

Resource-Aware Embodied Autonomy for Space Robots: A Review

Jia Deng, Zizhe Wang, Xin Lu, and Zehao Jia

Astroup Lab¹ | astroup.com

{dengjia, wangzizhe, luxin, jiazehao}@astroup.com

Abstract—Space robots make decisions under closely linked limits on motion, power, temperature, computing, and communication. A maneuver changes both vehicle dynamics and resource demand; a larger perception model can improve sensing while increasing processing time and heat. This structured narrative survey examines those connections through resource management, motion control, and task planning. It compares robotic platforms, onboard computing, thermal prediction and control, resource scheduling, and autonomy demonstrations, distinguishing component experiments from operational uses. Three findings guide the analysis. Lower processor power can come with higher energy per inference when throughput falls. A thermal forecast is useful for task selection only when its error is considered alongside the available temperature margin. Slow supervisory reasoning can help with recurring thermal conditions, but poorly timed updates can degrade control. Existing mission planners provide a basis for integrating these capabilities through resource checks and plan revision. Shared resource balances connect task choices to electrical loads, heat, and motion limits; an illustrative capture scenario uses them to compare immediate, reduced-load, and deferred operation. The review also examines verification, fallback, and changes to models and agent configurations. The resulting priorities are to measure costs across the sensing-to-action chain, test forecasts near operating limits, and match decision timescales to the physical process being controlled.

Index Terms—Embodied intelligence, space robotics, spacecraft autonomy, thermal control, energy management, onboard computing, system integration.

I. INTRODUCTION

AUTONOMOUS spacecraft must make decisions that their physical systems can carry out. A collision-free trajectory can still be unusable because a joint is too cold, the battery cannot supply the peak load, or attitude control has too little authority. A more accurate perception model may also take longer to run and produce more heat. These connections motivate an embodied view of intelligence: behavior grows out of the interaction between an agent, its physical body, and the environment [1], [2].

Recent robot-learning methods connect perception, language, and action through shared datasets [3] and vision-language-action models [4]–[6]. In space, their use depends on radiation exposure, illumination, maintenance access, communication, and the coupling between robot motion and spacecraft attitude. These conditions shape both the hardware design and the tasks that can be completed with the available resources [7].

This survey uses *embodied intelligence for space* to describe autonomous decision-making that uses feedback about

the environment, the physical platform, and its resources to select and revise actions. The scope includes orbiting vehicles, station-supported robots, and planetary surface systems, using either model-based or learned methods.

A. Embodiment, Autonomy, and Learning

Embodiment concerns how sensing, structure, and dynamics shape behavior. Autonomy describes which decisions a system can make without immediate human direction. Learning estimates models or policies from data, while adaptation changes behavior as conditions change. These properties can appear in different combinations: conventional controllers use physical feedback, and a trained neural controller can operate with fixed parameters throughout a mission.

Spacecraft already coordinate control, power regulation, and fault protection. Remote Agent combined planning, execution, and diagnosis [8], with selected functions demonstrated on Deep Space 1 in 1999 [9]. More recently, the Mars 2020 onboard planner has brought activity scheduling, preheating, and wake/sleep management into routine rover operations [10]. This heritage provides a practical starting point for assessing what learning adds to resource-aware autonomy.

The division between ground and onboard decisions follows the task’s response time. Contact and navigation hazards often need a local response, whereas longer-term objectives can be revised during communication windows. Link availability, light-time, computing capacity, and the consequences of a delayed decision all influence that division.

B. Scope and Approach

This structured narrative survey is organized around three questions: how hardware limits enter autonomous decisions, what thermal and energy models contribute to those decisions, and how integrated behavior has been tested. Cases are selected for their relevance to these questions and for the availability of identifiable methods, operating conditions, or mission records. Component studies, such as image filtering and thermal prediction, explain the capabilities and costs available to a larger system. Operational cases show how decisions are executed, checked, and revised. The selection includes established planning and control alongside learning-based methods, and follows directly related studies where later work changes the evidence.

Research papers supply methods and experimental results; agency reports and mission records establish implementation

¹Astroup Lab: 向上跃升（北京）科技有限公司.

and operational use; standards describe engineering requirements and guidance. Product information is used for device specifications, while program announcements establish stated milestones. Comparisons record both the *execution location* of a method and the *source of its validation data*. A model evaluated on the ground using flight telemetry is therefore distinct from a controller running onboard. Simulation, laboratory tests, bounded flight experiments, and sustained mission use are described in their own terms. Source updates were checked through October 15, 2026.

Existing reviews cover servicing and manipulation [11], [12], learning for guidance and control [13], [14], and satellite autonomy [15]. This paper connects those methods to the hardware and resource conditions that determine whether an activity can proceed. In particular, it examines why lower processor power draw need not mean lower inference energy, how forecast error changes task acceptance, and why the timing of supervisory decisions can matter as much as their content.

C. Organization and Contributions

The paper develops a reference model and example interfaces, compares hardware and resource-management approaches, and connects their operating limits to verification and recovery. Section II introduces the functional organization. Sections III and IV examine hardware, thermal prediction, and energy management. Sections V and VI connect these resources to motion and task planning. Section VII uses resource balances and a capture scenario to show how the interfaces work together. Sections VIII and IX discuss assurance evidence and priorities for further integration.

II. A REFERENCE MODEL FOR INTEGRATED AUTONOMY

Figure 1 groups operational intelligence into three functions. *Resource management* estimates and regulates temperature, electrical power, and stored energy. *Motion control* estimates motion and controls the vehicle and its mechanisms. *Task planning* selects activities, schedules them, and monitors their completion. The shared physical platform includes structures, mechanisms, sensors, processors, and power and thermal hardware. This organization makes it easier to follow how a task becomes a feasible physical action.

A. Operation and Development

A planner depends on resource forecasts and motion controllers to carry out an accepted task. These functions can share a computer or be distributed across devices. Their update rates follow the physics: electrical protection can respond rapidly, motion control tracks vehicle dynamics, and thermal forecasts support decisions over longer horizons.

Design and verification support all parts of the system. Thermal models help size radiators and place components; simulation supports controller development; hardware tests reveal model errors and operating limits. Training a policy belongs to this development process, while evaluating the policy to drive an actuator belongs to operation. Test results

and returned telemetry improve later models and software releases. Changes made during operation need additional controls, discussed in Section VIII.

B. Interfaces and Command Authority

An interface carries a value together with its units, timestamp, validity period, and uncertainty. For resource management, two messages are especially useful: the power that can be supplied now and the energy available over the planned activity. Temperature forecasts add the time for which a workload can continue before a limit is reached. Their uncertainty determines how much margin is needed before accepting that workload, as discussed in Section VII. Table I shows how these messages inform requests and responses.

Command authority matters when constraints conflict. Motion control enforces collision and actuator limits; electrical and thermal protection preserve essential functions. The planner selects among activities that fit those limits. During a close approach, for example, navigation and controlled retreat may need to continue while optional image processing stops. The same resource shortage can therefore lead to different actions at different stages of a task.

Hardware and software are designed together. Physical limits constrain the available actions, while the intended workload determines processor capacity, actuator authority, and thermal design. The reference model exposes those shared assumptions; Sections VII and VIII show how they enter resource checks and verification.

III. HARDWARE FOR SPACE OPERATIONS

Hardware determines what a space robot can sense, compute, and do, and how long it can keep working. An arm inside a pressurized station, an orbital servicer, and a lunar rover face different environments and operating demands. This section connects those demands to morphology, sensing, actuation, computing, and energy storage, with particular attention to the costs that influence onboard decisions.

A. The Space Environment as a Design Driver

Environmental requirements follow from the orbit or landing site, mission duration, operating modes, shielding, and acceptable risk. ECSS-E-ST-10-04 provides a framework for specifying these conditions [16]. Table II summarizes the main environmental distinctions.

Radiation affects electronics through total ionizing dose (TID), displacement damage, and single-event effects (SEE). Exposure depends on the orbit, shielding, and mission duration, while component technology determines how that exposure affects operation [17]. Comparing low Earth orbit (LEO) with geostationary orbit (GEO) therefore requires a mission-specific radiation model. Solar cells, detectors, memories, and power devices also respond differently, leading to different tests and protection measures.

External vacuum removes convective heat transfer to the surroundings. Heat moves through solid conduction, radiation, and internal fluid transport; forced convection is also available

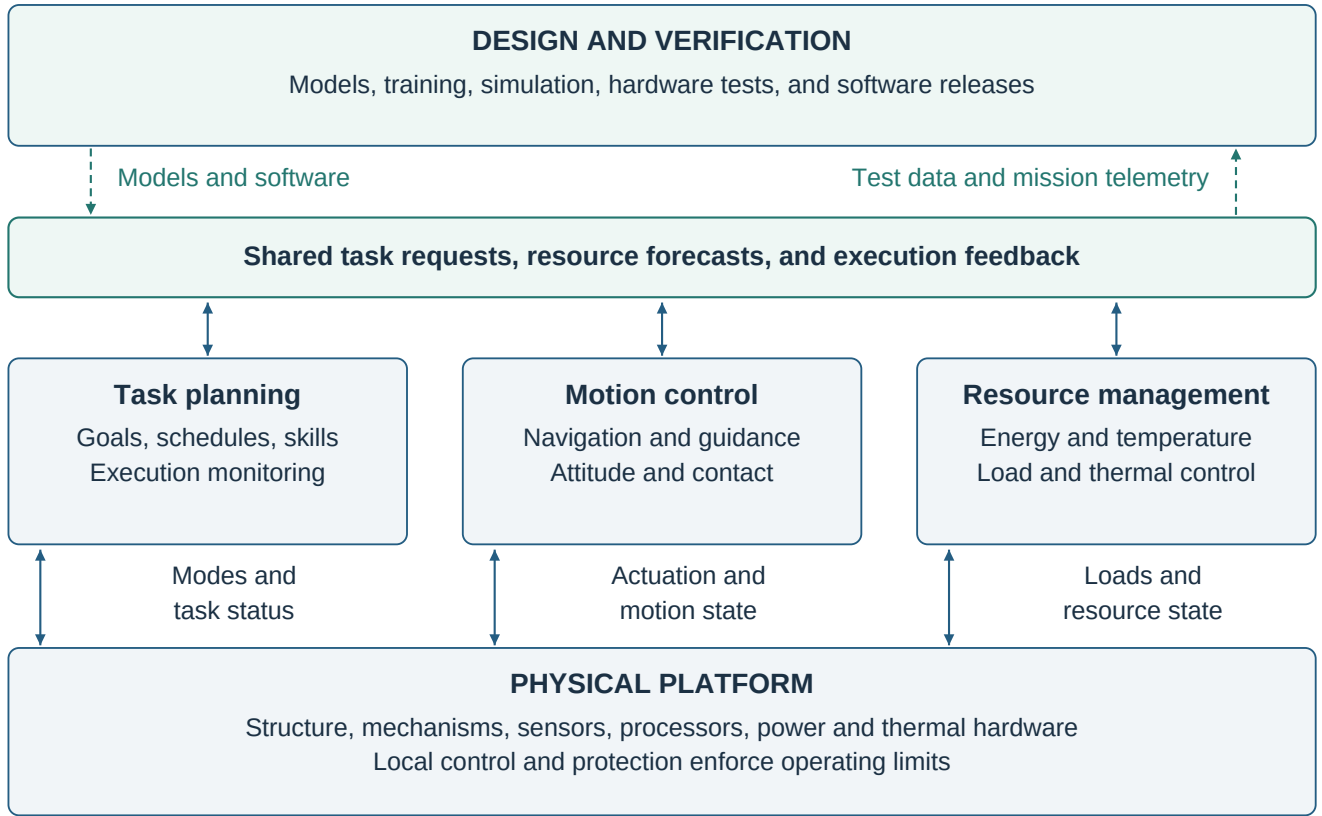


Fig. 1. Functional organization of embodied space autonomy. The three operational functions exchange task requests, resource forecasts, and execution feedback. The physical platform supplies sensing and actuation; design and verification use test and mission data to improve the models and software.

TABLE I
EXAMPLE EXCHANGES BETWEEN OPERATIONAL FUNCTIONS. LIMITS AND UPDATE PERIODS FOLLOW THE MISSION DYNAMICS AND HARDWARE.

Sender → receiver	Message	Receiver action	Response to a problem
Planner → resource manager	Proposed activity, duration, timing options, sensor and compute demand	Forecast loads and temperatures; return a feasible time window and budget	Reject or suggest a shorter or later activity
Resource manager → planner	Stored energy (J), available power (W), temperatures (K), forecast validity and uncertainty	Admit, defer, or reschedule activities	Preserve recovery reserves and request a revised plan
Planner → motion controller	Target state or trajectory, deadline, resource budget, permitted alternatives	Check and execute the feasible motion	Refuse an infeasible request; return the limiting condition
Motion controller → planner and resource manager	Tracking state, torque (N m), wheel momentum (N m s), contact state, revised load estimate	Update task status and resource forecasts	Adjust timing or request an approved hold or retreat
Sensors → estimators and protection	Time-stamped motion, temperature, current, voltage, and health measurements	Update state estimates and enforce local limits	Isolate inconsistent data; trigger the appropriate fallback

inside pressurized compartments [18], [19]. Illumination cycles change the thermal boundary conditions, while mounting, insulation, dissipation, and thermal inertia shape equipment temperatures. On the Moon, the sunrise-to-sunrise period is about 708.7 hours. At low latitudes, daylight and darkness each last roughly half that period, with local terrain and polar illumination changing the exposure [20]. Activity schedules consequently depend on the energy and heater capacity available through the dark period.

Vacuum compatibility begins with material screening. ECSS-Q-ST-70-02C specifies recovered mass loss (RML) below 1.0% and collected volatile condensable material (CVCN) below 0.10% as minimum screening criteria, with stricter limits possible near sensitive surfaces [21]. RML excludes reabsorbed water, which distinguishes it from total mass loss. The choice of adhesives, lubricants, and flexible sensors also depends on their installed configuration and operating conditions.

TABLE II
MISSION ENVIRONMENTS AND THEIR IMPLICATIONS FOR EMBODIED OPERATION

Mission setting	Main environmental drivers	Information needed for design	Consequence for autonomy
Exposed LEO systems	Orbit-dependent radiation, illumination and eclipse, atomic oxygen, and debris [16].	Altitude, inclination, mission life, shielding, attitude, surface materials, and thermal interfaces.	Plan within changing power and temperature limits; detect sensor or processor faults.
GEO systems	Energetic electrons, surface and internal charging, seasonal eclipses, and radiation exposure [16].	Dose and charging analysis for the actual geometry, shielding, and mission duration.	Include recovery after electrical faults and loss of sensing; protect energy reserves during eclipses.
Lunar surface	Long illumination and darkness intervals, vacuum, radiation, dust, and terrain-dependent shadow [20], [22].	Landing-site illumination, component survival limits, dust exposure, and energy storage.	Coordinate travel, observation, heating, and sleep; preserve a feasible survival plan.
Interplanetary missions	Solar flux varies with distance; solar particles, cosmic rays, and planetary radiation belts vary along the trajectory [16].	Trajectory, planetary encounters, communication schedule, shielding, and thermal operating modes.	Handle long periods without ground intervention while respecting local hardware limits.
Pressurized interiors	Controlled atmosphere and ventilation, crew interaction, enclosed geometry, and station safety requirements [23].	Cabin airflow, contact hazards, permitted workspaces, charging facilities, and access for maintenance.	Use collision avoidance and bounded contact forces; account for people and available supervision.

B. Morphology and Operational Context

Morphology can be described through three practical questions: how the base moves, how the robot contacts its surroundings, and where it operates. ETS-VII illustrates the dynamic coupling between a spacecraft and its mounted arm [24], while GITAI S1 demonstrates dexterous manipulation inside a pressurized airlock [25]. Humanoid shape, dual-arm geometry, and modularity add further design choices within these operating arrangements.

Table III groups representative systems by their main control constraints. Station-mounted arms transfer reaction loads into a large supporting structure. Spacecraft-mounted arms move a smaller carrier and use its available attitude authority. Surface robots depend on traction and terrain contact, while relocatable arms change support points and verify each new attachment. These differences shape the state estimates, training dynamics, and motion limits used by a controller.

The progression from GITAI S1's internal assembly and panel operations in 2021 to S2's external assembly and manipulation in 2024 illustrates how changing the operating environment changes the demands on a robot [25], [26]. Each demonstration establishes experience with a particular configuration and task; the algorithms and interfaces used in a new task determine how much of that experience carries over.

C. Actuation and Mechanisms

Actuation exposes physical limits directly to control: available torque or thrust, speed, temperature-dependent duty cycles, backlash, friction, and accumulated use. Space manipulators combine geared joints, end effectors, and compliant control in different ways [11]. Force/torque feedback helps regulate contact, while grasp stability also depends on geometry, friction, and the motion of both bodies.

Lubrication, wear, and heat transfer determine how these capabilities change over a mechanism's life. A one-shot release device and a repeatedly used joint have different operating

profiles and qualification needs [39]. For learned control, measured friction, saturation, and actuator response provide the link between simulation and hardware. LeLaR, for example, brings a learned attitude controller into operation with real reaction-wheel dynamics [40]. At the hardware interface, speed, current, and temperature limits define the available control authority.

D. Embodied Sensing

Sensors describe motion, surroundings, and internal condition. Encoders and inertial sensors support motion estimation; cameras, lidar, and radar support navigation and inspection; temperature, current, voltage, and strain measurements reveal equipment condition. Bringing these measurements together lets a planner relate an action to both its physical outcome and its resource cost.

Calibration and sensing conditions matter alongside perception accuracy. The Mars 2020 arm's six-axis force/torque sensor uses temperature compensation to interpret contact loads [36]. Event cameras instead offer asynchronous measurements of brightness changes, useful under rapid motion and strong contrast [41], [42]. Rathinam et al. combine synthetic event-camera data with physical Akida V1 hardware and an Akida V2 cloud simulation for spacecraft pose estimation [43]. The benchmark uses ground-truth bounding boxes to crop the network inputs; a Perspective- n -Point (PnP) solver then recovers pose from the predicted keypoints. V1 quantization reduces pose accuracy, while the different V2 network improves it. Table IV compares V1's power and throughput with a Jetson baseline.

E. Onboard Computing and Fault Tolerance

The useful computing capacity of a spacecraft is the work it can finish, check, and sustain within its power and thermal limits. Radiation-hardened processors such as GR740 support

TABLE III
REPRESENTATIVE EMBODIMENTS: DEMONSTRATED CONTEXT AND CONTROL CONSTRAINTS

Operating arrangement	Representative systems	Demonstrated context	Main control constraint
Station-mounted manipulation	Canadarm2, Dextre, and China's space-station arms [27]–[29].	Manipulation on operational stations; selected arms can relocate between attachment points.	Structural flexibility, contact loads, work envelopes, and coordination with the station.
Spacecraft-mounted manipulation	ETS-VII and Orbital Express [24], [30].	Orbital demonstrations of robotic operations and servicing.	Coupled base–arm dynamics, target motion, capture transients, and attitude-control authority.
Intravehicular free flight	Astrobee [23], [31].	Free flight and robotic activities inside the ISS; three robots arrived in 2019.	Air-driven propulsion, localization in a changing cabin, collision avoidance, and docking.
Mounted dexterous manipulation	Robonaut 2 and GITAI S1; external GITAI S2 [25], [26], [32].	Robonaut 2 arrived at the ISS in 2011 [33]; S1 and S2 demonstrated specified tasks in 2021 and 2024.	End-effector compatibility and force limits; internal and external environments require different verification.
Surface mobility and sampling	Perseverance rover and its sampling arm [34]–[36].	Mars surface driving, sample acquisition, and calibrated force/torque sensing.	Terrain contact, wheel slip, arm loads, sensing uncertainty, and daily resource limits.
Relocatable and manufacturing prototypes	MARM and OSAM-2/Archinaut [37], [38].	MARM: ground prototype. OSAM-2: ground demonstrations; project concluded in 2023 before flight.	Reliable attachment, interface alignment, changing geometry, and verification of assembly steps.

general-purpose onboard software [44]. Commercial accelerators support more demanding workloads, as demonstrated by Myriad 2 cloud detection aboard Φ -sat-1 [45], [46]. Radiation-tolerant adaptive devices such as the Versal XQR family combine processors, programmable logic, and acceleration engines [47]. These options suit different combinations of real-time control, image processing, and planning.

Table IV compares four implemented configurations using the tasks and conditions reported in each study. The clearest flight pattern is to give a specialized processor a bounded workload and integrate its output through a host computer. On Φ -sat-1, image preprocessing and latch-up protection support the Myriad inference engine; memory and power budgets led the cloud model to use three selected spectral bands [45]. The value of the accelerator thus comes from the complete sensing and data-handling chain.

The Perseverance case makes the scheduling consequences particularly clear. Its Snapdragon 801 resides in the Ingenuity Helicopter Base Station and exchanges data with the RAD750 over a slow serial link. Transfer can overlap other rover activities, while processor throttling keeps operation below a 90°C limit. After memory faults appeared in flight, the implementation added memory screening and avoided weak locations; repeated localization and host-side integrity and consistency checks govern acceptance of a result [48]. Scheduling this workload therefore means reserving time for transfer, computation, checking, and recovery, as well as providing enough thermal headroom.

Neuromorphic computing illustrates a different exchange. The Loihi 2 radio-resource benchmark saves energy at the cost of longer measured solution time, and remote access affects that timing [49]. For Akida V1 and Jetson, dividing reported average dynamic power by network throughput gives about 54.6 and 7.59 mJ per inference, respectively [43]. Akida's

value is about 7.2 times higher in this configuration: lower power does not translate into lower energy per inference. This calculation covers the reported network benchmark. Comparing complete sensing chains or operation at the same target frame rate would also require preprocessing, PnP, host, and idle-state costs. Processor selection therefore depends on the required accuracy, update rate, and operating schedule as well as power.

Fault protection also operates at several levels. Component selection and shielding address cumulative exposure, while error-correcting codes, redundancy, memory or configuration scrubbing, and restart mechanisms address recoverable upsets [17], [50]. Current limiting and power removal protect against latch-up. Software checks then help detect corrupted calculations and recover useful service. Together, these measures determine how much reliable computation is available to resource management (Section IV).

F. Energy Hardware

Solar arrays, batteries, power conversion, and radioisotope sources set the energy available to a robot [51]. Generation changes with illumination and orientation; usable storage changes with temperature, state of charge, and age. Radioisotope heaters supply heat, while radioisotope power systems convert part of that heat to electricity. These choices set the duration of useful activity and the reserves needed during darkness or faults.

These hardware limits set the power envelope and reserves used by resource management in Section IV.

G. Hardware–Algorithm Co-Design

Hardware–algorithm co-design chooses the body and controller together. Body properties can simplify sensing and

TABLE IV
COMPUTING CONFIGURATIONS: DEMONSTRATED WORKLOADS AND PRACTICAL TRADE-OFFS

Configuration and task	Evaluation setting	Reported result	Main integration cost or trade-off
Myriad 2 / Φ -sat-1: cloud screening [45]	Model benchmarks and in-orbit image processing.	About 102 ms per inference using eight vector processors; flight images supported cloud-screening evaluation.	Image preprocessing, host communication, and electrical protection remain part of the payload; memory and power limit the input bands.
Snapdragon 801 + RAD750: global localization [48]	Perseverance flight demonstrations on Mars.	In the sol 1762 demonstration, position uncertainty fell from 8.4 to 1.5 m; RAD750 accepted the result after host checks.	About 30 min of data transfer at 10 KB/s, partly overlapped with other work; thermal throttling, memory checks, and repeated computation.
Loihi 2: radio-resource management [49]	Ground benchmark through Intel's research cloud; VCK5000 reference.	Reference/Loihi energy ratio above 10^2 , with reference/Loihi solution-time ratio below 1: lower energy and longer measured time.	Input encoding, batch size, cloud interface, and shared access affect the measured timing.
Akida V1: keypoint inference for pose estimation [43]	Synthetic SPADES data; quantized Akida V1 versus floating-point Jetson Orin Nano.	On cropped 2D-histogram inputs, network throughput is 17 versus 349 frames/s; reported dynamic power is 0.928 versus 2.650 W.	V1 quantization reduces pose accuracy; host interfacing limits throughput. At these rates, lower power comes with higher dynamic energy per network inference.

control [1]; in space, they also shape capture loads, recovery options, heat rejection, and the energy needed for each task. Examples include trading sensor placement against perception accuracy, manipulator compliance against contact force, and processor throughput against cooling demand.

MARM provides a concrete example: candidate limb designs are evaluated with motion planning, trajectory optimization, and constrained inverse dynamics, followed by preliminary laboratory validation [37]. The resulting design is judged by the motions it can execute under physical constraints. For learning-based co-design, a fixed-hardware baseline and common task and fault cases provide a useful comparison. Standard attachment, power, and data interfaces make reconfiguration easier, while design and verification connect model predictions to measured operating limits.

IV. THERMAL CONTROL AND ENERGY MANAGEMENT

Resource management keeps the spacecraft within its temperature, power, and energy limits while supporting mission tasks. It combines thermal hardware, state estimation, load scheduling, and independent protection. Fig. 2 shows how these functions work together: forecasts guide the schedule, while fast protection responds directly to unsafe conditions. Learning can improve the forecasts or the choice of operating mode; the installed hardware determines the available heating, cooling, and storage capacity.

A. Where Learning Helps

Spacecraft already manage heat and power through hot/cold-case analysis, operating envelopes, thermostats, and load scheduling [19], [52]. Autonomous operation adds more combinations of perception, communication, maneuvers, and mechanism heating. Better prediction can help use the resulting gaps in a schedule or avoid an expensive peak load. Fixed power caps and duty cycles remain practical choices when demand is predictable and margins are generous. An adaptive

method earns its place by improving temperature forecasts, completed work, or resource use under the same operating limits.

Its role also determines how it is evaluated. A design surrogate speeds up preflight analysis. A ground model helps operators choose a plan. An onboard scheduler adjusts tasks during flight, where computation time, command feasibility, and recovery from errors become part of its performance. These uses share models and data, but place different demands on the software.

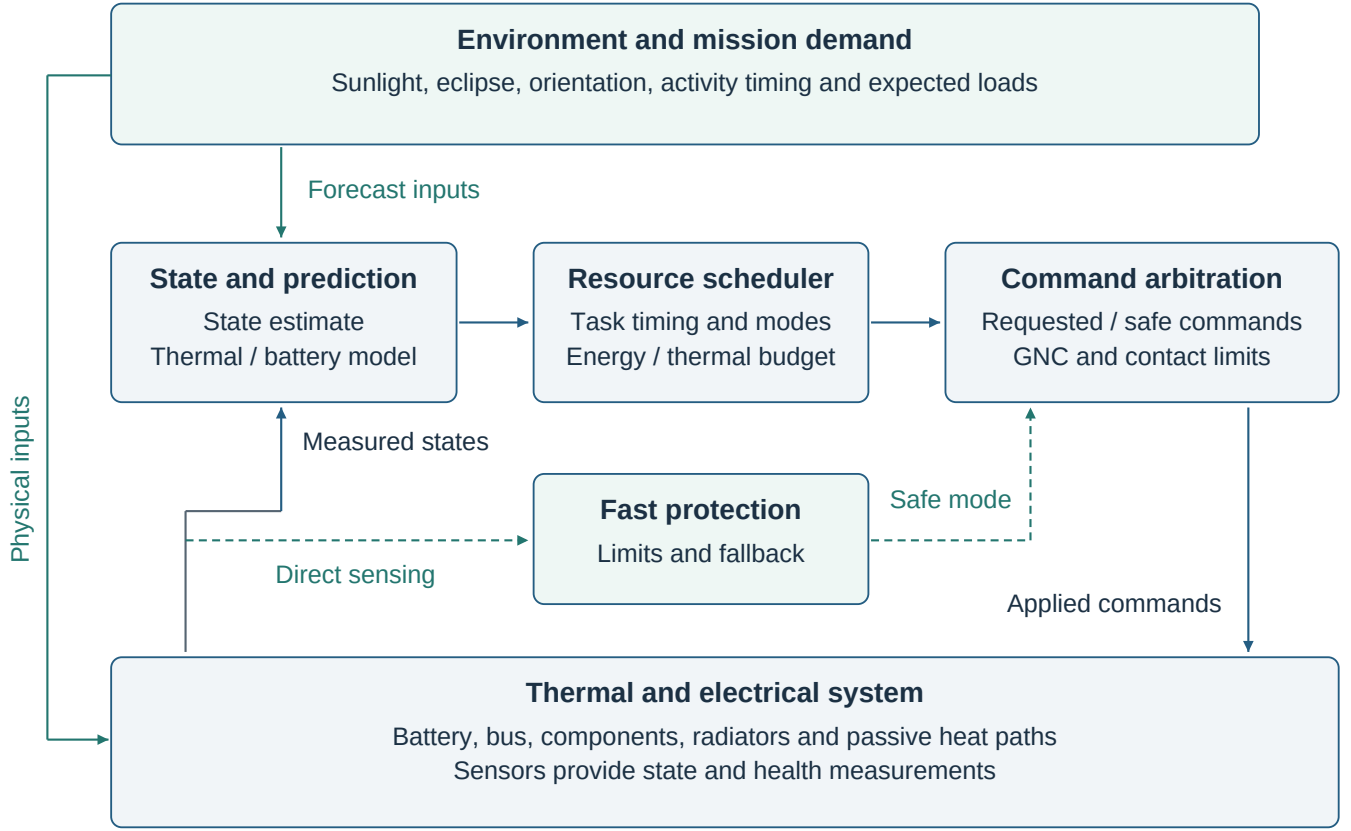
B. Hardware and Environmental Constraints

Coatings, multilayer insulation, radiators, conductive paths, and ordinary heat pipes set the heat-flow network. Heaters, pumps, and motor-driven mechanisms provide commanded inputs. Some louvers respond passively to temperature [18]. Variable-conductance heat pipes can regulate heat flow passively or use reservoir heaters, depending on the design [19], [53]. Software authority therefore depends on the devices actually installed. Surface absorptance, emittance, and their degradation remain important inputs to both physical and learned models [54].

Absorbed sunlight, albedo, planetary infrared radiation, self-shadowing, and radiative view factors vary with orbit and configuration. Thermal inertia determines how quickly the hardware responds. Moving an external arm can change its solar exposure and its conductive path to the bus. A useful operating plan accounts for this history as well as the temperature at the start of a task.

C. Thermal Surrogates: Methods and Uses

Table V compares three research lines. Tanaka and Nagai use proper orthogonal decomposition (POD) with physics-informed learning for steady-state temperature fields [55]. Yamashita et al. add recurrent networks to represent transient behavior [56]. Jang et al. progress from a reduced model



Solid: forecasting and control. Dashed: independent protection.

Fig. 2. Thermal and electrical resource management. Forecasts support task and load scheduling. A separate protection path uses measurements directly and can select an approved safe command without waiting for the scheduler.

checked against thermal-vacuum measurements to a ground prediction model refined with actual orbital temperatures [57], [58]. Together, these studies address design sweeps, temperature histories, and correction of simulation-to-flight differences.

The choice follows the question being asked. A steady-state surrogate is useful for scanning heat loads or design parameters. A recurrent model is more suitable when the timing of loads and thermal inertia matter. Transfer learning is useful when telemetry reveals a persistent difference between simulation and flight. In the Jang work, active learning selects informative simulation cases, while transfer learning adapts the model to orbital measurements [58].

For engineering use, an error in kelvin is easiest to interpret alongside the component's available temperature margin. A field-wide L_2 error summarizes reconstruction quality, whereas the largest local error and the timing of a temperature peak matter when choosing a load schedule. Likewise, training cost describes model preparation and prediction cost describes repeated use. Comparisons need the same forecast interval, output locations, and computing platform. Method choice therefore rests on the prediction task and available data, with timing and local error checked for the intended application.

D. From Forecasts to Thermal Control

Model predictive control (MPC) uses a forecast to schedule heating and loads over a future interval. A learned component can improve the thermal model, estimate uncertain parameters, or help select a policy [59]. Mechanism preheating before a planned operation and deferring optional processing during a hot period are natural applications. The scheduler operates within the heat-rejection capacity of the installed radiators and the commands available to it.

Direct payload experiments show how learning can control the heat generated by computation. Mousist's ground study uses Soft Actor-Critic (SAC) on a 16-core ARM board to choose the number of active cores and their frequency, with temperature margin and rate of change as feedback [60]. The follow-up ASTREA study reports experiments on an external ISS payload in September 2025 [61]. Its RL controller manages 15 loaded cores, while a quantized language model asynchronously adjusts SAC's entropy coefficient. The reported outcome depends strongly on the supervisory timescale: a 15-minute analysis window produces more temperature-limit violations, shorter episodes, and lower CPU utilization than the RL-only baseline. A 90-minute window aligned with the orbit improves all three measures. These are controlled computing-

TABLE V
SPACECRAFT THERMAL SURROGATES: PREDICTION TASKS AND PRACTICAL USES

Study and method	Prediction task and data	Reported comparison	Use supported by the study
Tanaka and Nagai [55]; POD with physics-informed learning	Steady-state fields for two thermal mathematical models; varied heat inputs and training data	Full-model temperatures are the reference. Total training cost falls by 26–81% relative to data-driven learning in the reported cases.	Repeated steady-state design cases when a heat-balance model is available.
Yamashita et al. [56]; POD with recurrent networks	Full-system transient temperature histories from thermal-analysis sequences	Reported average temperature error: 1.53 K. Prediction cost: 0.23% of the reference thermal mathematical model (TMM) analysis.	Fast temperature-history prediction across the studied analysis cases.
Jang et al. [57], [58]; PCA and neural networks, followed by active and transfer learning	2024: simulations varying 20 thermal design parameters, checked against hot/cold thermal-vacuum tests. 2025: simulation training plus orbital temperature data.	2024 uses L_2 -norm errors for reduced-model assessment and measured device temperatures for comparison. The 2025 model refines temperature histories with flight data; predictions take seconds.	Ground design and operational temperature prediction, including correction toward measured flight behavior.

POD and principal component analysis (PCA) reduce the temperature field to a small set of modes. The reported training cost, prediction cost, and temperature errors measure different quantities. These studies evaluate temperature prediction; the Jang flight-data model runs as a ground prediction tool.

load experiments; Thales Alenia Space also lists the in-orbit thermal-management experiment in its IMAGIN-e operational record [62]. The contrast between the two windows shows why a supervisor’s information window and response delay must fit the thermal process it influences.

Attitude-based thermal management has a long history: Apollo used slow spacecraft rotation for passive thermal control [63]. Modern planning can coordinate orientation with joint motion and electrical loads. An orientation that reduces absorbed heat may compete with antenna pointing, solar-array illumination, collision avoidance, or camera visibility. Reorientation also uses wheel or thruster resources. Resource management therefore requests an orientation through motion control, which checks its feasibility with the rest of the maneuver.

E. Energy-Aware Scheduling and Computing

Energy estimates connect the physical power system to the task schedule. State of charge describes stored charge relative to capacity; state of health (SoH) describes degradation; remaining useful life forecasts future aging under assumed use. Battery temperature and discharge rate affect both available energy and allowable current. Coulomb counting and equivalent-circuit estimation provide established baselines [51], while Gaussian-process prognostics add uncertainty estimates to degradation forecasts [64]. The planner needs a usable power limit and energy reserve, together with the conditions over which they apply. Telemetry anomaly detection can support this process by directing attention to unusual behavior [65], [66].

Several spacecraft studies already connect resource models to scheduling. Doff-Sotta et al. formulate a convex MPC problem that allocates power to secondary tasks while meeting thermal and propulsion demand, using a simulation case based on Eutelsat 172B [67]. The motion is prescribed, which makes the resource allocation easier to solve. GomX-3 uses

a different route: a ground scheduler generates plans with a simple battery model, then checks depletion risk using a more detailed stochastic model before upload [68]. M2020 brings activity and heating schedules onboard [10], as discussed in Section VI. Table VI compares their decisions and operating settings.

Task granularity, available computing time, and feedback frequency determine which scheduling method fits. A plan with fixed trajectories can use detailed power forecasts; a contact task needs to revise them as motion and perception change. Computing is part of that load. Workload admission, processor selection, clock control, and task timing can reduce peaks or move processing to cooler periods [7], [69]. The relevant measurements are energy per completed operation, execution time, peak current, and component temperature, including memory and conversion losses.

F. Survival and Onboard Use

Lunar-night operation makes the link between hardware and scheduling clear. CNSA reports that the Chang’e-4 lander and Yutu-2 rover survived their first lunar night using radioisotope heat sources and autonomous wake-up as illumination returned [70]. Hibernation conserved resources while the thermal hardware supported survival. A good autonomous plan includes such low-activity modes as well as productive ones.

Thermal surrogates, resource schedulers, and learned payload controllers now address different parts of this operating problem. The ground and ISS experiments establish direct learning-based control of computing loads, while rover scheduling shows how heating can support daily activities. The next step is to connect these capabilities: a processor controller should know which jobs can pause and which must continue, and a task planner should receive revised duration and energy estimates when computing is throttled. That connection lets thermal control preserve useful work as well as temperature margin.

TABLE VI
RESOURCE SCHEDULING: DECISIONS, EXECUTION, AND METHOD SELECTION

Method	Decisions and resource model	Execution and feedback	Suitable use and main tradeoff
Doff-Sotta et al. [67]; convex MPC	Power allocation and heating with battery, thermal, and propulsion models; prescribed motion	Numerical study based on Eutelsat 172B. Receding-horizon updates address model mismatch.	Continuous allocation with known motion; jointly choosing trajectories changes the optimization structure.
GomX-3 [68]; timed automata and battery validation	Payload activity schedule; linear battery model for planning and stochastic kinetic model for risk evaluation	Plans generated on the ground and executed in orbit. Current and voltage measurements inform subsequent planning.	Discrete payload plans prepared ahead of execution; richer battery checks add validation cost.
M2020 [10]; priority-based scheduling	Activity placement with preheating, maintenance heating, and wake/sleep constraints	In onboard operation since October 2023. Repanning responds to earlier or later activity completion.	Rapid feasible schedules with ground-tuned priorities; placed mission activities are retained during each scheduling pass.

V. MOTION INTELLIGENCE AND ITS INTERFACES

Guidance, navigation, and control (GNC) connect a task request to physical motion. Navigation estimates the vehicle or target state; guidance specifies a feasible motion; control generates actuator commands. Fault detection, isolation, and recovery (FDIR) supports these functions and also spans power, thermal, computing, and communication subsystems. Learning can contribute to each stage, with different demands on accuracy, response time, and recovery [13], [14].

A. Learning in Navigation and Control

Neural pose estimators infer a target spacecraft's relative position and attitude from images [71]. Their errors depend on illumination, target appearance, and the difference between training images and deployment conditions. SPEED+ examines this transfer by combining synthetic training imagery with images of a physical mockup under controlled lighting [72]. For the navigation loop, the next step is to assess how estimation errors affect the commanded motion. Occasional large errors and delayed estimates can matter more than average image-level accuracy.

Reinforcement learning has also been studied for proximity guidance. Hovell and Ulrich train a policy to generate velocity commands for a conventional tracking controller, then evaluate it in simulation and ground experiments [73]. This division lets learning select the motion while the tracking controller handles execution. A flight implementation also needs to account for keep-out regions, fuel, actuator limits, and navigation errors. Learning-based model predictive control offers another route: a learned dynamics or disturbance model improves predictions while a constrained optimizer selects the action [59]. Its practical value depends on both prediction accuracy and the time available to solve the control problem.

LeLaR demonstrated learned inertial pointing on InnoCube, with reported steady-state attitude errors below 1° on each axis [40]. Network inference took about 38 ms within a 1 Hz control loop. Experiments were limited to 15 min for power and safety reasons. An independent Safety Cage monitored motion and actuator limits; its fallback was configured to disable control and later stop the reaction wheels. That choice

relied on InnoCube remaining power-positive while tumbling, with solar panels on its long sides. The reported flights completed pointing maneuvers without triggering this protection. In this case, inference fitted the control deadline, while power balance set the experiment duration and recovery mode.

APIARY tested a learned six-degree-of-freedom controller on Astrobee inside the ISS in May 2025 [74]. About seven minutes covered eight maneuvers, including two redocking attempts. The first exceeded a position-error threshold, triggering a return to the baseline controller to hold position; an operator initiated the successful second attempt. The learned controller completed basic maneuvers, although tracking in the illustrated undocking maneuver was poorer than the study's ideal simulated baseline. The main result was successful integration with existing flight software and recovery behavior within Astrobee's spare computing capacity; improving tracking remained a further step.

Telemetry-based anomaly detection contributes at a different point in the loop. Hundman et al. evaluate a learned detector on spacecraft telemetry [65]. Such a detector can flag unusual behavior for diagnosis. Selecting a recovery action then requires the subsystem state, the likely cause, and a set of tested responses.

B. From a Task Request to an Executable Motion

A motion interface accepts a target state or trajectory together with timing and resource limits. It returns the achieved state, tracking error, estimated uncertainty, and whether the request remains feasible. For manipulation, it also needs the base-motion assumptions, permissible contact loads, and stopping or retreat conditions. Free-floating and station-attached arms require different models even when their joint arrangements look similar [24], [75].

Resource effects belong in this exchange. A longer slew may reduce peak torque but change the illumination history; an arm motion can increase both electrical load and attitude-control demand. The planner needs these forecasts before accepting a task. During execution, motion control checks incoming commands and reports changes that require a new plan.

High-level decisions therefore reach the actuators through the mission’s tested motion interface.

Evaluation should combine task success with tracking error, constraint violations, latency, actuator saturation, and energy use. Comparing a learned method with its conventional baseline under the same sensing, actuation, and disturbance conditions makes the tradeoffs visible: better accuracy may require more computation, and lower energy use may come with slower motion.

VI. TASK PLANNING AND OPERATIONAL AUTONOMY

Task planning turns mission goals into activities, follows their execution, and revises the schedule when conditions change. Resource allocation and recovery are central to this process. Remote Agent demonstrated goal-based planning and fault handling for selected Deep Space 1 subsystems [8], [9]; rover autonomy provides further examples of onboard decisions under ground supervision [34]. This heritage offers a useful starting point for connecting newer learning methods to spacecraft operations.

A. Planning with Resource Constraints

The Autonomous Sciencecraft Experiment on Earth Observing 1 combined science-data analysis, CASPER planning, and the Spacecraft Command Language execution system [76]. CASPER repaired plans as goals and conditions changed, checking power, storage, timing, and other spacecraft constraints. Before issuing commands, the execution system checked available power and battery energy again. Computing capacity also shaped the design: about 4 of the processor’s 8 million instructions per second were available to the autonomy software. Hierarchical planning and incremental updates kept the workload manageable. This is a concrete example of resource-aware decisions running within a small onboard computing budget.

Perseverance’s onboard scheduler, used for primary operations since October 2023, extends this approach to daily rover activities [10]. It places activities in priority order while accounting for preheating, maintenance heating, and wake/sleep periods. Apart from those supporting activities, an activity’s placement is retained during a scheduling pass. The ground Copilot system uses Monte Carlo analysis to tune priorities and other parameters before uplink. This division combines a compact onboard scheduler with more extensive ground analysis and lets the rover adjust to activities finishing earlier or later than expected.

Each method fits a different kind of planning problem. CASPER repairs an existing plan when changed conditions make it inconsistent, while M2020’s priority-based insertion produces a feasible schedule with limited onboard search. Language and visual models can propose activities from instructions or new observations; a planner or scheduler can then check their resource demands and timing before execution. The choice depends on whether the task needs plan repair, rapid scheduling, or interpretation of unfamiliar inputs. Comparisons should measure completed science, resource margins, replanning time, and operator workload.

B. Foundation Models and Skill-Based Execution

Language and vision-language models produce different kinds of outputs. In terrestrial robotics, SayCan selects executable skills by combining language-based task relevance with estimates of what each skill can achieve [77]. RT-2 and OpenVLA map visual and language inputs to robot actions [4], [5]. These outputs need different interfaces: skill selection requires an execution routine, while action outputs need checks at the command interface.

SpaceMind uses “skills” as natural-language knowledge modules assembled into the model prompt; its actions are tool calls for operations such as positioning or camera adjustment [78]. It evaluates this architecture in UE5 simulation and on a Mecanum-wheeled myAGV Pro moving in a three-degree-of-freedom plane, using Qwen3-VL-235B-A22B throughout. The results show reuse of visual decision-making and tool interfaces across the two environments. The UE5 simulation excludes orbital dynamics, and the tests do not measure processor load, power demand, or inference latency on a spacecraft computer. Its experience-based updates change prompt knowledge while keeping model weights fixed, making those knowledge modules part of the configuration managed in Section VIII.

A mission design should therefore specify where the model runs and how long each decision may take. A ground-based model can prepare plans for uplink when communications permit. An onboard model can select among tested skills at a slower rate than motion control, with a fixed response deadline. If that deadline is missed, the execution system continues the current permitted step or enters a defined hold state. The choice depends on the task and available resources.

Each executable skill needs clear preconditions, expected effects, resource demand, and stopping conditions. A capture skill, for example, requires adequate navigation confidence, acceptable relative motion, sufficient control authority, and enough thermal–energy margin to complete or retreat. Motion control and resource management provide these conditions; task planning selects among the feasible alternatives.

C. Feedback, Ground Support, and Recovery

Sensor feedback must distinguish success, partial completion, and failure. The planner can then retry a permitted step, choose another skill, or request operator assistance. Retry limits matter because repeated attempts consume energy, mechanism life, and communication opportunities. APIARY’s operator-initiated second docking attempt illustrates how automatic recovery and human decisions can work together [74].

Ground teams set goals and inspect unusual results when the communication schedule permits, while local software handles events that need an immediate response. The allocation follows the task’s timing, computing resources, and consequences of failure. For example, optional image analysis can wait during a thermal excursion, whereas contact motion may depend on uninterrupted pose estimation. Encoding these dependencies in mission rules and skill definitions connects planning to the resource constraints in Section VII and the recovery tests in Section VIII.

VII. CROSS-DOMAIN COUPLING

Physical action, computing, and resource use are closely linked. Arm motion changes base motion [75], [79], [80], electrical loads produce heat, and communication gaps determine how long a spacecraft acts on its own. Task planning brings these relationships together because a robot's next action changes both its workload and its physical state. Fig. 3 shows the information exchanged between task planning, resource management, and motion control.

A. Couplings That Change a Task Decision

Energy, temperature, and computing: processing draws electrical power and releases heat. Battery limits constrain short bursts, while thermal storage and heat rejection constrain their duration and repetition. Eclipse and orientation change generation and thermal conditions. Temperature in turn affects battery power limits and solar-cell output [19], [51], [69]. A useful schedule tracks this history across successive tasks.

Manipulation and attitude: arm trajectories exchange momentum with a free-floating base, and capture changes the combined inertia [75], [79]. Coupled-dynamics control provides the motion model. Task planning chooses when to move, which configuration to use, and when to hold or retreat. These choices affect wheel-momentum margin, motor heating, and available camera views.

Temperature and pointing: orientation and articulation change absorbed radiation and radiator exposure, while thermal deformation can change alignment [19], [52]. A thermal maneuver therefore shares pointing and collision constraints with other actions.

Communication and autonomy: outages, contact windows, propagation delay, and ground response time determine how long flight software must operate before receiving a new command [23], [34]. Plans need enough local information and recovery options to cover that interval.

B. Resource Balances and Task Interfaces

A compact shared model can connect these decisions while keeping each resource in its own units. Let E be stored battery energy in joules, T_i component temperatures in kelvin, and h_j wheel angular momenta in N m s. For a simple battery model,

$$\dot{E} = \eta_{\text{ch}} P_{\text{ch}} - P_{\text{dis}} / \eta_{\text{dis}}, \quad (1)$$

where charge and discharge powers are measured at the bus, and the efficiencies account for storage-path losses. The bus balance is

$$P_{\text{gen}} + P_{\text{dis}} = P_{\text{load}} + P_{\text{ch}} + P_{\text{curt}}, \quad (2)$$

where P_{gen} is available generation and P_{curt} is curtailed power. Loads include avionics, computing, heaters, mechanisms, communications, and electric propulsion where present. Propellant is tracked as a separate resource.

A lumped thermal balance is

$$C_i \dot{T}_i = Q_{i,\text{env}} + Q_{i,\text{diss}} + Q_{i,\text{heater}} + \sum_j G_{ij}(T_j - T_i) - Q_{i,\text{rad}}, \quad (3)$$

with thermal capacitance C_i in J/K, conductance G_{ij} in W/K, and heat-flow terms in watts. Here $Q_{i,\text{diss}}$ excludes the separately listed heater input. Heater electricity is counted once in P_{load} and its delivered heat once in $Q_{i,\text{heater}}$. Battery and converter losses are assigned to the corresponding thermal nodes. Motor power is divided between mechanical work and local losses, so the heat model follows where energy is actually deposited.

The coupling enters through the task modes and motion. Let u collect activity and computing modes, heater commands, and actuator commands, q the spacecraft attitude, a the mechanism configuration, and t the mission time along a candidate trajectory. Writing $z = (q, a)$ for the combined configuration, the interfaces provide

$$\begin{aligned} P_{\text{load}} &= p(u, z, \dot{z}, t), \\ Q_{i,\text{diss}} &= d_i(u, z, \dot{z}, T), \\ P_{\text{gen}} &= g(q, a, t, T_{\text{pv}}), \\ Q_{i,\text{env}} &= b_i(q, a, t), \end{aligned} \quad (4)$$

where T_{pv} is solar-array temperature. Radiative rejection also depends on temperature and on the view factors set by q and a . These mappings can come from measured load profiles, physical models, or calibrated surrogates. Changing a task thus updates its electrical and heat loads; changing an orientation updates generation and thermal exposure. The fixed-motion optimization of Doff-Sotta et al. [67] provides one starting point for this calculation. A robot that can choose its motion evaluates the mappings for each candidate action.

Over the planning horizon, representative constraints are

$$\begin{aligned} E_{\text{reserve}} &\leq E \leq E_{\text{max}}(\text{SoH}), \\ T_{i,\text{min}} &\leq T_i \leq T_{i,\text{max}}, \\ |h_j| &\leq h_{j,\text{max}}, \quad 0 \leq P_{\text{dis}} \leq P_{\text{dis,max}}(E, T_b, \text{SoH}). \end{aligned} \quad (5)$$

Here T_b is battery temperature. The stored-energy state changes through the power balance; temperature-dependent access to that energy enters the discharge limit and the reserve. Capacity is treated as fixed over this short planning horizon. The reserve covers unavailable energy under the expected battery conditions and the energy needed for a safe transition. Charge limits, mutually exclusive charge/discharge commands, execution deadlines, pointing limits, and contact conditions complete the model for a particular mission.

Before accepting a task, resource management applies margins based on forecast and state-estimation errors measured over the intended operating conditions. A predicted peak of 79°C against an 80°C limit leaves little room if errors are typically 2 K; the average error alone is not an upper bound for setting that margin. When conditions fall outside the model's tested range, a conservative load envelope or an already approved plan provides an alternative. This is how the uncertainty passed in Table I changes the decision made with (5).

C. Authority, Timing, and Recovery

Task planning proposes activities and priorities. Resource management returns feasible load modes and timing, while

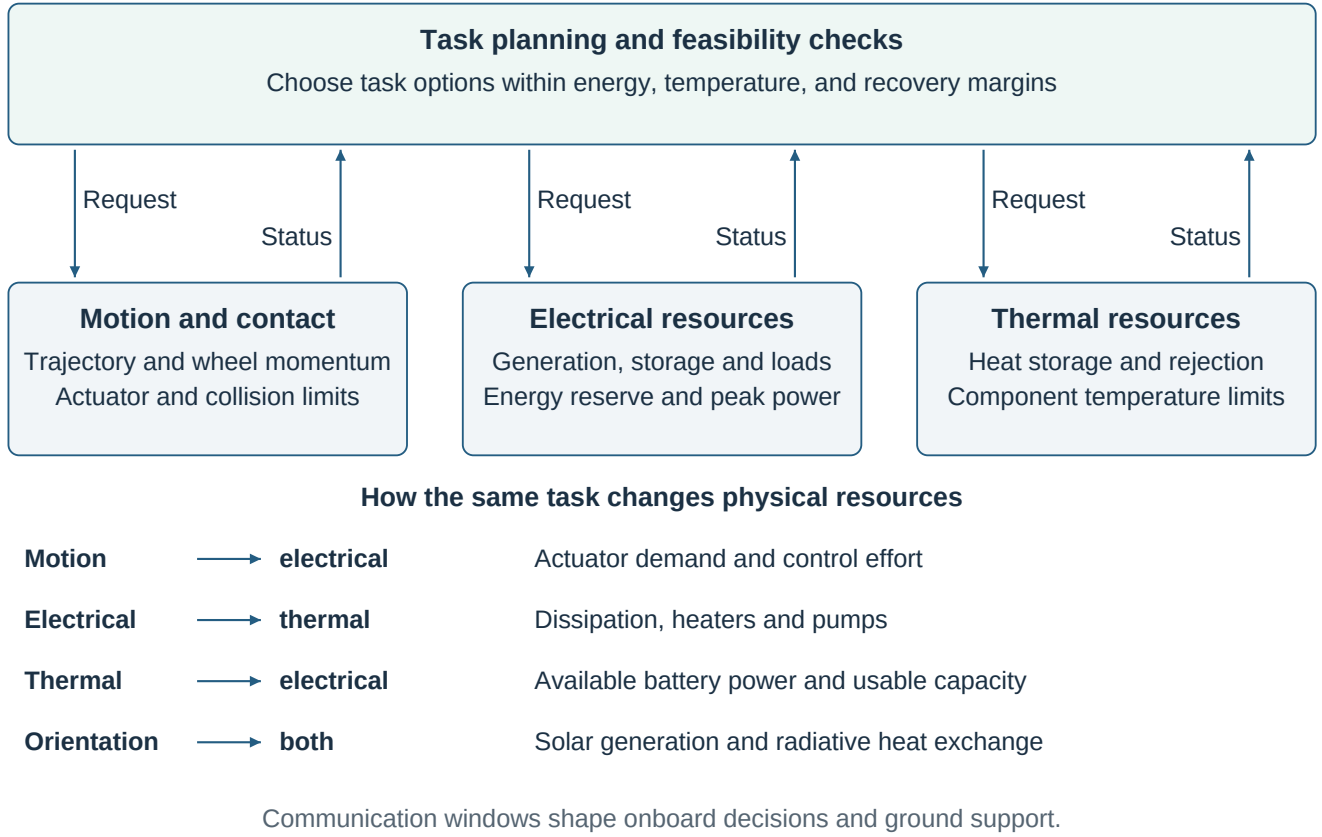


Fig. 3. Coupling between tasks, motion, and resources. Actions change electrical demand and thermal conditions; temperature also changes battery and generation limits. Forecasts and feasible motion options guide the next task decision.

motion control supplies trajectories and enforces fast attitude, proximity, and contact limits. Collision avoidance and a protected retreat reserve remain part of the task's acceptance conditions. MPC, constrained optimization, and precomputed rules are alternative ways to implement scheduling within this division of responsibility.

Fast protection and motion loops run independently of slower forecasts. If the scheduler misses its deadline, command arbitration retains an approved plan or selects a defined safe transition. A protection monitor needs enough time for the fallback controller to complete that transition [81]. Resolving resource conflicts before task acceptance keeps routine scheduling separate from the abort or survival actions used when operating limits are threatened.

D. Illustrative Capture Scenario

Consider a servicer preparing to capture a non-cooperative client before eclipse. Its processor has limited heat rejection, and its battery must support approach, contact, and recovery. The planner compares three options: capture now with the full processing workload (A), capture now while deferring optional image analysis and communications (B), or wait for the next opportunity (C). Navigation and collision avoidance continue in all three. The following comparison is illustrative;

the mission's load profiles and predicted motion would supply the actual values.

For each option, (4) supplies the electrical and thermal loads. Equations (1)–(3) propagate energy and temperature through approach, capture, and the following eclipse. Suppose A retains the required energy reserve but its predicted processor temperature exceeds the limit after allowing for forecast error. It is rejected on thermal grounds even though its trajectory is collision-free. B uses the same capture trajectory with a lower optional workload. If its revised forecast meets the temperature, battery-power, energy-reserve, and wheel-momentum limits in (5), it can complete capture in the current window. B is then selected when that timing has priority.

C offers time to cool or recharge, at the cost of a later capture and additional waiting loads. If B still leaves too little temperature margin, the planner evaluates C from the state reached after waiting, including eclipse and station-keeping demand. During execution, measurements update the forecasts. Optional loads can be stopped first; a continued loss of margin triggers a feasible hold or retreat. After contact, recovery also accounts for the attached client's motion. The choice therefore follows a specific bottleneck: thermal headroom decides between A and B, while the cost of waiting determines whether C is viable.

VIII. VERIFICATION, DEPLOYMENT, AND STANDARDS

An embodied controller must meet requirements for both its actions and the resources used to carry them out. A perception model may estimate pose accurately yet take too long for a control update; an attitude maneuver may track well while reducing solar power. Verification therefore covers the learned component, its interfaces, and the spacecraft response as one connected system.

A. Applicable Requirements and Guidance

ECSS-E-ST-40C Rev. 1 and ECSS-Q-ST-80C Rev. 2, issued in April 2025, address software engineering and software product assurance [82], [83]. ECSS-E-ST-70-11C Rev. 1, issued in October 2025, addresses space-segment operability [84]. Projects tailor these requirements to the mission. The 2024 ECSS machine-learning handbook provides complementary guidance: Section 6.4.1 connects data coverage to operating scenarios, Section 6.4.3.1.2 links model tests to functional and interface requirements, and Sections 6.4.4.3.3–6.4.4.3.4 connect failure analysis to safety cages that monitor learned outputs and select fallback actions [85]. These are engineering recommendations, with project requirements set through the applicable standards and mission process.

NASA’s software engineering requirements also cover AI/ML software [86]. Section 7.25 of its online handbook advises against AI/ML in safety-critical or other high-criticality functions and recommends advisory use with outputs independently confirmed by non-AI means [87]. Projects set the permitted control authority through the applicable mission rules and technical authorities.

Radiation testing addresses hardware behavior under particle exposure. ECSS-Q-ST-60-15C Rev. 1 defines a project-specific radiation-hardness assurance process [88], while ESCC 25100 specifies single-event-effects test methods [89]. Results should be tied to the tested particle, electrical, temperature, and operating conditions. The system assessment then examines upset consequences, detection time, and recovery, including faults shared by the controller and its monitor.

Verification checks requirements; validation checks usefulness for the intended task; qualification tests the design against specified conditions; mission acceptance considers the assembled evidence. Keeping these roles clear helps connect component tests to an operational decision. Aviation guidance such as DO-178C can offer useful practice, while its FAA recognition applies to airborne software [90].

B. Evidence at the Interfaces

Table VII applies this guidance to coupled spacecraft hazards. For a pose estimator, data coverage includes lighting, occlusion, and relative motion; interface tests also check timing and the handling of invalid outputs. A safety cage needs evaluation alongside the learned controller, including shared power, memory, and data interfaces and the transition to fallback control [85]. The thermal and energy tests in the table extend these checks through recovery: a component may keep heating after its workload stops, and a fallback may consume

appreciable battery energy. Trigger thresholds therefore need to leave enough time and resources for the resulting trajectory, allowing for sensor error and detection delay.

The flight examples in Section V show two uses of protection. APIARY’s error monitor triggered a return to baseline control during a docking failure [74]. LeLaR’s protection was available throughout its pointing tests, and its idle fallback was chosen to fit InnoCube’s power balance [40]. Testing should cover the monitor, the transfer of control, and the resulting spacecraft state.

Formal neural-network analysis can establish properties over a specified input range [91]; control barrier functions can restrict actions using a dynamics model and state estimate [92]. Their use depends on the accuracy of those descriptions and the delay in applying a command. Runtime assurance combines a monitor with a fallback controller [81]. For spacecraft, the fallback must preserve energy and thermal reserves as well as position and attitude, so recovery tests need to continue beyond the immediate switch.

C. Managing Changes After Deployment

Deployment can involve fixed model parameters, bounded adaptation, retraining, or a software upload. Each changes a different part of the system. Fixed-parameter inference needs testing across the expected inputs; adaptation also needs limits on the quantities allowed to change. Ground retraining allows regression tests before release. Onboard learning needs checks that remain effective as the model changes.

For each update, the project should record the data, model, and software versions; identify affected requirements; and define activation and rollback rules. For agents, the recorded configuration also includes prompts, persistent memory, skill libraries, and tool configurations, because behavior can change with fixed model weights. Authentication checks where an update came from and whether it arrived intact. Behavioral tests check its actions, tool calls, and resource demand; rollback restores a compatible configuration and execution state. OPS-SAT illustrates how experimentation can be separated from spacecraft protection: its payload supported software uploads and reconfigurable hardware, while the bus monitored the payload and could regain control [93].

D. From Simulation to Flight

A practical test sequence combines software simulation, target-processor tests, hardware-in-the-loop tests, environmental and facility tests, and bounded flight experiments. Optical testbeds and air-bearing facilities reproduce useful parts of the sensing and motion problem [72], [94]. The test plan should assign each facility a clear role and use analysis or further tests for the remaining uncertainties.

Telemetry replay helps diagnose failures and compare candidate updates on recorded observations. Closed-loop simulation is needed to examine how a changed action alters later observations. Together, these methods support flight experiments with defined limits on duration, control authority, and recovery time. The results can then support a wider operating range.

TABLE VII
EXAMPLES OF COUPLED HAZARDS, PROTECTIVE ACTIONS, AND TESTS

Hazard	Monitor and candidate fallback	Tests needed
Compute overload delays control and increases heat	Monitor deadlines, queues, and temperatures; suspend optional inference while retaining essential control	Measure execution and switching times on target hardware under concurrent load and temperature variation, including the fallback's workload.
Attitude or task choices exhaust energy reserves	Check charge, predicted generation, and thermal margins; defer tasks or enter a power-safe attitude	Test eclipse and low-generation cases with battery uncertainty and sensor faults. Check recovery using the energy available at the trigger.
Incorrect target pose causes a hazardous approach or contact	Check pose confidence, separation, and contact force; hold, retreat, or stop	Combine optical testbeds, dynamics simulation, and contact tests with lighting changes, occlusion, and delays. Include stopping distance and contact loads.
Radiation faults corrupt inference or commands	Use memory checks, watchdogs, and command bounds; reset or transfer to a qualified control mode	Combine irradiation with fault injection and hardware-in-the-loop recovery tests. Examine faults affecting both the monitor and controller.
A model or agent configuration update changes behavior	Check versions, integrity, outputs, and resource use; reject activation or restore a known configuration	Test interrupted uploads, incompatible tools or interfaces, corrupted parameters, and rollback in representative mission modes. Check state consistency after recovery.

IX. OPEN CHALLENGES AND RESEARCH PRIORITIES

The comparisons above expose three gaps between a promising component result and a useful operational capability. Processor benchmarks often omit work around inference; thermal-model error is rarely translated into a task decision; and a supervisor's update period can be poorly matched to the changing environment. These gaps suggest concrete comparisons for the next studies.

A. Compare Complete Tasks

A useful benchmark follows a task from sensing through computation and motion to completion or recovery. Examples include an inspection before eclipse, a capture with limited momentum margin, and a traverse with uncertain energy demand. The initial state, environment, resource limits, and available recovery actions define what success means in each case.

The baseline should include a well-tuned conventional planner or controller with the same sensors, actuators, and resource limits. Existing resource-aware scheduling provides a stronger comparison than a fixed timeline alone. Reports can then relate task completion to energy use, peak temperature, execution time, and fallback frequency. For example, the Akida comparison in Table IV motivates measuring both steady inference and operation at the required sensing rate, including idle periods, input preparation, and pose recovery. The resulting energy per accepted pose or completed maneuver is more useful for selection than power alone. Tests with changed illumination, loads, and sensor quality show where that choice changes.

B. Link Hardware and Policy Design

Hardware changes can alter what the control software needs to do. Better sensor placement may simplify pose estimation; a compliant manipulator may reduce contact loads; a different processor may change both inference speed and cooling demand. Co-design studies can separate these effects by comparing a new policy on unchanged hardware with a joint

hardware-policy revision under the same mission constraints. The outcome of interest is useful work over the mission, including mass, energy, durability, and the effort needed to verify the design.

Thermal and energy models have a particularly direct role. Two forecasts can have similar average error yet lead to different choices near a temperature limit. A useful comparison places the same scheduler behind a physical model and a learned surrogate, then records accepted activities, unnecessary deferrals, and actual margin violations. The margins described in Section VII should be checked under missing measurements and delayed forecasts as well as nominal conditions. This tests whether faster prediction creates additional feasible work and whether the model recognizes conditions it handles poorly.

Long-duration surface missions make this connection tangible. Installed power, storage, insulation, and heat sources set the range of feasible activity. Resource-aware autonomy can use favorable periods more effectively, choose reduced activity or sleep when needed, and retain enough energy for wake-up and recovery. The research question is how much useful work can be completed within that physical budget.

C. Build on Ground and Flight Experience

A practical development path begins with reproducible models, moves to integrated tests on representative hardware, and then introduces a bounded capability in flight. Broader use follows evidence across the relevant operating modes. The ECSS technology-readiness guidance similarly relates maturity to a specific technology and its intended environment [95].

For supervisory systems, timing deserves its own experiment. ASTREA's different outcomes with short and orbit-length analysis windows motivate comparing update periods under the same load and environmental history. The comparison should record decision age when applied, temperature excursions, and computing time left for payload work. For motion control, the recovery events and test limits documented by APIARY and LeLaR identify useful cases for longer and more varied trials. Ground tests and telemetry can then refine

models and configurations before a later software release expands their operating range.

X. CONCLUSION

Embodied intelligence connects what a space robot decides with what its hardware can sustain. The reviewed computing, thermal-control, and planning studies show why that connection changes engineering choices. A low-power processor can spend more energy on an inference if it runs much more slowly. A fast temperature predictor supports additional activity only when its errors leave enough operating margin. A slow supervisor can exploit recurring environmental conditions, while updates based on an unsuitable time window can make control worse.

Operational planners already provide mechanisms for scheduling activities, checking resources, and revising plans. Learned perception, thermal models, and controllers can extend those mechanisms where their accuracy and timing suit the task. The resource balances and capture example show how to make the resulting choice explicit: compare candidate loads and trajectories, identify the binding constraint, and retain a feasible recovery action.

The next comparisons should therefore measure the work surrounding inference, the decisions changed by a forecast, and the effect of supervisory timing. These tests connect a component's reported performance to the additional observations, maneuvers, or operating time it enables on a particular spacecraft.

ABOUT ASTROUP

Founded in 2026 and headquartered in Beijing, AstroUp focuses on the development and application of embodied robots for space.

Drawing on embodied intelligence, brain-inspired models, and multimodal perception, the company develops robotic systems with perception, decision-making, and manipulation capabilities for satellite servicing, in-orbit research, development, and manufacturing, and lunar and deep-space exploration.

The team consists of young scientists in aerospace and artificial intelligence. Its core members have experience in the overall design of space robotic systems and in multiple in-orbit validation campaigns. By combining engineering experience gained from real space missions with artificial intelligence, AstroUp advances the development, in-orbit validation, and practical deployment of space robotic systems.

REFERENCES

- [1] R. Pfeifer and J. Bongard, *How the Body Shapes the Way We Think*. MIT Press, 2006.
- [2] J. Duan, S. Yu, H. L. Tan, H. Zhu, and C. Tan, "A survey of embodied AI: From simulators to research tasks," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 2, pp. 230–244, 2022.
- [3] A. O'Neill et al., "Open X-Embodiment: Robotic learning datasets and RT-X models," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 6892–6903.
- [4] B. Zitkovich et al., "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. 7th Conf. Robot Learning, Proc. Mach. Learn. Res.*, vol. 229, pp. 2165–2183, 2023. [Online]. Available: <https://proceedings.mlr.press/v229/zitkovich23a.html>
- [5] M. J. Kim et al., "OpenVLA: An open-source vision-language-action model," in *Proc. 8th Conf. Robot Learning, Proc. Mach. Learn. Res.*, vol. 270, pp. 2679–2713, 2025. [Online]. Available: <https://proceedings.mlr.press/v270/kim25c.html>
- [6] Y. Ma, Z. Song, Y. Zhuang, J. Hao, and I. King, "A survey on vision-language-action models for embodied AI," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2026, doi: 10.1109/TNNLS.2025.3650584.
- [7] G. Furano et al., "Towards the use of artificial intelligence on the edge in space systems: Challenges and opportunities," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 35, no. 12, pp. 44–56, 2020.
- [8] N. Muscettola, P. P. Nayak, B. Pell, and B. C. Williams, "Remote Agent: To boldly go where no AI system has gone before," *Artif. Intell.*, vol. 103, no. 1–2, pp. 5–47, 1998.
- [9] NASA Jet Propulsion Laboratory, "Remote Agent experiment meets all objectives," Jun. 7, 1999. [Online]. Available: <https://www.jpl.nasa.gov/news/remote-agent-experiment-meets-all-objectives/>
- [10] NASA Jet Propulsion Laboratory, "M2020 onboard scheduler," Artificial Intelligence Group, project documentation, status Jan. 2025. Accessed: Oct. 15, 2026. [Online]. Available: <https://ai.jpl.nasa.gov/public/projects/m2020-scheduler/>
- [11] A. Flores-Abad, O. Ma, K. Pham, and S. Ulrich, "A review of space robotics technologies for on-orbit servicing," *Prog. Aerosp. Sci.*, vol. 68, pp. 1–26, 2014.
- [12] N. Fallahiazar-zoodar and Z. H. Zhu, "Review of autonomous space robotic manipulators for on-orbit servicing and active debris removal," *Space: Sci. Technol.*, vol. 5, Art. no. 0291, 2025.
- [13] M. Tipaldi, R. Iervolino, and P. R. Massenio, "Reinforcement learning in spacecraft control applications: Advances, prospects, and challenges," *Annu. Rev. Control*, vol. 54, pp. 1–23, 2022.
- [14] D. Izzo, M. Märten, and B. Pan, "A survey on artificial intelligence trends in spacecraft guidance dynamics and control," *Astrodynamics*, vol. 3, no. 4, pp. 287–299, 2019.
- [15] K. Thangavel, R. Sabatini, A. Gardi, K. Ranasinghe, S. Hilton, P. Servidia, and D. Spiller, "Artificial intelligence for trusted autonomous satellite operations," *Prog. Aerosp. Sci.*, vol. 144, Art. no. 100960, 2024, doi: 10.1016/j.paerosci.2023.100960.
- [16] European Cooperation for Space Standardization, "Space engineering: Space environment," ECSS-E-ST-10-04C Rev. 1, Jun. 15, 2020. [Online]. Available: [https://ecss.nl/wp-content/uploads/2020/07/ECSS-E-ST-10-04C-Rev.1\(15June2020\).pdf](https://ecss.nl/wp-content/uploads/2020/07/ECSS-E-ST-10-04C-Rev.1(15June2020).pdf)
- [17] NASA Electronic Parts and Packaging Program, "Electronic Radiation Characterization Project (ERC)," Accessed: Oct. 15, 2026. [Online]. Available: <https://nepp.nasa.gov/pages/Radiation.cfm>
- [18] NASA, "7.0 Thermal control," *State-of-the-Art Small Spacecraft Technology*. Accessed: Oct. 15, 2026. [Online]. Available: <https://www.nasa.gov/smallsat-institute/sst-soa/thermal-control/>
- [19] D. G. Gilmore, *Spacecraft Thermal Control Handbook*, 2nd ed. Aerospace Press, 2002.
- [20] NASA Goddard Space Flight Center, "100 lunar days—Parts I and II," Scientific Visualization Studio, Oct. 6, 2017, updated May 3, 2023. [Online]. Available: <https://svs.gsfc.nasa.gov/12739>
- [21] European Cooperation for Space Standardization, "Space product assurance: Thermal vacuum outgassing test for the screening of space materials," ECSS-Q-ST-70-02C, ESA-ESTEC, Noordwijk, The Netherlands, 2008.
- [22] J. R. Gaier, "The effects of lunar dust on EVA systems during the Apollo missions," NASA Glenn Research Center, Cleveland, OH, USA, NASA/TM-2005-213610, 2005.
- [23] T. Smith, J. Barlow, M. Bualat, T. Fong, C. Provencher, H. Sanchez, E. Smith, and the Astrobe Team, "Astrobee: A new platform for free-flying robotics on the International Space Station," in *Proc. Int. Symp. Artif. Intell., Robot. Autom. Space (i-SAIRAS)*, Beijing, China, 2016. [Online]. Available: <https://ntrs.nasa.gov/citations/20160007769>
- [24] K. Yoshida, "Engineering Test Satellite VII flight experiments for space robot dynamics and control," *Int. J. Robot. Res.*, vol. 22, no. 5, pp. 321–335, 2003.
- [25] GITAI, "Space robotics start-up GITAI completes successful technology demonstration inside the ISS," Oct. 28, 2021. [Online]. Available: <https://gitai.tech/2021/10/28/iss-tech-demo-ja/>
- [26] GITAI, "GITAI completes fully successful technology demonstration outside the ISS," Mar. 19, 2024. [Online]. Available: <https://gitai.tech/2024/03/19/gitai-completes-fully-successful-technology-demonstration-outside-the-iss/>

- [27] eoPortal Satellite Missions, “ISS: MSS (Mobile Servicing System),” 2024. [Online]. Available: <https://www.eoportal.org/satellite-missions/iss-mss>
- [28] Canadian Space Agency, “Dextre, the ISS’s versatile robot (Special Purpose Dexterous Manipulator),” mission documentation, 2008. [Online]. Available: <https://www.asc-csa.gc.ca/eng/iss/dextre/>
- [29] X. Wang, Q. Zhang, and W. Wang, “Design and application prospect of China’s Tiangong space station,” *Space: Sci. Technol.*, vol. 3, Art. no. 0035, 2023, doi: 10.34133/space.0035.
- [30] R. B. Friend, “Orbital Express program summary and mission overview,” in *Proc. SPIE, Sensors and Systems for Space Applications II*, vol. 6958, Orlando, FL, USA, 2008, Art. no. 695803, doi: 10.1117/12.783792.
- [31] NASA Ames Research Center, “Astrobee,” Accessed: Oct. 15, 2026. [Online]. Available: <https://www.nasa.gov/ames/astrobee/>
- [32] M. A. Diffler *et al.*, “Robonaut 2—The first humanoid robot in space,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Shanghai, China, 2011, pp. 2178–2183. [Online]. Available: <https://www.cs.columbia.edu/~allen/S19/robonaut-2.pdf>
- [33] NASA, “Mission to the International Space Station,” Robonaut 2 mission documentation. Accessed: Oct. 15, 2026. [Online]. Available: <https://www.nasa.gov/robonaut2/mission-to-the-international-space-station/>
- [34] V. Verma, M. Maimone, D. M. Gaines *et al.*, “Autonomous robotics is driving Perseverance rover’s progress on Mars,” *Sci. Robot.*, vol. 8, no. 80, Art. no. eadi3099, 2023.
- [35] R. C. Moeller *et al.*, “The sampling and caching subsystem (SCS) for the scientific exploration of Jezero crater by the Mars 2020 Perseverance rover,” *Space Sci. Rev.*, vol. 217, Art. no. 5, 2021.
- [36] E. W. Schaler *et al.*, “Two-stage calibration of a 6-axis force-torque sensor for robust operation in the Mars 2020 robot arm,” *Adv. Robot.*, vol. 35, no. 21–22, pp. 1347–1358, 2021.
- [37] E. Mingo Hoffman *et al.*, “Design and validation of a multi-arm relocatable manipulator for space applications,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, London, U.K., 2023, pp. 11887–11893, doi: 10.1109/ICRA48891.2023.10160389.
- [38] NASA, “On-orbit servicing, assembly, and manufacturing 2 (OSAM-2),” mission documentation. Accessed: Oct. 15, 2026. [Online]. Available: <https://www.nasa.gov/mission/on-orbit-servicing-assembly-and-manufacturing-2-osam-2/>
- [39] European Cooperation for Space Standardization, “Space engineering: Mechanisms,” ECSS-E-ST-33-01C Rev. 2, ESA-ESTEC, Noordwijk, The Netherlands, 2019.
- [40] K. Djebko, T. Baumann, E. Dilger, F. Puppe, and S. Montenegro, “LeLaR: The first in-orbit demonstration of an AI-based satellite attitude controller,” *IEEE Access*, vol. 14, pp. 49348–49367, 2026, doi: 10.1109/ACCESS.2026.3678816.
- [41] G. Gallego *et al.*, “Event-based vision: A survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, 2022, doi: 10.1109/TPAMI.2020.3008413.
- [42] G. Cohen *et al.*, “Event-based sensing for space situational awareness,” *J. Astronaut. Sci.*, vol. 66, no. 2, pp. 125–141, 2019.
- [43] A. Rathinam, J. Lecomte, J. Reelsen, G. Lenz, A. von Arnim, and D. Aouada, “Efficient onboard spacecraft pose estimation with event cameras and neuromorphic hardware,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2026, pp. 10339–10347. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2026W/A14Space/html/Rathinam_Efficient_Onboard_Spacecraft_Pose_Estimation_with_Event_Cameras_and_Neuromorphic_CVPRW_2026_paper.html
- [44] Frontgrade Gaisler, “GR740: Quad-core LEON4FT microprocessor,” product documentation. Accessed: Oct. 15, 2026. [Online]. Available: <https://www.gaisler.com/products/gr740>
- [45] G. Giuffrida *et al.*, “The Φ -sat-1 mission: The first on-board deep neural network demonstrator for satellite Earth observation,” *IEEE Trans. Geosci. Remote Sens.*, vol. 60, Art. no. 5517414, pp. 1–14, 2022, doi: 10.1109/TGRS.2021.3125567.
- [46] G. Furano, “PhiSat-1 successfully applies AI to process Earth Observation imagery in-flight in historic first for Space,” in *14th ESA Workshop Avionics, Data, Control Softw. Syst. (ADCSS)*, Oct. 2020. [Online]. Available: <https://indico.esa.int/event/338/contributions/5692/>
- [47] AMD, “Radiation tolerant Versal AI Core series data sheet,” DS946, ver. 1.2, Feb. 19, 2025. [Online]. Available: <https://docs.amd.com/r/en-us/ds946-xqr-versal-ai-core/General-Description>
- [48] J. Nash, V. Verma, E. Graser, H. Wang, G. Kubiak, M. Maimone, S. Myint, J. Biesiadecki, and P. Romano, “A software-based approach to radiation mitigation for planetary missions,” in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, 2026, pp. 1–18, doi: 10.1109/AERO66936.2026.11519792.
- [49] E. Lagunas *et al.*, “Performance evaluation of neuromorphic hardware for onboard satellite communication applications,” *IEEE Wireless Commun.*, vol. 31, no. 6, pp. 78–84, 2024.
- [50] M. Wirthlin, “High-reliability FPGA-based systems: Space, high-energy physics, and beyond,” *Proc. IEEE*, vol. 103, no. 3, pp. 379–389, Mar. 2015.
- [51] M. R. Patel, *Spacecraft Power Systems*. CRC Press, 2005.
- [52] J. Meseguer, I. Pérez-Grande, and A. Sanz-Andrés, *Spacecraft Thermal Control*. Woodhead Publishing, 2012.
- [53] NASA, “NASA investigates a backup cooling device for Stirling radioisotope power systems,” Feb. 17, 2017, updated Jul. 29, 2024. Accessed: Oct. 15, 2026. [Online]. Available: <https://science.nasa.gov/science-research/science-enabling-technology/nasa-investigates-backup-cooling-device-stirling-radioisotope-power-systems/>
- [54] J. H. Henninger, “Solar absorptance and thermal emittance of some common spacecraft thermal-control coatings,” NASA Reference Publication 1121, 1984. [Online]. Available: <https://ntrs.nasa.gov/citations/19840015630>
- [55] H. Tanaka and H. Nagai, “Thermal surrogate model for spacecraft systems using physics-informed machine learning with POD data reduction,” *Int. J. Heat Mass Transfer*, vol. 213, Art. no. 124336, 2023, doi: 10.1016/j.ijheatmasstransfer.2023.124336.
- [56] D. Yamashita, H. Tanaka, T. Ikami, and H. Nagai, “Transient system-wide surrogate model for spacecraft thermal analysis using recurrent neural networks with POD data reduction,” *Int. J. Heat Mass Transfer*, vol. 255, pt. 2, Art. no. 127854, 2026, doi: 10.1016/j.ijheatmasstransfer.2025.127854.
- [57] B. Jang, W. Lee, J.-J. Lee, and H. Jin, “Artificial neural network-based temperature prediction of a lunar orbiter in thermal vacuum test: Data-driven reduced-order models,” *Aerosp. Sci. Technol.*, vol. 145, Art. no. 108867, 2024, doi: 10.1016/j.ast.2023.108867.
- [58] B. Jang, M.-J. Jeon, S. Lee, and H. Jin, “Neural network-based thermal model for virtual metrology of lunar orbiter temperatures via active and transfer learning,” *Int. Commun. Heat Mass Transfer*, vol. 165, Art. no. 109055, 2025, doi: 10.1016/j.icheatmasstransfer.2025.109055.
- [59] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, “Learning-based model predictive control: Toward safe learning in control,” *Annu. Rev. Control Robot. Auton. Syst.*, vol. 3, pp. 269–296, 2020, doi: 10.1146/annurev-control-090419-075625.
- [60] A. D. Mousist, “Autonomous payload thermal control,” in *Proc. SPAICE*, 2024, doi: 10.5281/zenodo.13885627. [Online]. Available: <https://arxiv.org/abs/2307.15438v3>
- [61] A. D. Mousist, “ASTREA: Introducing agentic intelligence for orbital thermal autonomy,” arXiv:2509.13380v2, Oct. 11, 2025. [Online]. Available: <https://arxiv.org/abs/2509.13380v2>
- [62] Thales Alenia Space, “IMAGIN-e payload unlocks the potential of space edge computing aboard the ISS,” Jan. 30, 2026. Accessed: Oct. 15, 2026. [Online]. Available: <https://www.thalesaleniaspace.com/en/news/imagin-e-payload-unlocks-potential-space-edge-computing-aboard-iss>
- [63] NASA, “Apollo 11 technical crew debriefing,” Jul. 31, 1969, sec. 14.4, “Passive thermal control.” Accessed: Oct. 15, 2026. [Online]. Available: <https://www.nasa.gov/wp-content/uploads/static/history/alsj/a11/a11tcdh.html>
- [64] R. R. Richardson, M. A. Osborne, and D. A. Howey, “Gaussian process regression for forecasting battery state of health,” *J. Power Sources*, vol. 357, pp. 209–219, 2017, doi: 10.1016/j.jpowsour.2017.05.004.
- [65] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Soderstrom, “Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding,” in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 387–395.
- [66] A. Fejjari, A. Delavault, R. Camilleri, and G. Valentino, “A review of anomaly detection in spacecraft telemetry data,” *Appl. Sci.*, vol. 15, no. 10, Art. no. 5653, 2025.
- [67] M. Doff-Sotta, M. Cannon, and J. R. Forbes, “Spacecraft energy management using convex optimisation,” *Energy Convers. Manag.*, vol. 16, Art. no. 100325, 2022, doi: 10.1016/j.ecmx.2022.100325.
- [68] M. Bisgaard, D. Gerhardt, H. Hermanns, J. Krčál, G. Nies, and M. Stenger, “Battery-aware scheduling in low orbit: The GomX-3 case,” *Formal Aspects Comput.*, vol. 31, no. 2, pp. 261–285, 2019, doi: 10.1007/s00165-018-0458-2.
- [69] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, “Temperature-aware microarchitecture:

- Modeling and implementation,” *ACM Trans. Archit. Code Optim.*, vol. 1, no. 1, pp. 94–125, 2004.
- [70] China National Space Administration, “Chang’e-4 lander and Yutu-2 rover survive the lunar night and begin their second lunar day,” Jan. 31, 2019 (in Chinese). Accessed: Oct. 15, 2026. [Online]. Available: <https://www.cnsa.gov.cn/n6758823/n6758838/c6805330/content.html>
- [71] S. Sharma and S. D’Amico, “Neural network-based pose estimation for noncooperative spacecraft rendezvous,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 6, pp. 4638–4658, 2020.
- [72] T. H. Park, M. Märtens, G. Lecuyer, D. Izzo, and S. D’Amico, “SPEED+: Next-generation dataset for spacecraft pose estimation across domain gap,” in *Proc. IEEE Aerosp. Conf.*, 2022, pp. 1–15.
- [73] K. Hovell and S. Ulrich, “Deep reinforcement learning for spacecraft proximity operations guidance,” *J. Spacecr. Rockets*, vol. 58, no. 2, pp. 254–264, 2021.
- [74] S. Chapin, K. Stewart, R. Leontie, and C. G. Henshaw, “Autonomous Planning In-space Assembly Reinforcement-learning free-flyer (APIARY) International Space Station Astrobotic testing,” in *Proc. Int. Conf. Space Robot. (iSpaRo)*, Sendai, Japan, 2025, pp. 418–424, doi: 10.1109/iSpaRo66239.2025.11436341.
- [75] E. Papadopoulos and S. Dubowsky, “On the nature of control algorithms for free-floating space manipulators,” *IEEE Trans. Robot. Autom.*, vol. 7, no. 6, pp. 750–758, 1991.
- [76] S. Chien *et al.*, “Using autonomy flight software to improve science return on Earth Observing One,” *J. Aerosp. Comput. Inf. Commun.*, vol. 2, no. 4, pp. 196–216, 2005, doi: 10.2514/1.12923. [Online]. Available: <https://www.jpl.nasa.gov/nmp/st6/TECHNOLOGY/AIAA-12923-137.pdf>
- [77] B. Ichter *et al.*, “Do as I can, not as I say: Grounding language in robotic affordances,” in *Proc. 6th Conf. Robot Learning, Proc. Mach. Learn. Res.*, vol. 205, pp. 287–318, 2023. [Online]. Available: <https://proceedings.mlr.press/v205/ichter23a.html>
- [78] A. Wu, H. Han, X. Luo, R. Wang, S. He, and X. Wan, “SpaceMind: A modular and self-evolving embodied vision-language agent framework for autonomous on-orbit servicing,” arXiv:2604.14399, 2026. [Online]. Available: <https://arxiv.org/abs/2604.14399>
- [79] S. Dubowsky and E. Papadopoulos, “The kinematics, dynamics, and control of free-flying and free-floating space robotic systems,” *IEEE Trans. Robot. Autom.*, vol. 9, no. 5, pp. 531–543, 1993.
- [80] Y. Umetani and K. Yoshida, “Resolved motion rate control of space manipulators with generalized Jacobian matrix,” *IEEE Trans. Robot. Autom.*, vol. 5, no. 3, pp. 303–314, 1989.
- [81] J. D. Schierman, M. D. DeVore, N. D. Richards, and M. A. Clark, “Runtime assurance for autonomous aerospace systems,” *J. Guid. Control Dyn.*, vol. 43, no. 12, pp. 2205–2217, 2020, doi: 10.2514/1.G004862.
- [82] European Cooperation for Space Standardization, “Space engineering: Software,” ECSS-E-ST-40C Rev. 1, Apr. 30, 2025. [Online]. Available: <https://ecss.nl/standard/ecss-e-st-40c-rev-1-software-30-april-2025/>
- [83] European Cooperation for Space Standardization, “Space product assurance: Software product assurance,” ECSS-Q-ST-80C Rev. 2, Apr. 30, 2025. [Online]. Available: <https://ecss.nl/standard/ecss-q-st-80c-rev-2-software-product-assurance-30-april-2025/>
- [84] European Cooperation for Space Standardization, “Space engineering: Space segment operability,” ECSS-E-ST-70-11C Rev. 1, Oct. 15, 2025. [Online]. Available: <https://ecss.nl/standard/ecss-e-st-70-11c-rev-1-space-segment-operability-15-october-2025/>
- [85] European Cooperation for Space Standardization, “Space engineering: Machine learning handbook,” ECSS-E-HB-40-02A, Nov. 15, 2024. [Online]. Available: [https://ecss.nl/wp-content/uploads/2024/12/ECSS-E-HB-40-02A\(15November2024\).pdf](https://ecss.nl/wp-content/uploads/2024/12/ECSS-E-HB-40-02A(15November2024).pdf)
- [86] NASA, “NASA software engineering requirements,” NPR 7150.2D, Washington, DC, 2022.
- [87] NASA, “Super intelligence and software engineering,” *Software Engineering Handbook*, sec. 7.25, online guidance. Accessed: Oct. 15, 2026. [Online]. Available: <https://swehb.nasa.gov/spaces/SWEHBVD/pages/184320054/7.25%2B-%2BSuper%2BIntelligence%2BAnd%2BSoftware%2BEngineering>
- [88] European Cooperation for Space Standardization, “Space product assurance: Radiation hardness assurance—EEE components,” ECSS-Q-ST-60-15C Rev. 1, Mar. 20, 2025. [Online]. Available: <https://ecss.nl/standard/ecss-q-st-60-15c-rev-1-radiation-hardness-assurance-20-march-2025/>
- [89] European Space Components Coordination, “Single event effects test method and guidelines,” ESCC Basic Specification No. 25100, issue 2, Oct. 2014. [Online]. Available: <https://spacecomponents.org/specification/view?id=3095>
- [90] Federal Aviation Administration, “Airborne software development assurance using EUROCAE ED-12() and RTCA DO-178(),” Advisory Circular AC 20-115D, Jul. 21, 2017. [Online]. Available: https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_20-115D.pdf
- [91] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, “Verisig: Verifying safety properties of hybrid systems with neural network controllers,” in *Proc. 22nd ACM Int. Conf. Hybrid Syst.: Comput. Control (HSCC)*, 2019, pp. 169–178.
- [92] A. D. Ames *et al.*, “Control barrier functions: Theory and applications,” in *Proc. Eur. Control Conf. (ECC)*, 2019, pp. 3420–3431.
- [93] European Space Agency, “OPS-SAT,” mission documentation. Accessed: Oct. 15, 2026. [Online]. Available: https://www.esa.int/Enabling_Support/Operations/OPS-SAT
- [94] T. Rybus *et al.*, “Application of a planar air-bearing microgravity simulator for demonstration of operations required for an orbital capture with a manipulator,” *Acta Astronaut.*, vol. 155, pp. 211–229, 2019.
- [95] European Cooperation for Space Standardization, “Space engineering: Technology readiness level (TRL) guidelines,” ECSS-E-HB-11A, Mar. 1, 2017. [Online]. Available: <https://ecss.nl/home/ecss-e-hb-11a-technology-readiness-level-trl-guidelines-1-march-2017/>