimals at the longer horizons and larger fleets, where the certified bound also carries dual slack of at most 0.002. Bounds are certified conservative by construction.

Finally, reproducibility claims are machine-scoped. Byte-exact retraining was validated on one machine and one thread topology, and all measurements this cycle shared a constant background load on both sides of every comparison.

10. FUTURE WORK

Several directions follow from the ceilings above. The weather gate, now a deterministic replay, should become a stochastic realization at execution scored against an omniscient bound so headroom stays visible. A doctrine-coherent conditioned front should be retried with the reward-shaping interaction of Section 7 tested directly and training budget scaled as a second variable, since 300 iterations may be too few. The horizon and fleet-size confounds of Section 8 can be resolved by matching training budget to condition, separating genuine transfer from under-training. The cooperation result, resting on single stand-down perturbations, invites dose-response and simultaneous-stand-down probes. The domain-proxy gap of Section 3.3 can be closed from the simulation side with space-looking sensor physics, such as solar phase angle, illumination geometry, and glint, which vary continuously rather than aligning to fixed planning periods, real catalog task structure, and larger fleets. On the learning side, a space-looking sensing task may also call for reconsidering how the policy represents strategic priorities, since the fixed per-period menu structure used here assumes opportunities can be cleanly grouped into discrete windows. And long-horizon bounds can be tightened with deterministic node budgets, applying the counterfactual machinery only where it earns its compute. Decision periods significantly shorter than the six hours used here are also worth exploring. The learned layer only prices the opportunity menu rather than re-deriving a schedule from first principles, so its computational overhead scales far more gently than a purely classical solver's would. This suggests the approach could support much more frequent replanning, and in turn larger constellations, without a proportional growth in decision latency.

Scaling this architecture to fleets much larger than those evaluated here raises its own questions. Parameter sharing keeps the policy itself cheap regardless of fleet size, but the counterfactual credit term does not: Section 6.3 shows its per-agent marginal signal dilutes as the fleet grows while its compute cost keeps climbing, so at some fleet size a flat, per-satellite architecture may need to give way to a hierarchical one, with a higher-level coordinator managing sub-fleets that each run the present per-satellite policy. Short of a hierarchical redesign, an evaluation method for selecting which subset of a larger available fleet to commit to a given campaign, rather than committing the whole fleet, would extend the zero-shot transfer results of Section 8.1 into an explicit fleet-selection tool for operators managing more satellites than a single campaign requires.

11. CONCLUSION

The coupled execution-versus-stewardship balance can be learned from reward alone. From reward alone, a fleet learns to decide when to collect and when to preserve itself, holding statistical parity with an expertly hand-tuned composite on task execution while ending campaigns with visibly healthier fleets: 0.8169 against the composite's 0.8201 inside the confidence interval, at 0.7505 end minimum health against 0.6294. It does this inside a two-layer framework where an exact solver keeps safety by construction and the learned layer only reprices, so the autonomy can be enabled, disabled, or overridden without feasibility risk.

The behaviors remain valid where operations needs them to. They endure across campaign lengths and transfer across fleet sizes without retraining, each condition measured against its own certified bound, and doctrine is selectable at launch by a flag rather than a retrain. Along the way the credit-assignment story carries a reusable lesson: the failure mode where agents refuse maintenance exactly when it matters is a genuine risk, and it can be secured structurally rather than with runtime machinery, with the requirement depending on both architecture and scale. The ceiling is as informative as the headline: at long horizons, four fifths of what remains is battery arithmetic, not judgment, which indicates where further effort will and will not pay.

The measurement discipline is the reusable part. Certified per-episode bounds, strong baselines, and pre-registered evaluation let each claim be checked against a provable standard, and we offer that pattern, as much as any single result, as the contribution most portable to applied reinforcement learning for SDA, where fielding an autonomous tasker at all depends on being able to certify what it will and will not do.

REFERENCES

- [1] A. Herrmann and H. Schaub, "Reinforcement Learning for the Agile Earth-Observing Satellite Scheduling Problem," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 5, pp. 5235–5247, 2023.
- [2] D. Huterer Prats, H. Schaub, and C. Wheeler, "Reinforcement Learning for Space-to-Space Surveillance: Autonomous Scheduling for Resident Space Object Imaging," *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2025.

- [3] A. Herrmann, M. Stephenson, and H. Schaub, "Single-Agent Reinforcement Learning for Scalable Earth-Observing Satellite Constellation Operations," *Journal of Spacecraft and Rockets*, vol. 61, no. 1, pp. 114–132, 2024.
- [4] R. Linares and R. Furfaro, "Dynamic Sensor Tasking for Space Situational Awareness via Reinforcement Learning," *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2016.
- [5] P. M. Siew, D. Jang, T. G. Roberts, and R. Linares, "Space-Based Sensor Tasking Using Deep Reinforcement Learning," *The Journal of the Astronautical Sciences*, vol. 69, no. 6, pp. 1855–1892, 2022.
- [6] P. M. Siew and R. Linares, "Optimal Tasking of Ground-Based Sensors for Space Situational Awareness Using Deep Reinforcement Learning," *Sensors*, vol. 22, no. 20, art. 7847, 2022.
- [7] P. M. Siew, T. Smith, R. Linares, and R. Ponmalai, "Scalable Multi-Agent Sensor Tasking Using Deep Reinforcement Learning," *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2023.
- [8] J. R. Fletcher and J. E. Kadan, "Autonomous interactive agents for global telescope network orchestration," *Proc. SPIE 13101, Software and Cyberinfrastructure for Astronomy VIII*, 131012U, 2024.
- [9] M. A. Hady, S. Hu, M. Pratama, J. Cao, and R. Kowalczyk, "Multi-Agent Reinforcement Learning for Autonomous Multi-Satellite Earth Observation: A Realistic Case Study," *arXiv:2506.15207*, 2025.
- [10] Y. Bengio, A. Lodi, and A. Prouvost, "Machine Learning for Combinatorial Optimization: a Methodological Tour d'Horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.
- [11] C. F. Hayes, R. Rădulescu, E. Bargiacchi, et al., "A Practical Guide to Multi-Objective Reinforcement Learning and Planning," *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, art. 26, 2022.
- [12] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A Survey of Multi-Objective Sequential Decision-Making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.
- [13] R. Rădulescu, P. Mannion, D. M. Roijers, and A. Nowé, "Multi-Objective Multi-Agent Decision Making: A Utility-Based Analysis and Survey," *Autonomous Agents and Multi-Agent Systems*, vol. 34, no. 1, art. 10, 2020.
- [14] H. Hersbach, B. Bell, P. Berrisford, et al., "The ERA5 global reanalysis," *Quarterly Journal of the Royal Meteorological Society*, vol. 146, no. 730, pp. 1999–2049, 2020.
- [15] F. A. Oliehock and C. Amato, *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems, Springer, 2016.
- [16] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe Reinforcement Learning via Shielding," *Proc. AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, pp. 2669–2678, 2018.
- [17] J. García and F. Fernández, "A Comprehensive Survey on Safe Reinforcement Learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.
- [18] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 24611–24624, 2022.
- [19] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments," *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 6379–6390, 2017.
- [20] S. Nayak, K. Choi, W. Ding, S. Dolan, K. Gopalakrishnan, and H. Balakrishnan, "Scalable Multi-Agent Reinforcement Learning through Intelligent Information Aggregation," *Proc. International Conference on Machine Learning (ICML)*, PMLR vol. 202, pp. 25817–25833, 2023.
- [21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.
- [22] W. Kool, H. van Hoof, and M. Welling, "Attention, Learn to Solve Routing Problems!," *International Conference on Learning Representations (ICLR)*, 2019.
- [23] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer Networks," *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2692–2700, 2015.

- [24] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual Multi-Agent Policy Gradients," Proc. AAAI Conference on Artificial Intelligence, vol. 32, no. 1, pp. 2974–2982, 2018.
- [25] P. Sunehag, G. Lever, A. Gruslys, et al., "Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward," Proc. International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), pp. 2085–2087, 2018.
- [26] T. Rashid, M. Samvelyan, C. Schroeder de Witt, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning," Proc. International Conference on Machine Learning (ICML), PMLR vol. 80, pp. 4295–4304, 2018.
- [27] A. Ramesh and B. Ravindran, "Physics-Informed Model-Based Reinforcement Learning," Proc. Learning for Dynamics and Control Conference (L4DC), PMLR vol. 211, pp. 26–37, 2023.
- [28] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse control tasks through world models," Nature, vol. 640, no. 8059, pp. 647–653, 2025.
- [29] R. Yang, X. Sun, and K. Narasimhan, "A Generalized Algorithm for Multi-Objective Reinforcement Learning and Policy Adaptation," Advances in Neural Information Processing Systems (NeurIPS), pp. 14610–14621, 2019.