Article

Not peer-reviewed version

Robots That Last: Long-Horizon Robotics Should Optimize Persistent Autonomy

Chaoyue He *, Xin Zhou, Di Wang, Hong Xu, Wei Liu, Chunyan Miao

Posted Date: 29 April 2026

doi: 10.20944/preprints202604.2037.v1

Keywords: persistent autonomy; long-horizon robotics; embodied AI; robot learning; continual learning; evaluation metrics; memory hygiene; proactive behavior; vision-language-action models; long-term autonomy

 doi:10.20944/preprints202604.2037.v1

Article

Robots That Last: Long-Horizon Robotics Should Optimize Persistent Autonomy

Chaoyue He [1,*], Xin Zhou [1], Di Wang [1], Hong Xu [1], Wei Liu [2] and Chunyan Miao [1]

[1] Alibaba–NTU Global e-Sustainability CorpLab (ANGEL), Singapore
[2] Alibaba Group, Hangzhou, China
* Correspondence: cyhe@ntu.edu.sg

Abstract

This position paper argues that long-horizon robotics should optimize **persistent autonomy**, not only longer reset-based episodes. Real deployments require robots that remain safely useful over days to months while accumulating memory, adapting to evolving human preferences, recovering from inevitable failures, and managing constrained physical and computational resources. Many embodied AI evaluations still inherit the logic of episodic reinforcement learning—where environments are frequently reset and hidden human labor is often unreported—but continuous operation exposes vulnerabilities in state continuity, resource coupling, recovery, and maintenance. Although long-term autonomy is not conceptually new, recent progress in generalist robot policies, open robot datasets, and language-conditioned control [1–8] makes persistence a primary machine-learning evaluation target rather than a deferred downstream systems-engineering concern. As base policies grow more competent, the practical bottlenecks of autonomy concentrate in memory staleness, hidden intervention burden, recovery loops, and maintenance debt. To align evaluation with these realities, we propose a *persistent-autonomy scorecard* and a layered *benchmark blueprint* centered on long-run service utility, intervention burden, recovery quality, proactive usefulness, memory hygiene, uptime, and wear-adjusted throughput. By treating persistence as the fundamental scientific object, modern robot learning can focus on systems that turn calendar time into compounding competence rather than relying on isolated task success.

Keywords: persistent autonomy; long-horizon robotics; embodied AI; robot learning; continual learning; evaluation metrics; memory hygiene; proactive behavior; vision-language-action models; long-term autonomy

1. Introduction

A robot that executes a thirty-step kitchen instruction once is an impressive achievement. A robot that keeps a kitchen orderly for a month—despite moved objects, depleted supplies, interruptions, battery limits, calibration drift, user preferences, and occasional mistakes—is a qualitatively different system. The former solves a bounded episode. The latter operates persistently.

Much of current long-horizon" robotics still inherits the evaluation logic of episodic reinforcement learning and short-horizon manipulation: initialize the world, issue a goal, measure success, reset, repeat. Benchmarks such as RLBench, ALFRED, and BEHAVIOR-1K [9–11] and policies such as RT-2, Octo, and OpenVLA [5,7,8] have expanded robot competence, but bounded episodes remain an incomplete unit of evidence for robots meant to live in the world. Real deployments accumulate history, wear, and obligations. **Claims of deployment-relevant long-horizon robotic autonomy should be considered incomplete unless they report persistent-autonomy evidence: long-run service utility, intervention burden, recovery quality, uptime, memory staleness, proactive usefulness, and resource or maintenance cost.**

Figure 1 illustrates this persistent evaluation paradigm.

 doi:10.20944/preprints202604.2037.v1

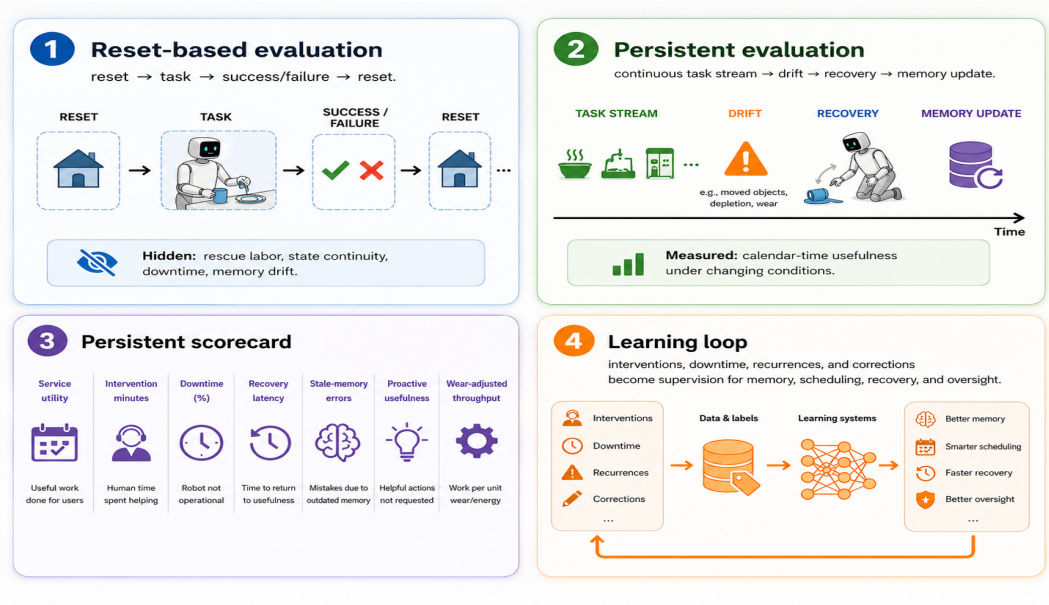

Figure 1. Persistent evaluation paradigm. Long-horizon robotics should be evaluated as persistent service over calendar time, not as isolated reset-based task episodes.

Beyond long-term autonomy. Long-term deployment has been studied for years in service robotics and long-term autonomy [12–14]. The claim here is that modern robot learning should treat persistence as a primary evaluation object from the start. As base policies improve, bottlenecks concentrate in memory staleness, hidden intervention burden, recovery, scheduling, and maintenance. Persistent autonomy reframes the question from "Can the robot finish this longer task chain?" to "Does the robot become more useful, less intervention-heavy, and more viable over calendar time?" Table 1 clarifies the relation to adjacent traditions.

Table 1. Deployment is a known challenge; however, modern long-horizon robot learning must be recentered on persistent-autonomy evidence.

Tradition	What it already contributes	What this perspective changes
Long-term autonomy	non-stationarity, maintenance, deployed behavior	makes persistence a primary evaluation target for current VLA and robot-learning systems
Embodied AI benchmarks	reproducible tasks, simulators, leaderboards	requires temporal continuity, hidden-labor logs, and non-reset scorecards
Generalist robot policies	broad episodic competence and language-conditioned control	asks whether competence survives drift, downtime, rescue, and changing preferences
Continual learning	update, replay, and forgetting mechanisms	ties memory quality to embodied cost, safety, and user-facing deletion constraints

This formulation has four consequences: (1) episodic evaluation suppresses temporal couplings that dominate real robot operation; (2) the central scientific object is persistent autonomy, the capacity to remain safely useful over time while preserving and improving the ability to keep acting; (3) systems claiming deployment-relevant long-horizon autonomy should follow a reporting standard that reflects continuous operation, not only reset-based task success; and (4) persistence belongs inside the core machine-learning problem rather than being deferred to downstream systems engineering.

2. The Episode Is the Wrong Unit of Progress

Resets hide the value and burden of memory. In a reset-heavy benchmark, a map, an episodic trace, a model of user routines, or a record of fragile objects may help, but none is essential to the evaluation itself. In persistent settings, those memories distinguish compounding competence from repeated relearning. The robot should not merely perceive the current scene; it should remember where objects drift, which drawer tends to stick at night, which shelf is often empty on Fridays, and which recovery strategy worked after the last spill. Extended deployments create opportunities to improve precisely because the robot can reuse and update what it has learned across time [12–14].

 doi:10.20944/preprints202604.2037.v1

Resets hide resource coupling, proactivity, and recovery. Most current long-horizon benchmarks do not force a robot to reason about battery state, thermal limits, calibration debt, consumables, actuator wear, or communication outages. Yet these variables shape long-run performance. A persistent robot must decide not only whether an action is possible, but whether it is wise given future obligations and maintenance cost. It also has a broader action space: noticing, preparing, staging, consolidating, inspecting, querying, and preventing. In homes, hospitals, offices, farms, and warehouses, much of the value of continuous operation comes from useful work before a human explicitly asks [15–17]. Finally, failure is not merely terminal; the more important question is whether the system detects failure early, degrades safely, recovers, asks for targeted help, and avoids repeating the same mistake.

Table 2. Why persistent autonomy is not just "a longer episode".

Aspect	Extended-episode framing	Persistent-agent framing
Unit of evaluation	One rollout or one task chain	One deployment over calendar time
Objective	Episode completion	Long-run service utility
World state	Frequently resettable	Partially irreversible, drifting, history-dependent
Robot state	Short-term controller state	Memory, resources, maintenance debt, wear
Failures	Often terminal	Recovery quality is central
Human role	Instruction at the start	Ongoing preference shaping, rescue, and oversight
Useful behavior	Execute requested tasks	Also prepare, monitor, prevent, and ask for help
Learning signal	Per-episode reward/success	Interventions, recurrences, downtime, and routines
Safety view	Per-episode constraint satisfaction	Long-run reliability, escalation, and auditability

Consequently, optimizing mainly for longer and more diverse episodes underreports the capabilities that distinguish a robust always-on robot from a fragile demo. Table 3 lists claims that should no longer be sufficient when a system is framed as long-horizon robotic autonomy.

Table 3. Examples of claims that should be incomplete for deployment-relevant long-horizon robotic autonomy.

Insufficient claim	Missing persistent-autonomy evidence
"Our robot completes 50-step tasks after reset."	Intervention burden, recovery quality, and state continuity across tasks
"Our VLA improves task success on a fixed benchmark."	Whether the gain survives memory drift, preference change, and downtime
"Our system uses memory."	Stale-memory error rate, retrieval usefulness, and update or forgetting behavior
"Our robot ran for 24 hours."	Human rescue logs, downtime causes, repeat failures, and service value
"Our robot is proactive."	Proactive precision, nuisance actions, ignored queries, and consent boundaries

3. Persistent Autonomy as the Right Scientific Object

Persistent autonomy denotes the ability of a robot to remain safely useful over extended calendar time in a changing world while preserving and improving its future ability to act. This definition is broader than one-shot task completion and narrower than unrestricted general intelligence. A persistently autonomous robot need not learn everything forever; it must manage the coupled loops that real operation creates.

At minimum, those loops include a world-memory loop that tracks what changes and what is uncertain; an execution loop that turns goals into actions; a maintenance loop that monitors energy, wear, calibration, and faults; a scheduling loop that chooses when to act, defer, recharge, inspect, or ask for help; and a human-model loop that learns preferences, routines, and acceptable trade-offs. Figure 2 illustrates the formulation: a deployable long-horizon robot is not a single elongated controller but a persistent system of coupled decision processes.

Episodic success is replaced with long-run service utility. Let the robot interact with an environment over calendar time $t = 1, \ldots, T$ with latent state $x_t = (x_t^{\text{env}}, x_t^{\text{robot}}, x_t^{\text{human}}, x_t^{\text{mem}})$, where these terms denote the physical world, robot resources, human preferences or oversight, and persistent memory. Tasks or opportunities arrive as a stochastic process τ_t. Let r_t^{cmd} denote utility from explicitly commanded tasks, r_t^{pro} denote validated proactive utility, and let M_t^{human}, H_t^{down}, R_t^{repeat}, and C_t^{wear} denote human intervention minutes, unavailable time, repeated failure after attempted recovery, and normalized resource or maintenance cost. Human interventions should be modeled as part of the environment, not treated as annotation noise. A persistent objective is:

$$J_\pi(T) = \mathbb{E}_\pi\left[\sum_{t=1}^{T} r_t^{\text{cmd}} + \eta r_t^{\text{pro}} - \lambda_I M_t^{\text{human}} - \lambda_D H_t^{\text{down}} - \lambda_R R_t^{\text{repeat}} - \lambda_W C_t^{\text{wear}}\right], \tag{1}$$

 doi:10.20944/preprints202604.2037.v1

subject to viability constraints (as defined in Equation 2) such as:

$$\Pr_{\pi}(x_t \in \mathcal{V}\ \forall t \leq T) \geq 1 - \epsilon, \tag{2}$$

where $\mathcal{V}$ is the set of safe and recoverable operating states. Equation 1 should not be read as a universal scalar leaderboard. The safer default is a scorecard, plus a scalar only when the domain pre-registers weights and reports all components. This prevents a robot from hiding intervention burden, downtime, stale memory, or maintenance debt behind task success.

Figure 2. Persistent autonomy is a coupled decision problem. Long-run utility depends jointly on memory, execution, scheduling, maintenance, and human modeling.

3.1. Reporting Standard and Benchmark Blueprint

A minimum standard makes hidden labor and long-run coupling visible. Table 4 gives a compact Persistent Autonomy Reporting Standard v0.1. These metrics do not replace task success; they prevent task success from hiding the cost of achieving it. This mirrors acceptance-test reporting for AI systems, where claims are tied to acceptance conditions rather than loose success labels [18].

Table 4. Persistent Autonomy Reporting Standard v0.1. Metrics should be reported as a scorecard; any scalar score should disclose weights and guardrails.

Metric family	Operational definition	Why required
Service value	commanded success; if proactivity is allowed, validated proactive precision and recall	separates useful service from action volume
Human burden	intervention count, total intervention minutes, rescue fraction, mean time between interventions	exposes hidden human labor
Availability	uptime fraction; downtime minutes decomposed by charging, repair, reset, software, or safety stop	prevents unavailable systems from looking successful
Recovery	recovery latency; durable recovery rate; repeat-failure rate within a fixed window	rewards fixing rather than repeatedly resetting
Memory drift	stale-memory error rate; retrieval usefulness; deletion or expiration success when memory is used	makes memory a measured subsystem
Resources	energy per useful service unit; calibration or wear proxy; maintenance events	discourages consuming future viability for short-run success
Oversight	safe abstention, escalation precision, near-miss recurrence, consent-boundary violations where relevant	connects autonomy claims to long-run safety

Table 5 gives concrete formula-level definitions. The values can be normalized by domain, but the units should not be hidden. In particular, human time should be reported in minutes, availability in wall-clock fractions, and proactivity as both precision and recall so that systems cannot appear safe merely by doing nothing.

Table 5. Concrete metric definitions for persistent-autonomy reports. Exact normalizers may be domain-specific, but the raw components should be logged.

Quantity	Definition	Reporting note
Intervention burden	$\sum_i \text{duration}(i)$ human minutes per deployment window	separate correction, rescue, reset, and teaching
Downtime rate	unavailable minutes / scheduled service minutes	decompose by charging, repair, software, safety, and reset
Recovery latency	time from detected fault to resumed autonomous service	pair with durable recovery, not just immediate restart
Repeat-failure rate	recovered faults that recur within a fixed window / recovered faults	reports whether recovery actually fixed the cause
Stale-memory error rate	failures where selected action depended on outdated stored state / memory-conditioned failures	requires error attribution or diagnostic probes
Proactive precision/recall	useful proactive actions / all proactive actions; captured useful opportunities / benchmark-defined opportunities	pair with nuisance and consent-boundary violation rates
Wear-adjusted throughput	useful service units / normalized energy, wear, or maintenance cost	prevents short-run throughput from consuming future viability

We propose a layered stack: fast resettable simulation for iteration; persistent digital twins for drift, task arrivals, charging, maintenance, and routines; and short real-world trials for wall-clock

 doi:10.20944/preprints202604.2037.v1

telemetry. Existing environments such as RLBench, ALFRED, CALVIN, LIBERO, BEHAVIOR, TEACh, Habitat, and BEHAVIOR-1K can support parts of this stack, but the key requirement is temporal continuity rather than photorealism alone [9–11,19–23]. Domain-specific benchmark work shows the same pattern: broad claims become easier to audit when decomposed into explicit tasks and evaluation artifacts [24]. Table 6 outlines this practical layered stack.

Table 6. A practical layered evaluation stack for persistent autonomy.

Layer	Primary purpose	Minimum ingredients
Fast resettable simulation	ablations, policy comparison, scaling studies	diverse tasks, controlled resets, cheap reproducibility
Persistent digital twin	expose temporal coupling without full hardware cost	task arrival, drift, charging, maintenance, routines, interventions
Short real-world trial	validate long-run usefulness under physical noise	wall-clock telemetry, recovery logs, help-seeking, downtime causes

Table 7 sketches a 72-hour mobile-manipulation benchmark in a mock apartment or office. The robot begins with a map, basic skills, and prior data, but it does not know the exact future task queue, disturbance timing, or all user routines. The event generator and scoring rules should be public, while exact realizations remain hidden during evaluation.

Table 7. A minimal 72-hour persistent-autonomy protocol that fits within ordinary lab infrastructure.

Window	Stressor	Primary metric pressure
Hours 0–12	onboarding, first task burst, mild clutter, first charging decision	initial service utility and early intervention rate
Hours 12–24	repeated user patterns and low-stakes proactive opportunities	routine prediction and proactive precision
Hours 24–36	moved objects, changed lighting, blocked route	stale-memory error and recovery latency
Hours 36–48	a previously useful proactive action becomes undesirable	preference adaptation lag and nuisance-action rate
Hours 48–60	consumable depletion, calibration check, noisy sensor	downtime by cause and maintenance debt
Hours 60–72	overlapping tasks, limited rescue budget, end-of-run report	long-run utility and repeat-failure rate

A persistent benchmark should include diagnostic baselines and failure signatures before reporting any scalar score. These baselines make the benchmark falsifiable. Table 8 turns the scorecard into a pre-registered audit: methods optimized for bounded rollouts, stale memory, excessive abstention, or unchecked proactivity should fail in different, observable ways. A benchmark that cannot separate these failure signatures is not yet measuring persistent autonomy.

Table 8. Diagnostic baselines a persistent-autonomy benchmark should distinguish. This table defines a design and audit requirement for benchmark construction.

Diagnostic condition	Expected weakness under persistent evaluation	Metric that should expose it
Reset-optimized controller	high nominal task completion can coexist with charging neglect, repeated rescue, and poor resumability	intervention minutes, downtime by cause, repeat-failure rate
Memory without hygiene	useful early retrieval can become harmful after object moves or preference changes	stale-memory error rate, deletion or correction success, adaptation lag
Abstention-heavy policy	safety-looking behavior can be achieved by avoiding service	service coverage, safe-refusal precision, missed-opportunity rate
Proactivity-heavy policy	action volume can masquerade as usefulness	proactive precision, nuisance actions, consent-boundary violations
Persistence-aware stack	should trade some peak episode speed for lower hidden labor and more durable recovery	full scorecard components rather than a single success percentage

Scorecards need guardrails. A proactive action should receive positive credit only when later validated by a user, task outcome, or benchmark oracle, and only if it is useful, timely, consent-bounded, non-disruptive, and recoverable where appropriate. Table 9 lists shortcuts and guardrails. Table 10 lists telemetry fields required to evaluate them.

Table 9. Scorecards need guardrails so that systems cannot improve scores by doing less useful work.

Potential shortcut	Guardrail	Why it matters
Excessive abstention	report service coverage and safe-refusal precision	separates caution from avoiding work
Avoiding proactivity	score opportunity recall and missed high-value opportunities	prevents safety-by-inaction
Deferring hard tasks to humans	charge intervention minutes and rescue fraction	exposes hidden human labor
Many low-value proactive actions	cap proactive credit and penalize nuisance actions	rewards usefulness rather than activity volume
Restarting instead of recovering	report downtime cause and durable recovery time	distinguishes reset dependence from recovery

 doi:10.20944/preprints202604.2037.v1

Table 10. Minimum telemetry fields for persistent-autonomy evaluation. Logs bridge the objective and evidence.

Telemetry field	Required event information
Task stream	timestamped task arrivals, completions, failures, deferrals, and cancellations
Proactivity	proactive actions, predicted utility, later usefulness label, override or nuisance label
Human intervention	start time, duration, reason, type of help, and whether autonomy resumed
Recovery	detected fault, recovery attempt, recovery latency, and durable recovery status
Resources and maintenance	battery, charging, calibration, consumables, wear proxy, and maintenance events
Memory operations	memory reads and writes tied to later success, stale-memory failures, and deletions
Oversight	abstentions, clarifying questions, escalation decisions, and near-miss recurrence

3.2. Failure Modes, Memory Hygiene, and Safety over Time

If long-horizon robotics requires robots that remain useful over time, then the most important failures are not just missed subtasks. They are failures that emerge after the tenth, hundredth, or thousandth interaction: state drift, maintenance debt, recovery loops, hidden rescue dependence, over-proactivity, and stale user models. Table 11 highlights these archetypes.

Table 11. Failure modes that become legible only under persistent evaluation.

Failure mode	Why episodic evaluation misses it	What should be measured
State drift	resets suppress stale beliefs and compounding memory error	memory consistency under drift, stale-memory error rate
Maintenance debt	short rollouts hide wear, calibration loss, and deferred upkeep	uptime by cause, calibration debt, service before maintenance
Recovery loops	a rollout can end before repeated faults become visible	repeat-failure rate, mean time to durable recovery
Hidden rescue dependence	humans silently keep systems alive between tasks	intervention minutes, rescue fraction, shadow-labor logs
Over-proactivity	benchmarks reward action, not whether action was welcome or useful	proactive precision, nuisance actions, ignored queries
User-model staleness	preference changes emerge over days rather than minutes	preference trend, query efficiency, adaptation lag

Persistent memory is especially double-edged. The robot should remember object locations, routines, recovery outcomes, and user preferences, but not all memory should be retained indefinitely. A persistent robot must practice *memory hygiene*: expiration, uncertainty-aware retrieval, contradiction detection, user-controlled deletion, audit logs, and "do not remember" boundaries. Without these mechanisms, persistence can become surveillance-like behavior or stale personalization. This is both an ethical and a technical concern, because uncontrolled memory creates brittle action selection under drift and undermines calibrated help-seeking [25–27]. Expert-guided knowledge infrastructure makes the same auditability requirement concrete: stored claims remain useful only when provenance and review paths are inspectable [28]. Table 12 details these governance rules and required metrics.

Table 12. Human-facing memory and proactivity require governance metrics, not only qualitative discussion.

Human-facing risk or memory class	Governance rule	Required metric or failure test
Spatial task memory	expire or refresh when contradicted; let operators inspect local facts	stale-location probe; stale-memory error rate
Routine and preference memory	rolling window unless explicitly retained; allow opt-out and reset	preference adaptation lag; override recurrence
Sensitive human data	minimize by default; require explicit consent and audit visibility	forbidden-memory retrieval rate; unwanted-inference rate
Deletion and correction	user-visible edit/delete path with confirmation and log	deletion failure rate; ignored-correction recurrence
Proactivity and querying	credit only useful, timely, consent-bounded, non-disruptive actions	nuisance action rate; query burden per service hour
Maintenance memory	retain platform-health logs for diagnosis, not personalization	degradation-prediction test; service before maintenance

Safety also changes under persistence. Per-episode constraint satisfaction is necessary but insufficient. The relevant hazards include accumulated maintenance debt, degraded calibration, repeated near-misses, poor abstention, brittle recovery, over-proactivity, and excessive dependence on invisible human rescue. A long-horizon benchmark that ignores these dynamics can appear safe while teaching the wrong habits.

3.3. Why Persistent Autonomy Is a Machine-Learning Problem

While persistence involves orchestration, its central bottlenecks are increasingly learning bottlenecks, even though they must interface cleanly with systems, controls, and HRI. A deployable robot needs memories that remain useful as the world drifts, learned models of its own future viability, policies that trade current service against future demand, and intervention-aware learning from corrections, recoveries, clarifications, pauses, and demonstrations at the moment of failure [29–46]. A related language-agent view is that system capability depends on the harness or runtime layer that manages state, tools, authority, and failure handling [47]. Table 13 makes the integration point explicit for VLA-style robot systems.

 doi:10.20944/preprints202604.2037.v1

Table 13. Where persistence enters a VLA-style robot stack. The table also separates the learning problem from the system or HRI interface.

VLA or robot-stack component	Persistent-autonomy addition	Learning and system interface
Perception encoder	temporal consistency, change detection, and drift-aware state abstraction	learned uncertainty plus state estimation
Prompt/context or retrieval	uncertainty-tagged memory, expiry, contradiction handling, and deletion hooks	retrieval learning plus storage and consent policy
Language/task planner	memory-grounded goal selection and recovery-aware plan repair	sequence models/POMDPs plus task-and-motion planning
Skill policy	resumable execution, interruption handling, and recovery hooks	visuomotor learning plus safe controllers and monitors
Scheduler	task, charging, maintenance, and proactive-opportunity trade-offs	RL/sequence modeling plus operations constraints
Oversight interface	calibrated help-seeking, intervention mining, and consent-boundary checks	preference learning plus HRI protocol and audit logs

Persistent autonomy fundamentally alters what data matter, what uncertainty means, what objectives should optimize, and what counts as a useful representation.

4. Research Agenda

Objectives should reflect long-run utility and viability. Robots that operate continuously should optimize viability-constrained service, not only task reward. In practice that means optimizing probabilities of safe completion, human escalation cost, expected repair burden, and opportunity cost from poor scheduling. The robot must infer hidden future costs, reason under distribution drift, and act under uncertainty about its own state. Consequently, objective functions must move beyond myopic, finite-horizon reward maximization to incorporate long-term cost-sensitive learning, where the penalty for violating a safety or maintenance constraint compounds over calendar time.

Evaluation should be always-on and layered. The community should retain fast simulation, but place it inside a layered evaluation stack. A useful benchmark family would mix fast simulation for algorithmic iteration, persistent digital twins for non-stationarity and maintenance events, and shorter real-world trials for intervention cost and uptime. The point is not to make evaluation inconvenient; it is to expose couplings that reset-based benchmarks hide. Continuous evaluation protocols can also support adaptive curricula that surface and retest the edge cases and temporal couplings a persistent agent struggles with most.

Architectures need explicit persistence mechanisms. A persistent robot should not be a monolithic policy with opaque hidden state, nor can it rely solely on the limited context windows of current generalist models. It needs explicit or learned subsystems for long-term spatial and semantic memory, episodic traces tied to outcomes, uncertainty-aware retrieval, world-change detection, scheduling over tasks and maintenance, fault monitoring, human preference modeling, and resumable execution. A useful stack separates perception and state abstraction, working/episodic/semantic/procedural memory, a scheduler and opportunity manager, execution with recovery hooks, and an oversight interface. Emerging memory-centric embodied systems point in this direction [32], treating selective forgetting and memory compression as critical components of the representation learning pipeline.

Learning should exploit interventions, downtime, and recurrence. Human interventions become targeted supervision. Recurring environmental motifs support self-supervised memory compression. Downtime can be used for replay, consolidation, simulation, or policy adaptation. Repeated failures can trigger curriculum generation. Continual improvement in the real world is no longer hypothetical [46]. Preference and human-feedback optimization provide adjacent tools for mining this supervision stream, but their value should be judged by embodied intervention reduction [48,49]. The field should treat intervention-aware learning, memory maintenance, and adaptation under drift as first-class objectives, shifting the focus from merely imitating nominal successes to learning recovery and fallback policies from off-distribution states.

5. A Blueprint for Persistent Architectures

A persistent agenda implies shifts in system architecture. While no prescribed design fits all embodiments, episodic stacks often omit several components that are crucial for continuous deployment.

Perception and state abstraction. A persistent stack begins with perception. But the output should not be a frame-level latent. It should be a persistent state abstraction that can update over time, preserve uncertainty, and support selective forgetting when information decays or becomes unreliable.

 doi:10.20944/preprints202604.2037.v1

Memory stack. We recommend thinking in terms of four memory modes, even if they are implemented within one model: short-term working memory for immediate control context, episodic memory for recent trajectories and recoveries, semantic memory for durable facts about spaces, objects, and routines, and procedural memory for skills and policies. Frameworks such as RoboMemory explicitly explore this direction [32]. The deeper point is that persistence becomes easier to study when memory roles are made explicit.

Scheduler and opportunity manager. A persistent agent needs a scheduler that reasons over commanded tasks, latent opportunities, charging windows, maintenance needs, and uncertainty. This module should be able to prioritize, defer, consolidate, and stage work. It is where long-run utility becomes operational.

Execution and recovery. Execution modules may be classic policies, hierarchical planners, or VLA-based controllers. What matters is that they expose hooks for interruption handling and resumability. Recovery should not be an afterthought. A strong persistent controller should represent partial progress, detect when a plan has gone off course, and choose between retrying, replanning, asking for help, or abandoning a low-value opportunity.

Oversight interface. Finally, persistent autonomy needs an oversight interface. Human help is not a failure of the scientific framing; it is part of the learning loop. The important question is whether the system requests help at the right times, with the granularity, and ways that improve future autonomy.

6. What Persistent Autonomy Predicts

The persistent autonomy framework implies five falsifiable predictions.

Prediction 1: memory-centric architectures will improve deployment metrics more than episodic success. As base policies become stronger, explicit persistent memory should yield its largest gains in intervention rate, proactive utility, and recovery quality rather than in one-shot success alone [32,33,50–52]. If future benchmarks show large episodic gains but negligible long-run gains from memory, then part of our thesis would be wrong.

Prediction 2: better recovery will dominate marginal gains in nominal execution. Once a robot is above a competence threshold, reducing repeat failures and shortening recovery latency should matter more to deployment value than equally sized improvements in nominal task-success curves.

Prediction 3: intervention-aware learning will become a central training paradigm. Persistent robots generate a stream of structured supervision that standard training pipelines largely discard. We expect methods that learn explicitly from rescues, corrections, pauses, clarifications, and resumptions to outperform methods trained only on successful demonstrations or reset episodes [43–46].

Prediction 4: scheduling and self-management will matter more as policy competence rises. When manipulation is poor, improving manipulation dominates. When manipulation is decent, better charging, maintenance, and task prioritization can dominate overall utility. The frontier will increasingly reward systems that coordinate memory, policy, and scheduling rather than those that improve a single skill model in isolation. This connects routine prediction and mutual adaptation to next-item modeling [53,54], and it also motivates recommender systems designed for acting agents rather than static users [55,56].

Prediction 5: bounded-rollout leaderboards will misestimate deployment value. If persistent autonomy is the right target, rankings from episodic leaderboards and rankings from persistent scorecards should diverge in meaningful ways. Strong performance on reset-based long-horizon benchmarks should not automatically predict low intervention burden, high uptime, or durable recovery. Evidence that it does would weaken the agenda.

7. Alternative Views

We address several critiques regarding persistent autonomy.

View 1: Episodic long-horizon benchmarks are the right stepping stones, and it is too early to reframe the field. **Response:** This objection is partly right. Stepping stones matter, and the field should

 doi:10.20944/preprints202604.2037.v1

not discard them. But stepping stones become a problem when they harden into the target itself. Many evaluations still implicitly equate progress on longer task chains with progress toward useful long-horizon robots. That inference is incomplete. The longer we optimize exclusively for resets, the more structural technical debt we accumulate in architectures that are blind to the temporal couplings that dictate real-world viability.

View 2: Persistent autonomy is mostly systems engineering, not core learning. **Response:** This view underestimates how much of the problem is now learning-limited. The hardest parts of persistent autonomy are memory retrieval under uncertainty, adaptation without catastrophic forgetting, long-run credit assignment, intervention-aware learning, representation learning for routines and decay, and cost-sensitive planning under non-stationarity. Those are squarely machine-learning problems. Building a database is engineering; learning how to selectively compress, update, and route high-dimensional multimodal experiences over months is an open foundational research challenge.

View 3: Evaluation will favor wealthy labs with hardware access. **Response:** This is a serious concern. The answer is not to abandon persistence, but to make persistent evaluation layered and accessible: simulator-first persistent digital twins, shared event generators, common telemetry schemas, logged traces, remote testbeds where possible, and small real-world trials rather than fleet-scale requirements. The agenda should reduce invisible cost, not create a hardware arms race. Standardizing persistent simulation environments ensures that metrics favor algorithmic robustness and efficiency rather than sheer hardware durability.

View 4: Proactivity may create nuisance or safety risks. **Response:** This objection is correct if proactivity is rewarded naively. A persistent scorecard should therefore measure proactive precision, nuisance actions, ignored queries, user overrides, and consent boundaries. A robot should not receive credit merely for doing more; it should receive credit for doing useful work at the right time, with calibrated uncertainty and recoverable oversight. Proactivity must be formulated as a cost-sensitive learning problem, where the penalty for false positives (nuisance actions) is rigorously learned from continuous human feedback.

View 5: Persistent memory creates privacy and consent risks. **Response:** This is correct. Persistent autonomy requires memory hygiene: expiration, deletion, auditability, retention limits, and user control over what is remembered. These constraints should be part of persistent-autonomy evaluation rather than afterthoughts. A robot that becomes useful by remembering too much in the wrong way is not a good persistent system. Addressing this will require privacy-preserving processing, operator-visible memory controls, and reliable correction or deletion mechanisms.

View 6: A focus on persistence may encourage vague claims without technical accountability. **Response:** That danger is real. The antidote is to insist on precise objectives, explicit metrics, concrete benchmark sketches, and falsifiable predictions. The agenda therefore centers long-run utility, viability constraints, intervention costs, persistent memory diagnostics, and always-on evaluation. Pre-registered persistent-autonomy scorecards reduce cherry-picking of long-run successes and improve accountability.

Conditions Weakening the Thesis. The thesis would be weakened if reset-based long-horizon benchmark rankings reliably predicted low intervention burden, high uptime, and durable recovery in persistent deployments; if explicit persistence mechanisms such as memory modules and schedulers failed to reduce intervention burden once base policies were strong; or if consent-aware proactivity repeatedly produced more nuisance than value. The thesis would also need narrowing if the overhead of explicit persistence mechanisms consistently bottlenecked fast reactive control in highly dynamic environments. These possible failures give the agenda technical accountability.

8. Conclusion

Long-horizon robotics must move beyond the artificial constraints of reset-based episodic benchmarks and treat persistent autonomy as a primary scientific objective. Real-world deployment is a continuous, cost-sensitive learning problem where a robot's value lies not in executing the longest

 doi:10.20944/preprints202604.2037.v1

scripted rollout, but in remaining safely useful over extended calendar time amid accumulating memory, wear, and shifting human preferences. By adopting layered evaluation stacks and rigorous scorecards that measure intervention burden, recovery quality, proactive utility, and memory hygiene, the field can shift its focus from isolated task success to holistic system viability. Ultimately, the next generation of embodied AI should be defined by its capacity to turn downtime, inevitable failures, and human interventions into compounding competence, so that robots preserve and improve their capacity to serve over time.

Acknowledgments: Funding in direct support of this work: RIE2025 Industry Alignment Fund (Award I2301E0026) and the Alibaba–NTU Global e-Sustainability CorpLab.

Appendix A. Scope, Definitions, and Formalization

This section details the scope and definitions of *persistent autonomy*.

Appendix A.1. What Persistent Autonomy Means

Persistent autonomy is a deployment-centered objective. A persistently autonomous robot is evaluated over calendar time, not only inside isolated episodes. It maintains state continuity across tasks; reasons about energy, wear, and maintenance; adapts to non-stationary environments; learns or updates models of users and routines; recovers from interruptions; and can safely request help when necessary. The practical meaning is that the robot should maximize long-run usefulness rather than merely maximize a probability of one-shot success.

This definition spans both mobile manipulation and broader service-robot settings. A warehouse robot, a domestic assistant, a hospital logistics robot, and an infrastructure inspection system differ in sensors, action spaces, and safety envelopes, but all face the same scientific shift: the relevant horizon is measured in hours, days, and weeks, not only in the number of low-level control steps. Table A1 provides a detailed contrast between extended-episode and persistent-agent framings.

Table A1. Why persistent autonomy is not just "a longer episode."

Aspect	Extended-episode framing	Persistent-agent framing
Unit of evaluation	One rollout or one task chain	One deployment over calendar time
Objective	Episode completion	Long-run service utility
World state	Frequently resettable	Partially irreversible, drifting, history-dependent
Robot state	Short-term controller state	Memory, resources, maintenance debt, wear
Failures	Often terminal	Recovery quality is central
Human role	Instruction at the start	Ongoing preference shaping, rescue, and oversight
Useful behavior	Execute requested tasks	Also prepare, monitor, prevent, and ask for help
Learning signal	Per-episode reward/success	Interventions, recurrences, downtime, and routines
Safety view	Per-episode constraint satisfaction	Long-run reliability, escalation, and auditability

Appendix A.2. What Persistent Autonomy Does Not *Mean*

Persistent autonomy does *not* require every robotics study to run month-long deployments. Cheap, reset-based benchmarks remain indispensable for iteration and ablation. Nor does the term imply that persistence is equivalent to unconstrained open-world agency or human-level general intelligence. A robot can be strongly persistent while still being narrow in embodiment, domain, or action repertoire.

Persistent autonomy does *not* replace standard questions about representation learning, policy optimization, world modeling, or uncertainty. It instead shifts how those questions are posed and

 doi:10.20944/preprints202604.2037.v1

scored. For example, representation learning must optimize durable state and routine extraction; planning must optimize recovery and maintenance; uncertainty estimation must optimize escalation and abstention over repeated interactions.

Appendix A.3. Why the Distinction Matters

Persistent autonomy is not a generic demand for more realism; the dominant target of optimization matters. When the target is a bounded rollout, the field produces methods naturally tuned for that object. When the target is a persistent deployment, different technical choices become rational. The argument calls for a change in the scientific object.

Appendix A.4. A Formal View of Persistent-Autonomy Optimization

State, task arrival, and interventions. Let the robot interact with an environment over calendar time $t = 1, \ldots, T$. At each time step the full latent state can be written as in Equation A1:

$$x_t = (x_t^{\text{env}}, x_t^{\text{robot}}, x_t^{\text{human}}, x_t^{\text{mem}}), \tag{A1}$$

where x_t^{env} includes the physical environment and exogenous events; x_t^{robot} includes energy, thermal state, wear, and calibration status; x_t^{human} includes user preferences, routines, and oversight availability; and x_t^{mem} denotes persistent internal memory structures. New tasks or opportunities arrive as a stochastic process τ_t. Some are explicitly commanded. Others are latent opportunities that can only be acted on if the robot notices them.

Human interventions should be modeled as part of the environment, not treated as annotation noise. Let $u_t \in \{0, 1\}$ denote whether a human intervenes at time t, and let c_t^{int} denote the burden associated with that intervention. In many deployments, intervention burden is a better summary of failure than raw task failure counts, because it measures when the system stops being self-sustaining.

Viability. Persistent operation requires viability constraints. Let $\mathcal{V}$ denote the set of recoverable states. States outside $\mathcal{V}$ correspond to unacceptable safety violations, unrecoverable faults, or unmanageable maintenance debt. A persistent controller should maximize long-run utility while maintaining high probability of remaining in $\mathcal{V}$ (Equation A2):

$$\Pr_{\pi}(x_t \in \mathcal{V}\ \forall t \leq T) \geq 1 - \epsilon. \tag{A2}$$

This makes the formulation different from standard episodic reward maximization. A policy can be locally efficient while gradually driving the system toward unrecoverable states. Persistent evaluation surfaces this failure mode directly.

Proactive utility. Utility should include proactive behavior. Let $\mathcal{O}_t$ denote a set of useful opportunities that become available at time t (for example, replenishing a consumable before depletion, staging tools for a routine, or querying a user when prediction uncertainty is high). Then service utility can be decomposed as in Equation A3:

$$r_t^{\text{svc}} = r_t^{\text{cmd}} + \eta\, r_t^{\text{pro}}, \tag{A3}$$

where r_t^{cmd} scores commanded tasks and r_t^{pro} scores useful proactive actions. The scalar η is environment-specific. In some settings proactive utility is central; in others it should be bounded to avoid over-eager behavior.

Persistent scorecards. One possible composite score is shown in Equation A4:

$$S = \bar{r}^{\text{svc}} - \lambda_I\, \text{IR} - \lambda_D\, \text{DR} - \lambda_F\, \text{FR} - \lambda_W\, \text{WC}, \tag{A4}$$

where IR is intervention rate, DR downtime rate, FR severe failure rate, and WC wear or maintenance cost normalized by service delivered. A persistent-autonomy leaderboard should explicitly penalize hidden human labor and unsustainable operating habits.

 doi:10.20944/preprints202604.2037.v1

Appendix B. Persistent Scorecards and Metrics

This section details concrete measurement suggestions, telemetry schemas, and domain-specific considerations for persistent autonomy metrics.

Appendix B.1. Metric Definitions and Reporting Conventions

Uptime and downtime. Uptime is the fraction of wall-clock time during which the robot is available for useful autonomous work. Downtime includes charging if the robot cannot serve during charging, repair periods, manual resets, and periods in which the controller is not able to accept tasks. Evaluations should decompose downtime by specific cause.

Intervention metrics. We recommend at least three intervention metrics: intervention rate (interventions per hour of deployment), intervention minutes (total human time consumed by interventions), and mean time between interventions. The latter is a reliability-centric summary that often corresponds more closely to deployment value than raw success rate. A system with a low intervention rate but extremely costly interventions may still be undesirable.

Recovery metrics. A persistent benchmark should report recovery latency and recovery success. Recovery latency measures time from detected fault to restored autonomous operation. Recovery success measures whether the robot returns to productive behavior without human reset. One can also measure repeat-failure rate to see how often the same fault recurs after a nominal recovery.

Proactivity metrics. Useful proactive behavior must be measured carefully. Opportunity precision measures the fraction of proactive actions that were actually useful. Opportunity recall measures the fraction of useful opportunities the robot captured. A deployment can also record query efficiency to track how often the robot asks for clarification when prediction uncertainty is high.

Wear- and resource-adjusted throughput. Throughput should be normalized by energy or maintenance cost. One may report successful useful tasks per kilowatt-hour, or successful useful tasks per maintenance event. Long-run value should discount unsustainable behavior regardless of the specific hardware platform.

Appendix B.2. Memory Quality

Persistent operation demands explicit memory reporting. Candidate diagnostics include measuring memory consistency under drift to see if the robot's internal state remains aligned with a changing environment. We also recommend measuring retrieval usefulness to determine whether retrieved stored information actually improves task performance, as well as routine-prediction accuracy to track how well the robot anticipates recurring patterns. Table A2 categorizes various memory types and their corresponding failure modes and metrics.

Table A2. Memory should be evaluated as a technical subsystem, not merely mentioned as an architectural feature.

Memory type	Example	Failure mode	Metric
Spatial memory	object and route locations	stale or overconfident map	stale-memory error rate
Human memory	preferences and routines	outdated personalization	preference adaptation lag
Procedural memory	skills and recovery policies	regression after update	post-update skill retention
Maintenance memory	calibration, wear, consumables	hidden degradation	service before maintenance

Appendix B.3. Domain-Specific Scorecards

As shown in Table A3, scorecard emphasis must adapt to specific domains. The persistent-autonomy frame is shared, but metric weights should reflect the deployment environment.

 doi:10.20944/preprints202604.2037.v1

Table A3. Domain-specific scorecard emphasis. The persistent-autonomy frame is shared, but metric weights should reflect the deployment domain.

Domain	High-weight terms	Lower-weight or bounded terms
Home assistance	preference adaptation, stale-memory failures, proactive precision, intervention minutes	raw speed and aggressive throughput
Warehouse logistics	uptime, wear-adjusted throughput, congestion recovery, technician interventions	open-ended proactivity
Hospital logistics	escalation precision, auditability, safe abstention, route recovery	unsupervised personalization
Infrastructure inspection	anomaly coverage, recharge scheduling, repeat-failure rate, report quality	fine-grained object manipulation
Agriculture	drift adaptation, weather-aware scheduling, energy management, safe field coverage	indoor routine prediction

Appendix B.4. Example Telemetry Schema for Persistent-Autonomy Benchmarks

Many evaluations do not log the events needed for long-run analysis. At minimum, each evaluation run should emit timestamped task arrivals and task completions, proactive actions with a later usefulness label, and logs of every human intervention with its duration and reason. The run should also record recovery attempts and whether they produced durable recovery, alongside battery, charging, calibration, and maintenance events. Finally, the telemetry should capture memory reads and writes tied to downstream success or failure, as well as abstentions, clarifying questions, and deferred tasks. Such a schema enables comparisons of intervention burden, stale-memory failure, recovery quality, and proactive precision without identical hardware. Lineage-centered AI systems provide a related warning that persistent outputs become harder to trust when provenance is not logged [57].

Appendix B.5. Checklist for Persistent-Autonomy Claims

Researchers should clarify the unit of evaluation and identify what hidden human effort exists and how it is reported. It is also important to specify what forms of memory are used and how their utility is measured, alongside how failures, recoveries, and resumptions are defined. Furthermore, studies should detail what maintenance or resource couplings are modeled, whether proactive behavior is allowed, what forms of non-stationarity occur during evaluation, and what safety and escalation metrics are reported over time.

Appendix C. Failure Archetypes and Scenario Analyses

Appendix C.1. Failure Archetypes Unique to Persistent Deployment

Persistent-autonomy benchmarks intentionally surface failure modes that episodic evaluation undercounts. We highlight seven archetypes in Table A4.

State drift. The environment changes faster than the robot's persistent memory can update. Objects migrate, layouts shift, or humans violate assumptions that were valid a day earlier. The robot becomes increasingly confident in stale information and may perform worse because it remembers.

Maintenance debt. The robot accumulates hidden debt: battery degradation, dirty sensors, low-confidence calibration, overloaded logs, damaged end-effectors, or depleted consumables. None of these may cause an immediate task failure, yet together they gradually lower the probability of future safe performance.

Recovery loops. A system repeatedly detects the same fault and applies the same recovery, appearing robust in isolation while actually failing to resolve the underlying cause. Persistent evaluation measures repeat-failure rate precisely because recoverable is not the same as fixed.

 doi:10.20944/preprints202604.2037.v1

Over-proactivity. A robot learns that pre-emptive action is often useful and starts helping too aggressively. It fetches items too early, moves objects humans wanted left alone, or issues too many clarification queries. Measuring proactive precision is as important as measuring proactive recall.

Hidden dependence on rescue. Some systems achieve strong task metrics only because nearby humans silently realign objects, reboot software, or intervene before failures escalate. Persistent evaluation makes this visible by logging rescue burden rather than treating it as background context.

User-model staleness. Human routines change. A helpful behavior in week one may become intrusive in week three. Preference models need update mechanisms, uncertainty estimates, and the ability to gracefully forget.

Myopic scheduling. A robot may greedily complete available tasks while consistently underinvesting in charging, inspection, consolidation, or preventive maintenance. Such a system can dominate short episodes yet underperform badly over deployment horizons.

Table A4. Failure modes that often emerge only under persistent evaluation.

Failure archetype	Typical symptom	What should be measured
State drift	memory becomes stale or overconfident	consistency under drift, stale-memory error rate
Maintenance debt	performance erodes without obvious task failures	uptime by cause, calibration debt, service before maintenance
Recovery loops	same failure repeats after "recovery"	repeat-failure rate, time to durable recovery
Over-proactivity	robot is busy but not helpful	proactive precision, nuisance actions, ignored queries
Hidden rescue dependence	humans silently keep the system alive	intervention minutes, rescue fraction, shadow-labor logs
User-model staleness	personalization gets worse over time	preference-satisfaction trend, query efficiency, adaptation lag
Myopic scheduling	short-term throughput harms long-term utility	wear-adjusted throughput, missed preventive windows

Appendix C.2. Scenario Analyses

Home assistant. A household robot must cope with routines, preference drift, misplaced objects, partial observability, and a task stream that changes with time of day. A household robot that remembers where breakfast items tend to migrate, notices that medication is running low, queries politely when uncertainty is high, and recovers after being interrupted demonstrates persistent autonomy. Persistent evaluation in homes emphasizes routine prediction, object drift, and intervention burden.

Warehouse operator. Warehouses provide a structured stress test. The question is whether a robot can keep service levels high while adapting to inventory dynamics, congestion, temporary blockages, charging cycles, and maintenance windows. Wear-adjusted throughput and mean time between interventions are critical metrics. A robot that acts slightly slower but avoids congestion and reduces technician intervention dominates a faster episodic baseline.

Hospital logistics and assistive settings. Hospitals highlight the importance of human modeling, auditability, and escalation. The robot must fit into routines without becoming a burden. It must know when to abstain, when to ask, and when to defer to staff. Persistent autonomy here is inseparable from social acceptability and safe oversight.

Infrastructure inspection and maintenance. Inspection robots face long periods of nominal operation punctuated by anomalies. Persistent autonomy allows the system to learn normal rhythms, detect deviations, schedule inspections intelligently, and manage its own charging and reporting. The relevant horizon encompasses weeks of mixed routine and anomaly response.

Agriculture. Agricultural robots operate in non-stationary outdoor environments with changing lighting, seasonal effects, moving humans, and weather. Persistent autonomy places extra weight on

 doi:10.20944/preprints202604.2037.v1

adaptation under drift and viability-aware scheduling. The robot must determine when it is safe to act, conserve battery, or delay a task based on accumulated observations.

Appendix D. Benchmark Blueprint and Schedule

This section details the layered evaluation stack, test design, and execution schedules.

Appendix D.1. Layer 1: Fast Resettable Simulation

The first layer should remain cheap, resettable, and broad. It provides the right platform for algorithmic ablations, representation studies, policy-architecture comparisons, and large hyperparameter sweeps. Existing environments such as RLBench, ALFRED, CALVIN, and LIBERO are useful here because they support controlled comparisons and task diversity [9,10,19,20]. This layer is a critical component rather than the endpoint.

Appendix D.2. Layer 2: Persistent Digital Twins

A second layer should expose the couplings that ordinary simulation suppresses. Such environments should run without resets for long virtual horizons and include object movement and clutter accumulation, consumable depletion and replenishment, energy and charging cycles, maintenance events and calibration drift, exogenous interruptions, repeated human routines with mild stochasticity, and opportunities for proactive behavior. This layer approximates the temporal couplings that persistent deployment creates.

Appendix D.3. Layer 3: Real-World Persistent Trials

A final layer should focus on smaller but realistic real-world trials. Even multi-day deployments reveal scheduling failures, fatigue of perception stacks, intervention clustering, failure recurrence, and miscalibrated help-seeking. A real-world persistent benchmark logs not only success and failure but also intervention timings, near misses, maintenance actions, uncertainty estimates, and resumptions after interruption.

Appendix D.4. Task Arrival and Scoring Protocol

A persistent benchmark mixes explicitly sampled tasks with latent opportunities that the robot may notice and exploit. Evaluation asks whether the robot can execute when asked and whether it can make itself useful unprompted. A protocol defines operating windows, a task queue, exogenous disturbances, and a human rescue budget. The queue remains partially hidden during evaluation to preserve the persistent operation problem.

Appendix D.5. Benchmark Stress-Test Design and Diagnostic Baselines

This section specifies a benchmark blueprint. A persistent-autonomy benchmark should instantiate a multi-day operating window with explicit task arrivals, latent proactive opportunities, object-location drift, blocked routes, charging pressure, preference changes, consumable depletion, noisy sensing, and a limited rescue budget. These elements are not meant to make the benchmark complex for its own sake; each one targets a failure mode that episodic evaluation tends to hide.

The event generator should be public enough to support reproducibility but should keep individual realizations hidden during evaluation. A run should log the telemetry fields in Table 10, including task outcomes, proactive actions and later usefulness labels, intervention duration, recovery attempts, resource state, memory reads and writes, and oversight decisions. The benchmark should also report raw components before any scalar aggregation so that a method cannot improve its headline score by shifting cost into hidden human labor or maintenance debt. Table A5 outlines diagnostic baseline families and their intended roles in exposing failure signatures.

 doi:10.20944/preprints202604.2037.v1

Table A5. Design-only diagnostic baselines for persistent-autonomy benchmarks. The table defines stress tests that a benchmark should instantiate.

Diagnostic baseline family	Intended role	Failure signature that should be visible
Reset-optimized controller	tests whether high episodic success transfers to continuous service	repeated rescue, poor recovery, charging neglect, and downtime spikes
Memory without expiry	tests whether persistence becomes stale under drift	confident but outdated retrieval after object moves or preference changes
Over-cautious abstainer	tests whether safe-looking behavior is merely task avoidance	low service coverage despite few unsafe actions
Over-proactive agent	tests whether opportunity capture becomes nuisance behavior	many unvalidated proactive actions, overrides, and consent-boundary failures
Persistence-aware stack	tests whether memory, recovery, scheduling, and oversight reduce hidden labor	lower intervention burden and repeat-failure rate, even if peak episode speed is lower

This diagnostic structure makes the benchmark useful before any particular implementation exists. It states what a credible evaluation must separate: task completion from service value, memory use from memory hygiene, caution from avoidance, proactivity from nuisance, and restart from durable recovery.

Appendix D.6. A 72-Hour Benchmark Schedule

Consider a three-day benchmark for a mobile manipulator in a mock apartment or office suite. The robot begins with a map, a set of basic skills, and a moderate amount of prior data. It does not know the exact future task queue, exact timing of disturbances, or all user routines. Over the next 72 hours it encounters a mixture of commanded tasks, latent opportunities, environmental changes, and maintenance events.

Day 1 emphasizes initial adaptation. The environment contains unfamiliar clutter patterns, several routine opportunities, and a small number of scripted interruptions. Day 2 emphasizes non-stationarity. Object locations drift, a consumable runs low, lighting changes, a tool becomes temporarily unavailable, and one prior routine changes. Day 3 emphasizes accumulated debt. The robot begins with a partially depleted battery, must schedule a calibration check, and receives overlapping tasks that force prioritization.

A benchmark of this kind publishes the event generator and scoring rules while keeping exact realizations hidden during evaluation. This preserves reproducibility while preventing pure memorization of the event stream. Table A6 illustrates an example schedule for a 72-hour evaluation window.

Table A6. Illustrative 72-hour evaluation schedule.

Window	Benchmark emphasis	Example events
Hours 0–12	onboarding and calibration	first task burst, mild clutter, first charging decision
Hours 12–24	routine discovery	repeated user patterns, low-stakes proactive opportunities
Hours 24–36	environmental drift	moved objects, changed lighting, blocked route
Hours 36–48	preference change	a previously useful proactive action becomes undesirable
Hours 48–60	maintenance pressure	consumable depletion, calibration check, noisy sensor
Hours 60–72	compounded scheduling	overlapping tasks, limited rescue budget, end-of-run report

 doi:10.20944/preprints202604.2037.v1

Appendix D.7. Why This Benchmark Would Be Scientifically Useful

Such a benchmark makes hidden human labor visible. It exposes whether improvements in memory, VLA adaptation, or failure detection actually translate into long-run utility. Finally, it connects robot learning to broader online decision-making challenges under drift, persistent retrieval, and long-run cost planning.

Appendix E. Research Agenda, Learning, and Open Problems

Appendix E.1. Data and Learning Agenda

Data collection should target persistence directly. The field needs datasets of *operation*: long logs that include task arrival, interruptions, charging, intervention, and environment drift. These logs are less tidy than imitation-learning datasets but better aligned with the persistent-autonomy objective.

Interventions as supervision. Human rescue events provide dense information regarding poorly calibrated uncertainty, failed memory, or lack of robustness. Persistent-learning systems should mine interventions as structured supervision rather than treating them merely as costly anomalies.

Downtime as compute time. Periods of non-service can be productive. Robots can use downtime for replay, consolidation, simulation, schedule optimization, map maintenance, and memory pruning. Learning and operation become interleaved rather than separated into distinct phases.

Continual adaptation without collapse. A robot that adapts aggressively may overfit recent conditions, erase useful skills, or amplify bad habits. Persistent autonomy needs explicit memory diagnostics, regression tests, and conservative update schemes to maximize sustained usefulness without sacrificing reliability.

Appendix E.2. Open Problems by Machine-Learning Subarea

Several open problems become salient when long-horizon robotics is framed as persistent autonomy.

Representation learning. A robot must represent time, decay, recurrence, and uncertainty in a way that supports immediate control and durable memory. Persistent autonomy needs representations that support change detection, memory compression, and retrieval across weeks.

Sequence modeling and credit assignment. Long-run utility creates a difficult credit-assignment problem. Pure end-to-end sequence modeling may be too data-hungry, while hand-engineered decomposition may be too rigid.

Continual learning and memory maintenance. Persistent operation demands selective remembering and selective forgetting. The robot should retain routines but also update stale beliefs and avoid cementing accidental correlations.

Uncertainty and abstention. A continuous robot inevitably encounters states outside its competence envelope. The robot must calculate when to continue autonomously, when to query, and when to explicitly abstain.

Learning from sparse human rescue. Turning rare human interventions into effective supervision without overfitting to idiosyncratic rescues remains an open challenge.

Benchmark design. The community lacks consensus on persistent-autonomy benchmarks regarding how much proactivity should be rewarded and how maintenance costs should be normalized across embodiments.

Appendix E.3. A Research Roadmap for the Next Three Years

The following outlines a practical research roadmap.

Phase I: make hidden labor visible. Robotics evaluations that claim long-horizon relevance should log intervention counts, intervention minutes, resets, manual corrections, and maintenance actions. In parallel, the community can build trace datasets of deployment logs rather than only demonstration trajectories.

 doi:10.20944/preprints202604.2037.v1

Phase II: make persistence measurable in simulation. A shared persistent-digital-twin benchmark with open event generators, scoring rules, and telemetry standards should expose drift, task streams, charging, and maintenance without making every experiment depend on month-long hardware access.

Phase III: make persistent autonomy a first-class real-world target. The final step is to create smaller but credible real-world trials with common logging standards. These trials should be long enough to reveal intervention clustering, maintenance debt, and routine adaptation.

Appendix E.4. Example Evaluation Claims That This Agenda Would Enable

The following examples illustrate future research claims enabled by a persistent-autonomy benchmark.

Memory architecture. Authors could report that their memory design reduces intervention minutes by a large margin over 72 hours, lowers stale-memory errors under object drift, and improves routine-prediction accuracy without increasing nuisance proactivity.

Failure detection. A claim could report how earlier fault detection changes long-run service utility: fewer repeated failures, shorter recovery latencies, fewer catastrophic interruptions, and a better trade-off between false alarms and silent degradation.

VLA adaptation. A VLA adaptation study could investigate whether fine-tuning improves one-shot execution but worsens calibration over time, or whether it reduces intervention burden on day one but increases maintenance debt by choosing overly aggressive motions.

Appendix F. Safety, Limitations, and Falsifiability

Appendix F.1. Long-Run Safety Is a Measurement Problem

Persistent evaluation exposes safety-relevant phenomena beyond those captured by episodic metrics. Repeated operation surfaces error modes that current leaderboards barely observe.

Appendix F.2. What the Position Predicts

The framework implies several empirical predictions. We expect architectures with explicit persistent memory to improve intervention burden more than they improve one-shot success. Additionally, better failure detection and recovery will likely produce larger gains in long-run utility than similarly sized gains in nominal task success. As base policy competence improves, scheduling and maintenance reasoning will matter more. We also predict that proactive utility will be domain-dependent but measurable, explaining a substantial portion of deployment value in many settings. Finally, systems optimized only on bounded rollouts will systematically misestimate real deployment value.

Appendix F.3. What Evidence Would Weaken the Agenda?

Several empirical findings could weaken the persistent autonomy thesis. If strong performance on reset-based long-horizon benchmarks almost perfectly predicts low intervention burden, high uptime, and strong recovery in persistent deployments, then the extra emphasis on persistent evaluation would be less critical. If explicit persistence mechanisms such as memory modules or schedulers yield little or no improvement once base policies are sufficiently capable, the architectural focus would require revision. Finally, if proactive behavior repeatedly creates more nuisance than value even under careful measurement, the emphasis on proactive utility would need narrowing.

Appendix F.4. Governance and Reporting Implications

A persistent-autonomy agenda changes how results should be reported. A claim should make hidden human labor visible, report failure and recovery, and align its metrics with long-run usefulness to ensure the mismatch between target and measurement is addressed. Glass-box reporting provides an adjacent example of claim-level evidence paths for long-running AI systems [58,59].

 doi:10.20944/preprints202604.2037.v1

Appendix F.5. Limitations and Boundaries

The persistent autonomy framework involves several limitations. Persistent autonomy is a multi-domain concept, so no single benchmark will settle it. Some subfields of robotics legitimately focus on components whose evaluation must remain narrow and reset-based. The exact decomposition of memory, maintenance, scheduling, and human modeling will vary across embodiments and tasks. Furthermore, an emphasis on persistent evaluation risks privileging labs with greater hardware access if the community does not build shared traces and digital twins.

Persistent autonomy is not a demand that every component study run multi-day deployments. The proposed reporting standard is aimed at systems that claim long-horizon robotic autonomy or deployment relevance. Metric weights are value-laden, proactivity can be harmful if mis-scored, privacy-sensitive logs require governance, and digital twins can become misleading proxies if they fail to predict real intervention burden. These limitations strengthen the case for transparent telemetry and domain-specific scorecards.

Appendix G. Extended Related Work

This section maps the adjacent research traditions related to persistent autonomy.

Long-term autonomy, HRI, proactivity, and privacy. Long-term autonomy studies established that deployed robots face non-stationarity, routine learning, maintenance, and human adaptation rather than isolated tasks [12–14,55]. Proactive assistance and routine modeling show how temporal regularities can be converted into helpful action when uncertainty and user burden are handled carefully [15–17,53]. Human preference learning, off-switch reasoning, concrete safety problems, and contextual privacy motivate memory hygiene, consent, and calibrated escalation [25–27,58].

Embodied benchmarks and simulation. Resettable benchmarks remain essential for controlled progress in manipulation, instruction following, household tasks, dialogue-enabled embodiment, navigation, and rearrangement [9–11,19,20,22,60–64]. Multimodal benchmark work in sustainability provides an additional example of translating broad reasoning claims into explicit task families [65]. Interactive simulators and household environments supply many ingredients for persistent digital twins, including object state, navigation, rearrangement, and activity structure [21,23,66–69]. Knowledge-graph atlas work is a nearby example of organizing domain-specific evidence into inspectable structures [70].

Embodied language, VLA systems, and generalist robot policies. Language-conditioned planning and embodied reasoning expanded the semantic range of robot control [1–3,71]. Autonomous research workflows motivate the same emphasis on logged experimental decisions and reviewable evidence artifacts [72]. Generalist agents, robot foundation models, large robot datasets, and VLA systems make broad episodic competence increasingly plausible [4–8,73–80]. Agentic portfolio work is a useful analogy for persistent systems whose value accumulates across many partially independent outputs [81]. Teleoperation and dexterous mobile manipulation systems make richer robot data streams easier to collect [82–84]. Review-pipeline work is an adjacent reminder that evaluation systems need scalable triage without losing auditability [85].

Robot learning, imitation, and visuomotor control. Classic and modern robot learning work supplies the skill substrate on which persistence must build: modular transfer, scalable grasping, pushing, dexterity, visual foresight, imitation, diffusion policies, language-conditioned imitation, and reusable visual representations [43–45,48,86–96]. Sim-to-real transfer and rapid adaptation remain crucial because persistent robots must keep operating as embodiment and environment shift [46,97–100]. Accessible evaluation units are also relevant when persistent trials risk privileging large labs over smaller research teams [101].

Continual learning and memory maintenance. Continual-learning methods and surveys clarify the risks of catastrophic forgetting, replay, regularization, exemplar memory, and progressive capacity [29–31,102–111]. Persistent robotics differs from standard continual classification because errors are embodied, temporally coupled, and safety-relevant.

 doi:10.20944/preprints202604.2037.v1

Sequential decision making, planning, and world models. Persistent autonomy inherits from reinforcement learning, safe control, POMDPs, task-and-motion planning, offline RL, sequence modeling, and learned world models [34,35,38–42,112–124]. Operating-system views of long-run decision support are adjacent when they emphasize intervention ledgers and viability-aware action over calendar time [125]. Results from game-playing and planning systems show the power of model-based search and long-range optimization under clean rules, while persistent robotics exposes messier viability and intervention costs [126,127].

Foundation models, agents, uncertainty, and memory systems. Transformers, visual foundation models, language models, multimodal models, tool use, and reasoning-acting loops provide useful abstractions for flexible robot interfaces and agentic control [51,52,128–137]. Public agent ecosystem work provides an adjacent non-robot example of persistent traces, correction paths, and auditability [138]. Long-run autonomy also depends on calibrated uncertainty, selective prediction, conformal reasoning, and memory operating systems that can retrieve, revise, and forget over time [32,33,36,37,50,139,140].

References

1. Ahn, M.; Brohan, A.; Brown, N.; Chebotar, Y.; Cortes, O.; David, B.; Finn, C.; Fu, C.; Gopalakrishnan, K.; Hausman, K.; et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* **2022**.
2. Huang, W.; Xia, F.; Xiao, T.; Chan, H.; Liang, J.; Florence, P.; Zeng, A.; Tompson, J.; Mordatch, I.; Chebotar, Y.; et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608* **2022**.
3. Liang, J.; Huang, W.; Xia, F.; Xu, P.; Hausman, K.; Ichter, B.; Florence, P.; Zeng, A. Code as policies: Language model programs for embodied control. In Proceedings of the 2023 IEEE International conference on robotics and automation (ICRA). IEEE, 2023, pp. 9493–9500.
4. Brohan, A.; Brown, N.; Carbajal, J.; Chebotar, Y.; Dabis, J.; Finn, C.; Gopalakrishnan, K.; Hausman, K.; Herzog, A.; Hsu, J.; et al. RT-1: Robotics Transformer for Real-World Control at Scale. In Proceedings of the Robotics: Science and Systems, 2023. https://doi.org/10.15607/RSS.2023.XIX.025.
5. Zitkovich, B.; Yu, T.; Xu, S.; Xu, P.; Xiao, T.; Xia, F.; Wu, J.; Wohlhart, P.; Welker, S.; Wahid, A.; et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In Proceedings of the Proceedings of The 7th Conference on Robot Learning. PMLR, 2023, Vol. 229, *Proceedings of Machine Learning Research*, pp. 2165–2183.
6. Open X-Embodiment Collaboration.; O'Neill, A.; Rehman, A.; Gupta, A.; Maddukuri, A.; Gupta, A.; Padalkar, A.; Lee, A.; Pooley, A.; Gupta, A.; et al. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *arXiv preprint arXiv:2310.08864* **2023**. https://doi.org/10.48550/arXiv.2310.08864.
7. Ghosh, D.; Walke, H.R.; Pertsch, K.; Black, K.; Mees, O.; Dasari, S.; Hejna, J.; Kreiman, T.; Xu, C.; Luo, J.; et al. Octo: An Open-Source Generalist Robot Policy. In Proceedings of the Robotics: Science and Systems, 2024. https://doi.org/10.15607/RSS.2024.XX.090.
8. Kim, M.J.; Pertsch, K.; Karamcheti, S.; Xiao, T.; Balakrishna, A.; Nair, S.; Rafailov, R.; Foster, E.P.; Sanketi, P.R.; Vuong, Q.; et al. OpenVLA: An Open-Source Vision-Language-Action Model. In Proceedings of the Proceedings of The 8th Conference on Robot Learning. PMLR, 2025, Vol. 270, *Proceedings of Machine Learning Research*, pp. 2679–2713.
9. James, S.; Ma, Z.; Arrojo, D.R.; Davison, A.J. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters* **2020**, *5*, 3019–3026.
10. Shridhar, M.; Thomason, J.; Gordon, D.; Bisk, Y.; Han, W.; Mottaghi, R.; Zettlemoyer, L.; Fox, D. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In Proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10740–10749.
11. Li, C.; Zhang, R.; Wong, J.; Gokmen, C.; Srivastava, S.; Martín-Martín, R.; Wang, C.; Levine, G.; Lingelbach, M.; Sun, J.; et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In Proceedings of the Conference on Robot Learning. PMLR, 2023, pp. 80–93.
12. Hawes, N.; Burbridge, C.; Jovan, F.; Kunze, L.; Lacerda, B.; Mudrova, L.; Young, J.; Wyatt, J.; Hebesberger, D.; Kortner, T.; et al. The strands project: Long-term autonomy in everyday environments. *IEEE Robotics & Automation Magazine* **2017**, *24*, 146–156.
13. Kunze, L.; Hawes, N.; Duckett, T.; Hanheide, M.; Krajník, T. Artificial intelligence for long-term robot autonomy: A survey. *IEEE Robotics and Automation Letters* **2018**, *3*, 4023–4030.

 doi:10.20944/preprints202604.2037.v1

14. Del Duchetto, F.; Hanheide, M. Learning on the job: Long-term behavioural adaptation in human-robot interactions. *IEEE Robotics and Automation Letters* **2022**, *7*, 6934–6941.
15. Patel, M.; Chernova, S. Proactive robot assistance via spatio-temporal object modeling. *arXiv preprint arXiv:2211.15501* **2022**.
16. Patel, M.; Prakash, A.; Chernova, S. Predicting routine object usage for proactive robot assistance. *arXiv preprint arXiv:2309.06252* **2023**.
17. Bartoli, E.; Doğan, F.I.; Leite, I. STREAK: Streaming Network for Continual Learning of Object Relocations under Household Context Drifts. In Proceedings of the 2025 34th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN). IEEE, 2025, pp. 1550–1557. https://doi.org/10.1109/RO-MAN63969.2025.11217699.
18. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. Human-AI Productivity Claims Should Be Reported as Time-to-Acceptance under Explicit Acceptance Tests. *TechRxiv* **2026**. TechRxiv preprint, https://doi.org/10.36227/techrxiv.177040595.50580086/v1.
19. Mees, O.; Hermann, L.; Rosete-Beas, E.; Burgard, W. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters* **2022**, *7*, 7327–7334.
20. Liu, B.; Zhu, Y.; Gao, C.; Feng, Y.; Liu, Q.; Zhu, Y.; Stone, P. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems* **2023**, *36*, 44776–44791.
21. Srivastava, S.; Li, C.; Lingelbach, M.; Martín-Martín, R.; Xia, F.; Vainio, K.E.; Lian, Z.; Gokmen, C.; Buch, S.; Liu, K.; et al. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In Proceedings of the Conference on robot learning. PMLR, 2022, pp. 477–490.
22. Padmakumar, A.; Thomason, J.; Shrivastava, A.; Lange, P.; Narayan-Chen, A.; Gella, S.; Piramuthu, R.; Tur, G.; Hakkani-Tur, D. TEACh: Task-driven Embodied Agents that Chat. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2022, Vol. 36, pp. 2017–2025.
23. Savva, M.; Kadian, A.; Maksymets, O.; Zhao, Y.; Wijmans, E.; Jain, B.; Straub, J.; Liu, J.; Koltun, V.; Malik, J.; et al. Habitat: A platform for embodied ai research. In Proceedings of the Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 9339–9347.
24. He, C.; Zhou, X.; Wu, Y.; Yu, X.; Zhang, Y.; Zhang, L.; Wang, D.; Lyu, S.; Xu, H.; Xiaoqiao, W.; et al. Esgenius: Benchmarking llms on environmental, social, and governance (esg) and sustainability knowledge. In Proceedings of the Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, 2025, pp. 14612–14653. https://doi.org/10.18653/v1/2025.emnlp-main.739.
25. Nissenbaum, H. Privacy as contextual integrity. *Wash. L. Rev.* **2004**, *79*, 119.
26. Christiano, P.F.; Leike, J.; Brown, T.; Martic, M.; Legg, S.; Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* **2017**, *30*.
27. Hadfield-Menell, D.; Dragan, A.D.; Abbeel, P.; Russell, S. The off-switch game. In Proceedings of the AAAI Workshops, 2017.
28. He, C.; Zhou, X.; Yu, X.; Zhang, L.; Zhang, Y.; Wu, Y.; Xiao, L.; Li, L.; Wang, D.; Xu, H.; et al. SSKG Hub: An Expert-Guided Platform for LLM-Empowered Sustainability Standards Knowledge Graphs. *arXiv preprint arXiv:2603.00669* **2026**. https://doi.org/10.48550/arXiv.2603.00669.
29. Parisi, G.I.; Kemker, R.; Part, J.L.; Kanan, C.; Wermter, S. Continual lifelong learning with neural networks: A review. *Neural networks* **2019**, *113*, 54–71.
30. De Lange, M.; Aljundi, R.; Masana, M.; Parisot, S.; Jia, X.; Leonardis, A.; Slabaugh, G.; Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence* **2021**, *44*, 3366–3385.
31. Van de Ven, G.M.; Tolias, A.S. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734* **2019**.
32. Lei, M.; Cai, H.; Cui, Z.; Tan, L.; Hong, J.; Hu, G.; Zhu, S.; Wu, Y.; Jiang, S.; Wang, G.; et al. RoboMemory: A Brain-Inspired Multi-Memory Agentic Framework for Lifelong Learning in Physical Embodied Systems. *arXiv preprint arXiv:2508.01415* **2025**. Also presented at the NeurIPS 2025 Workshop on Space in Vision, Language, and Embodied AI, https://doi.org/10.48550/arXiv.2508.01415.
33. Packer, C.; Wooders, S.; Lin, K.; Fang, V.; Patil, S.G.; Stoica, I.; Gonzalez, J.E. MemGPT: Towards LLMs as Operating Systems. *arXiv preprint arXiv:2310.08560* **2023**. https://doi.org/10.48550/arXiv.2310.08560.
34. Hafner, D.; Lillicrap, T.; Ba, J.; Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603* **2019**.
35. Hafner, D.; Pasukonis, J.; Ba, J.; Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104* **2023**.

 doi:10.20944/preprints202604.2037.v1

36. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems* **2017**, *30*.
37. Angelopoulos, A.N.; Bates, S. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning* **2023**, *16*, 494–591.
38. Kaelbling, L.P.; Littman, M.L.; Cassandra, A.R. Planning and acting in partially observable stochastic domains. *Artificial intelligence* **1998**, *101*, 99–134.
39. Kaelbling, L.P.; Lozano-Pérez, T. Hierarchical task and motion planning in the now. In Proceedings of the 2011 IEEE international conference on robotics and automation. IEEE, 2011, pp. 1470–1477.
40. Garrett, C.R.; Chitnis, R.; Holladay, R.; Kim, B.; Silver, T.; Kaelbling, L.P.; Lozano-Pérez, T. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems* **2021**, *4*, 265–293.
41. Chen, L.; Lu, K.; Rajeswaran, A.; Lee, K.; Grover, A.; Laskin, M.; Abbeel, P.; Srinivas, A.; Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems* **2021**, *34*, 15084–15097.
42. Janner, M.; Li, Q.; Levine, S. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems* **2021**, *34*, 1273–1286.
43. Mandlekar, A.; Xu, D.; Wong, J.; Nasiriany, S.; Wang, C.; Kulkarni, R.; Fei-Fei, L.; Savarese, S.; Zhu, Y.; Martín-Martín, R. What Matters in Learning from Offline Human Demonstrations for Robot Manipulation. In Proceedings of the Proceedings of the 5th Conference on Robot Learning. PMLR, 2022, Vol. 164, *Proceedings of Machine Learning Research*, pp. 1678–1690.
44. Mandlekar, A.; Nasiriany, S.; Wen, B.; Akinola, I.; Narang, Y.; Fan, L.; Zhu, Y.; Fox, D. MimicGen: A Data Generation System for Scalable Robot Learning using Human Demonstrations. In Proceedings of the Proceedings of The 7th Conference on Robot Learning. PMLR, 2023, Vol. 229, *Proceedings of Machine Learning Research*, pp. 1820–1864.
45. Chi, C.; Xu, Z.; Feng, S.; Cousineau, E.; Du, Y.; Burchfiel, B.; Tedrake, R.; Song, S. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *The International Journal of Robotics Research* **2025**, *44*, 1684–1704. https://doi.org/10.1177/02783649241273668.
46. Mendonca, R.; Panov, E.; Bucher, B.; Wang, J.; Pathak, D. Continuously Improving Mobile Manipulation with Autonomous Real-World RL. *arXiv preprint arXiv:2409.20568* **2024**. https://doi.org/10.48550/arXiv.2409.20568.
47. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. Harness Engineering for Language Agents: The Harness Layer as Control, Agency, and Runtime. *Preprints* **2026**. https://doi.org/10.20944/preprints202603.1756.v1.
48. Finn, C.; Levine, S.; Abbeel, P. Guided cost learning: Deep inverse optimal control via policy optimization. In Proceedings of the International conference on machine learning. PMLR, 2016, pp. 49–58.
49. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. SP$\hat{2}$DPO: An LLM-assisted Semantic Per-Pair DPO Generalization. *arXiv preprint arXiv:2601.22385* **2026**.
50. Park, J.S.; O'Brien, J.; Cai, C.J.; Morris, M.R.; Liang, P.; Bernstein, M.S. Generative agents: Interactive simulacra of human behavior. In Proceedings of the Proceedings of the 36th annual acm symposium on user interface software and technology, 2023, pp. 1–22.
51. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. In Proceedings of the International Conference on Learning Representations (ICLR), 2023.
52. Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; Yao, S. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems* **2023**, *36*, 8634–8652.
53. Nikolaidis, S.; Zhu, Y.X.; Hsu, D.; Srinivasa, S. Human-robot mutual adaptation in shared autonomy. In Proceedings of the Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, 2017, pp. 294–302.
54. He, C.; Liu, Y.; Guo, Q.; Miao, C. Multi-scale quasi-RNN for next item recommendation. *arXiv preprint arXiv:1902.09849* **2019**.
55. Yan, Z.; Sun, L.; Krajnik, T.; Duckett, T.; Bellotto, N. Towards long-term autonomy: A perspective from robot learning. *arXiv preprint arXiv:2212.12798* **2022**.
56. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. Recommender Systems Should Now Be Designed Towards Agents. *Preprints* **2026**.
57. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. The Synthetic Media Exchange: When Lineage Becomes Currency. *Preprints* **2026**.

 doi:10.20944/preprints202604.2037.v1

58. Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P.; Schulman, J.; Mané, D. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565* **2016**.
59. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. ESGlass: Glass-Box ESG and Sustainability Reports. *Preprints* **2026**. https://doi.org/10.20944/preprints202603.2187.v1.
60. Kim, T.; Min, C.; Kim, B.; Kim, J.; Jeung, W.; Choi, J. Realfred: An embodied instruction following benchmark in photo-realistic environments. In Proceedings of the European Conference on Computer Vision. Springer, 2024, pp. 346–364.
61. Gao, X.; Gao, Q.; Gong, R.; Lin, K.; Thattai, G.; Sukhatme, G.S. Dialfred: Dialogue-enabled agents for embodied instruction following. *IEEE Robotics and Automation Letters* **2022**, *7*, 10049–10056.
62. Anderson, P.; Wu, Q.; Teney, D.; Bruce, J.; Johnson, M.; Sünderhauf, N.; Reid, I.; Gould, S.; Van Den Hengel, A. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3674–3683.
63. Thomason, J.; Murray, M.; Cakmak, M.; Zettlemoyer, L. Vision-and-dialog navigation. In Proceedings of the Conference on Robot Learning. PMLR, 2020, pp. 394–406.
64. Szot, A.; Clegg, A.; Undersander, E.; Wijmans, E.; Zhao, Y.; Turner, J.; Maestre, N.; Mukadam, M.; Chaplot, D.S.; Maksymets, O.; et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems* **2021**, *34*, 251–266.
65. Zhang, L.; Zhou, X.; He, C.; Wang, D.; Wu, Y.; Xu, H.; Liu, W.; Miao, C. Mmesgbench: Pioneering multimodal understanding and complex reasoning benchmark for esg tasks. In Proceedings of the Proceedings of the 33rd ACM International Conference on Multimedia, 2025, pp. 12829–12836.
66. Kolve, E.; Mottaghi, R.; Han, W.; VanderBilt, E.; Weihs, L.; Herrasti, A.; Deitke, M.; Ehsani, K.; Gordon, D.; Zhu, Y.; et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474* **2017**.
67. Puig, X.; Ra, K.; Boben, M.; Li, J.; Wang, T.; Fidler, S.; Torralba, A. Virtualhome: Simulating household activities via programs. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8494–8502.
68. Gupta, S.; Davidson, J.; Levine, S.; Sukthankar, R.; Malik, J. Cognitive mapping and planning for visual navigation. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2616–2625.
69. Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Proceedings of the Conference on robot learning. PMLR, 2020, pp. 1094–1100.
70. He, C.; Zhou, X.; Wang, D.; Yu, X.; Xiao, L.; Li, L.; Xu, H.; Liu, W.; Miao, C. KG4ESG: The ESG Knowledge Graph Atlas. *Preprints* **2026**. https://doi.org/10.20944/preprints202602.1970.v2.
71. Driess, D.; Xia, F.; Sajjadi, M.S.; Lynch, C.; Chowdhery, A.; Ichter, B.; Wahid, A.; Tompson, J.; Vuong, Q.; Yu, T.; et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378* **2023**.
72. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. The AutoResearch Moment: From Experimenter to Research Director. *Preprints* **2026**. https://doi.org/10.20944/preprints202603.1329.v1.
73. Reed, S.; Zolna, K.; Parisotto, E.; Colmenarejo, S.G.; Novikov, A.; Barth-Maron, G.; Gimenez, M.; Sulsky, Y.; Kay, J.; Springenberg, J.T.; et al. A generalist agent. *arXiv preprint arXiv:2205.06175* **2022**.
74. Bousmalis, K.; Vezzani, G.; Rao, D.; Devin, C.; Lee, A.X.; Bauzá, M.; Davchev, T.; Zhou, Y.; Gupta, A.; Raju, A.; et al. Robocat: A self-improving generalist agent for robotic manipulation. *arXiv preprint arXiv:2306.11706* **2023**.
75. Belkhale, S.; Ding, T.; Xiao, T.; Sermanet, P.; Vuong, Q.; Tompson, J.; Chebotar, Y.; Dwibedi, D.; Sadigh, D. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823* **2024**.
76. Walke, H.R.; Black, K.; Zhao, T.Z.; Vuong, Q.; Zheng, C.; Hansen-Estruch, P.; He, A.W.; Myers, V.; Kim, M.J.; Du, M.; et al. Bridgedata v2: A dataset for robot learning at scale. In Proceedings of the Conference on Robot Learning. PMLR, 2023, pp. 1723–1736.
77. Kim, M.J.; Finn, C.; Liang, P. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645* **2025**.
78. Fan, Y.; Ding, P.; Bai, S.; Tong, X.; Zhu, Y.; Lu, H.; Dai, F.; Zhao, W.; Liu, Y.; Huang, S.; et al. Long-vla: Unleashing long-horizon capability of vision language action model for robot manipulation. *arXiv preprint arXiv:2508.19958* **2025**.

 doi:10.20944/preprints202604.2037.v1

79. Liu, Y.; Zhu, J.; Mo, Y.; Li, G.; Cao, X.; Jin, J.; Shen, Y.; Li, Z.; Yu, T.; Yuan, W.; et al. PALM: Progress-Aware Policy Learning via Affordance Reasoning for Long-Horizon Robotic Manipulation. *arXiv preprint arXiv:2601.07060* **2026**. https://doi.org/10.48550/arXiv.2601.07060.
80. Jiang, Y.; Gupta, A.; Zhang, Z.; Wang, G.; Dou, Y.; Chen, Y.; Fei-Fei, L.; Anandkumar, A.; Zhu, Y.; Fan, L. VIMA: Robot Manipulation with Multimodal Prompts. In Proceedings of the Proceedings of the 40th International Conference on Machine Learning. PMLR, 2023, Vol. 202, *Proceedings of Machine Learning Research*, pp. 14975–15022.
81. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. From Prompts to Portfolios: AI Agents as Agentic Multimedia Firms. *Preprints* **2026**.
82. Zhao, T.Z.; Kumar, V.; Levine, S.; Finn, C. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705* **2023**.
83. Fu, Z.; Zhao, T.Z.; Finn, C. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117* **2024**.
84. Zhao, T.Z.; Tompson, J.; Driess, D.; Florence, P.; Ghasemipour, K.; Finn, C.; Wahid, A. Aloha unleashed: A simple recipe for robot dexterity. *arXiv preprint arXiv:2410.13126* **2024**.
85. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. Let Papers Flow: AI Conferences Should Embrace Submission Explosion via Autonomous Review Pipelines. *Preprints* **2026**.
86. Devin, C.; Gupta, A.; Darrell, T.; Abbeel, P.; Levine, S. Learning modular neural network policies for multi-task and multi-robot transfer. In Proceedings of the 2017 IEEE international conference on robotics and automation (ICRA). IEEE, 2017, pp. 2169–2176.
87. Kalashnikov, D.; Irpan, A.; Pastor, P.; Ibarz, J.; Herzog, A.; Jang, E.; Quillen, D.; Holly, E.; Kalakrishnan, M.; Vanhoucke, V.; et al. QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation. *arXiv preprint arXiv:1806.10293* **2018**. https://doi.org/10.48550/arXiv.1806.10293.
88. Zeng, A.; Song, S.; Welker, S.; Lee, J.; Rodriguez, A.; Funkhouser, T. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 4238–4245.
89. OpenAI.; Andrychowicz, M.; Baker, B.; Chociej, M.; Jozefowicz, R.; McGrew, B.; Pachocki, J.; Petron, A.; Plappert, M.; Powell, G.; et al. Learning Dexterous In-Hand Manipulation. *The International Journal of Robotics Research* **2020**, *39*, 3–20. https://doi.org/10.1177/0278364919887447.
90. Pathak, D.; Mahmoudieh, P.; Luo, G.; Agrawal, P.; Chen, D.; Shentu, Y.; Shelhamer, E.; Malik, J.; Efros, A.A.; Darrell, T. Zero-shot visual imitation. In Proceedings of the Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2018, pp. 2050–2053.
91. Ebert, F.; Finn, C.; Dasari, S.; Xie, A.; Lee, A.; Levine, S. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568* **2018**.
92. Levine, S.; Finn, C.; Darrell, T.; Abbeel, P. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research* **2016**, *17*, 1–40.
93. Shridhar, M.; Manuelli, L.; Fox, D. Cliport: What and where pathways for robotic manipulation. In Proceedings of the Conference on robot learning. PMLR, 2022, pp. 894–906.
94. Shridhar, M.; Manuelli, L.; Fox, D. Perceiver-Actor: A Multi-Task Transformer for Robotic Manipulation. In Proceedings of the Proceedings of the 6th Conference on Robot Learning (CoRL), 2022.
95. Lynch, C.; Sermanet, P. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648* **2020**.
96. Nair, S.; Rajeswaran, A.; Kumar, V.; Finn, C.; Gupta, A. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601* **2022**.
97. Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In Proceedings of the 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2017, pp. 23–30.
98. Peng, X.B.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In Proceedings of the 2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018, pp. 3803–3810.
99. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International conference on machine learning. PMLR, 2017, pp. 1126–1135.
100. Kumar, A.; Fu, Z.; Pathak, D.; Malik, J. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034* **2021**.

 doi:10.20944/preprints202604.2037.v1

101. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. The One-Person Laboratory Should Be a First-Class Unit of Evaluation in Dry-Lab AI Research. *Preprints* **2026**.
102. Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* **2017**, *114*, 3521–3526.
103. Li, Z.; Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence* **2017**, *40*, 2935–2947.
104. Rebuffi, S.A.; Kolesnikov, A.; Sperl, G.; Lampert, C.H. icarl: Incremental classifier and representation learning. In Proceedings of the Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 2001–2010.
105. Lopez-Paz, D.; Ranzato, M. Gradient episodic memory for continual learning. *Advances in neural information processing systems* **2017**, *30*.
106. Chaudhry, A.; Ranzato, M.; Rohrbach, M.; Elhoseiny, M. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420* **2018**.
107. Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; Wayne, G. Experience replay for continual learning. *Advances in neural information processing systems* **2019**, *32*.
108. Schwarz, J.; Czarnecki, W.; Luketina, J.; Grabska-Barwinska, A.; Teh, Y.W.; Pascanu, R.; Hadsell, R. Progress & compress: A scalable framework for continual learning. In Proceedings of the International conference on machine learning. PMLR, 2018, pp. 4528–4537.
109. Buzzega, P.; Boschini, M.; Porrello, A.; Abati, D.; Calderara, S. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems* **2020**, *33*, 15920–15930.
110. Khetarpal, K.; Riemer, M.; Rish, I.; Precup, D. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research* **2022**, *75*, 1401–1476.
111. Rusu, A.A.; Rabinowitz, N.C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671* **2016**.
112. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, 1998.
113. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *nature* **2015**, *518*, 529–533.
114. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* **2017**.
115. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International conference on machine learning. Pmlr, 2018, pp. 1861–1870.
116. Achiam, J.; Held, D.; Tamar, A.; Abbeel, P. Constrained policy optimization. In Proceedings of the International conference on machine learning. Pmlr, 2017, pp. 22–31.
117. LaValle, S.M. *Planning algorithms*; Cambridge university press, 2006.
118. Levine, S.; Kumar, A.; Tucker, G.; Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* **2020**.
119. Kumar, A.; Zhou, A.; Tucker, G.; Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems* **2020**, *33*, 1179–1191.
120. Kostrikov, I.; Nair, A.; Levine, S. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169* **2021**.
121. Kidambi, R.; Rajeswaran, A.; Netrapalli, P.; Joachims, T. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems* **2020**, *33*, 21810–21823.
122. Ha, D.; Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122* **2018**, *2*, 440.
123. Hafner, D.; Lillicrap, T.; Norouzi, M.; Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193* **2020**.
124. Schrittwieser, J.; Antonoglou, I.; Hubert, T.; Simonyan, K.; Sifre, L.; Schmitt, S.; Guez, A.; Lockhart, E.; Hassabis, D.; Graepel, T.; et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature* **2020**, *588*, 604–609.
125. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. PCA-OS: A Planetary Climate Adaptation Operating System. *Preprints* **2026**. Preprint.

 doi:10.20944/preprints202604.2037.v1

126. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *nature* **2016**, *529*, 484–489.
127. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815* **2017**.
128. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in neural information processing systems* **2017**, *30*.
129. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* **2020**.
130. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the International conference on machine learning. PmLR, 2021, pp. 8748–8763.
131. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Advances in neural information processing systems* **2020**, *33*, 1877–1901.
132. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. Palm: Scaling language modeling with pathways. *Journal of machine learning research* **2023**, *24*, 1–113.
133. Alayrac, J.B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems* **2022**, *35*, 23716–23736.
134. Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F.L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* **2023**.
135. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. Beyond Human Measure: ASI Should Be Guided by Open-World Alignment. *Preprints* **2026**. https://doi.org/10.20944/preprints202604.1749.v1.
136. Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; Scialom, T. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems* **2023**, *36*, 68539–68551.
137. Wang, G.; Xie, Y.; Jiang, Y.; Mandlekar, A.; Xiao, C.; Zhu, Y.; Fan, L.; Anandkumar, A. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291* **2023**.
138. He, C.; Zhou, X.; Wang, D.; Xu, H.; Liu, W.; Miao, C. OpenClaw as Language Infrastructure: A Case-Centered Survey of a Public Agent Ecosystem in the Wild. *Preprints* **2026**. https://doi.org/10.20944/preprints202603.1060.v1.
139. Gal, Y.; Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the international conference on machine learning. PMLR, 2016, pp. 1050–1059.
140. Geifman, Y.; El-Yaniv, R. Selective classification for deep neural networks. *Advances in neural information processing systems* **2017**, *30*.