WAS IT ME OR THE WORLD? SCOPE-AWARE ADAPTATION FOR EMBODIED AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An embodied agent that relearns the same limitation on every task spends its interaction budget on avoidable failures. Yet an isolated failure does not reveal whether a capability is unavailable everywhere, restricted locally, or temporarily disrupted. We introduce WIMW (*Was It Me or the World?*), a training-free method that uses diagnostic actions to determine the scope of future restrictions. Reference tests and target verification produce capability contracts that an execution guard enforces across tasks. Patch-VOI selects tests by their expected reduction in scope errors after verification, charging for the actions needed to obtain evidence. With a frozen Qwen3-VL-32B planner on EB-ALFRED and EB-Habitat, the reference controller reduces unexplained failed picks by 95.0% and 96.0% relative to No update under injected skill-wide faults across 300-task streams. Under these faults, matched comparisons against passive global inference show additional failure reductions of 12.3% and 13.3% at unchanged mean completion; actions rise by 2.6% in ALFRED and fall by 13.8% in Habitat. Numerical Patch-VOI lowers mean scope loss plus diagnostic cost by 9.1% and 7.7% relative to fixed-order acquisition, with slightly lower mean completion. Together, these results connect reusable failure memory to the decision of whether further evidence is worth its interaction cost. Code and data: anonymous repository.

1 INTRODUCTION

A failed action still consumes a robot's time and interaction budget. Reaching a target, preparing a manipulation, and retrying all incur costs, even when the attempt makes no progress. Across a sequence of tasks, a persistent limitation can make these costs recur. An agent that learns from experience should be able to avoid repeatedly spending its budget on the same unavailable capability.

Embodied agents reuse experience through written reflections (Shinn et al., 2023), retrieved memories (Chhikara et al., 2025; Xu et al., 2025), and revised skill libraries (Ju et al., 2026; Lu et al., 2026). Physical-memory systems also turn action outcomes into hypotheses and reusable principles (Li et al., 2026). For failure memory, a central challenge is deciding when an earlier outcome provides enough evidence to stop trying. The failed trajectory alone may leave this decision unresolved.

A single failure may reflect an impaired capability, a constraint specific to the current context, or a temporary disruption. These explanations call for different responses. An overly narrow update leaves the agent to encounter a shared limitation again, while an overly broad update can suppress actions that would succeed. Repetition alone need not resolve the ambiguity: retrying the original target may reproduce the failure without revealing whether it extends elsewhere. We ask how an agent can reduce repeated failed attempts while preserving actions that could still succeed.

Our approach uses diagnostic actions to decide which future attempts to avoid. Testing the same skill in other contexts informs the *scope* of a limitation. Revisiting the affected context checks for recurrence before the agent stores a restriction. Diagnosis also consumes actions. Patch-VOI formalizes this tradeoff by weighing the expected improvement in the restriction decision against the interaction and disturbance costs of the test.

We implement this idea in WIMW (*Was It Me or the World?*). After an unexplained failure, it retries the target, tests reference targets if the retry fails, and rechecks the affected target type before committing a restriction. The resulting *capability contract* specifies where a skill should be withheld.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

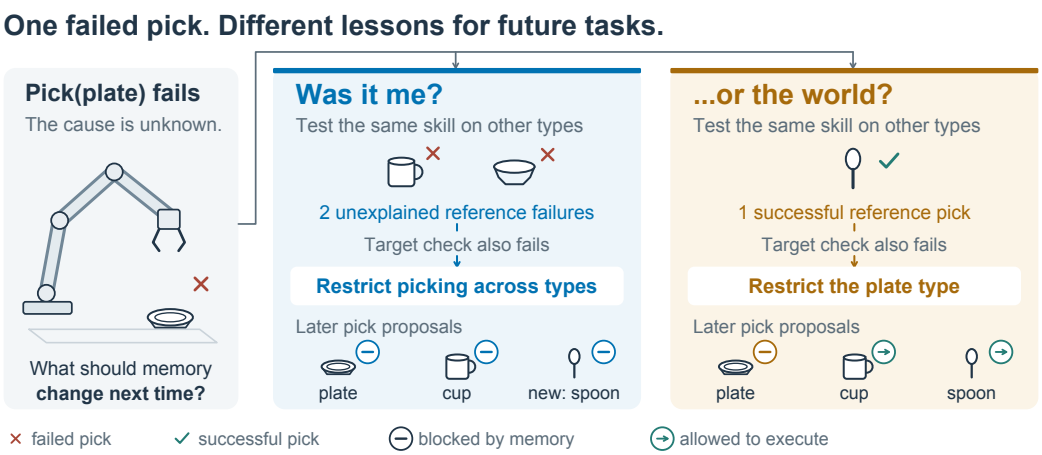


Figure 1: **Infer scope before carrying a failure forward.** The reference controller proposes skill-wide scope after two unexplained reference failures, or target-type scope after a reference success. A further unexplained target failure commits the contract. Skill-wide contracts also cover untested types, while type-specific contracts leave other types available. Circled symbols indicate permission, not outcomes; periodic body rechecks are omitted.

An execution guard applies the contract to later action proposals, allowing evidence from one task to change behavior in subsequent tasks (Figure 1). The visual implementation is training-free: it updates an execution ledger and capability contracts while keeping the planner's parameters fixed.

Our contributions are as follows:

- We formulate failure memory as selecting the scope of an action restriction. Patch-VOI values tests by the scope errors they can prevent, exposing when uncertainty reduction has no decision value. Its numerical selector includes acquisition and final verification costs.
- We develop WIMW, which turns cross-context evidence into verified capability contracts. By enforcing these contracts across tasks, a frozen planner can avoid rediscovering a skill-wide limitation on each new object type.
- We establish the benefit of contract-based adaptation and measure when active evidence pays off. WIMW reduces failed picks by 95.0% and 96.0% relative to No update under injected skill-wide faults in two visual environments. Matched passive-global controls separate reference acquisition from shared recovery and enforcement, while numerical Patch-VOI lowers mean scope loss plus diagnostic cost by 9.1% and 7.7% relative to fixed-order acquisition.

2 FAILURE SCOPE IN TASK STREAMS

Consider tasks $g_1, \ldots, g_T$ executed sequentially by a frozen vision-language planner π_θ with persistent memory M. From an image, the instruction, and execution feedback, the planner proposes a skill invocation $a_t = (k_t, x_t)$ with skill k_t and target x_t. The environment returns the next observation and an outcome y_t.

A failed invocation can depend on the robot's capability b, the target state w_x, and temporary execution conditions ϵ_t:

$$y_t \sim P(y \mid k_t, x_t, b, w_{x_t}, \epsilon_t). \tag{1}$$

A body fault limits a skill across object types; a world fault affects a particular type. A transient fault resolves without a persistent restriction. These cases require different restrictions on future skill–target pairs.

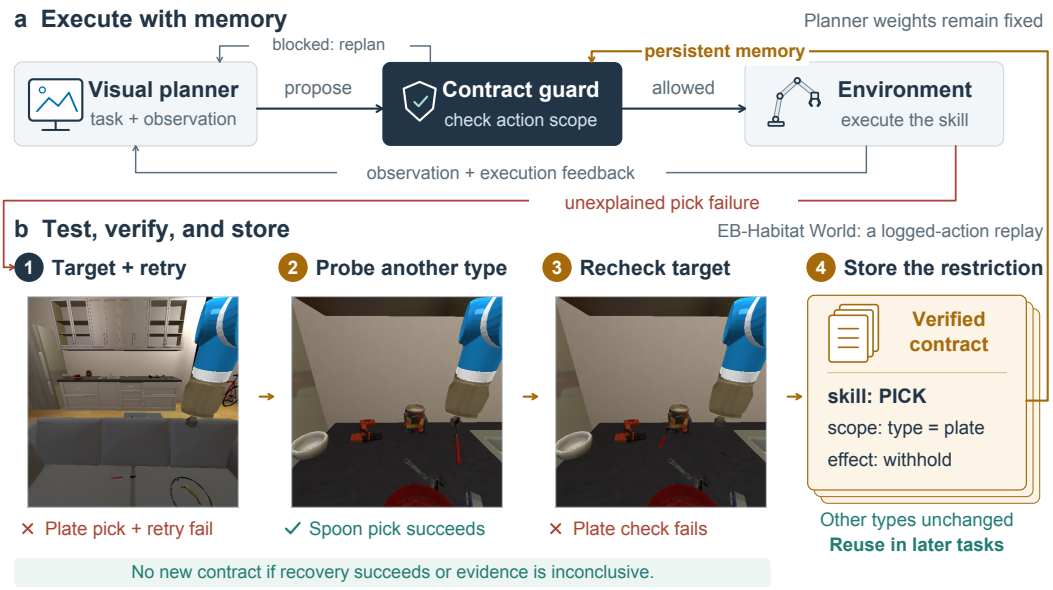


Figure 2: **From evidence to reusable memory.** (a) Stored contracts guard the frozen planner's actions. (b) Replayed EB-Habitat World actions: the plate pick and retry fail, the spoon pick succeeds, and an unexplained plate recheck failure commits a plate-type contract. The task remains incomplete. Navigation and releases are counted but omitted visually. Faults are injected at the skill interface.

Let $\mathcal{R}^*$ denote the pairs disabled by a persistent fault and $\widehat{\mathcal{R}}(M)$ those withheld by memory. Then $\widehat{\mathcal{R}}(M) \setminus \mathcal{R}^*$ removes useful actions, while $\mathcal{R}^* \setminus \widehat{\mathcal{R}}(M)$ leaves disabled actions available. The cost of either error depends on which targets recur.

3 The WIMW Method

WIMW turns diagnostic evidence into verified capability contracts that govern later actions. We define its decision objective and the numerical selector evaluated in Table 4. Figure 2 and Appendix B describe the reference controller with fixed diagnostic rules.

3.1 From failure hypotheses to capability contracts

A capability contract $m = (k, s, v)$ specifies a skill k, a scope predicate $s(x)$, and a restriction v that withholds the skill when $s(x) = 1$. The two persistent scopes are

$$s_{\text{body}}(x) = 1, \qquad s_{\text{world},c}(x) = \mathbb{I}[\text{type}(x) = c]. \tag{2}$$

A body contract covers all object types, including unseen ones. A world contract covers type c. We evaluate picking, with local scope defined by type rather than instance.

Let D be the collected evidence and $h \in \mathcal{H}$ a failure hypothesis with persistent disabled set $\mathcal{R}_h$; for a temporary failure, $\mathcal{R}_h = \varnothing$. A candidate memory update, or *patch* $q \in \mathcal{Q}$, adds a body contract, adds a world contract, or leaves memory unchanged. Its resulting restriction set is $\widehat{\mathcal{R}}_q$. Two causes that require the same patch need not be distinguished.

3.2 The cost of choosing the wrong scope

For a skill–target pair $u = (k, x)$, let $w(u) \geq 0$ denote expected demand in a fixed reference workload. We define the scope loss as

$$L(q, h; w) = \sum_u w(u) \Big(c_{\text{over}} \mathbb{I}[u \in \widehat{\mathcal{R}}_q \setminus \mathcal{R}_h] + c_{\text{under}} \mathbb{I}[u \in \mathcal{R}_h \setminus \widehat{\mathcal{R}}_q] \Big). \tag{3}$$

The nonnegative coefficients penalize restricting an available skill and leaving a persistent failure unaddressed. Demand is fixed across patches so a restriction cannot reduce its own penalty by suppressing requests. Scope loss differs from task completion, which also depends on navigation, alternative plans, and recovery.

Given a belief $b_D(h) = P(h \mid D)$, the Bayes risk and the associated patch are

$$R(D) = \min_{q \in \mathcal{Q}} \sum_h b_D(h) L(q, h; w), \qquad q^*(D) \in \arg\min_{q \in \mathcal{Q}} \sum_h b_D(h) L(q, h; w). \tag{4}$$

3.3 Patch-VOI: evidence for a decision

A diagnostic action a produces outcome y and evidence $D_{a,y} = D \cup \{(a, y)\}$. For a fixed hypothesis family and reference workload, *Patch-VOI* applies decision-theoretic value of information (Howard, 1966) to scope loss:

$$\mathrm{PatchVOI}(a \mid D) = R(D) - \mathbb{E}_{y|a,D}[R(D_{a,y})] - \lambda C(a \mid D) - \mu K(a \mid D). \tag{5}$$

Here C is the expected interaction cost of navigation, hand clearing, and manipulation, and K is the expected cost of disturbing the task state. Weights $\lambda, \mu \geq 0$ scale these costs against patch loss. The one-step policy selects the feasible probe with the highest positive Patch-VOI and otherwise stops. It need not be optimal over a sequence of probes (Appendix A).

Evidence has value when it can change the patch decision. If one patch is optimal after every outcome, the expected Bayes-risk reduction is zero even when information gain is positive (Appendix A).

3.4 Numerical Patch-VOI with verified commitment

The selector maintains a belief over skill-wide failure, failure specific to the triggering type, and no persistent failure. A tabulated model supplies likelihoods in Equation 8 from the test role, target type, navigation and hand state, and attempt history. A separate calibration split supplies outcome frequencies and action costs. Demand weights use requests recorded before applying memory. These estimates remain fixed during diagnosis (Appendix A.1).

Verification can cancel a patch, so its outcome enters the decision loss. Let q_0 leave memory unchanged and $g(q, z)$ retain q only after a fresh unexplained target failure z, returning q_0 otherwise. With final-check action cost c_v, the risk of stopping diagnosis is

$$R_v(D) = \min_{q \in \mathcal{Q}_D} \sum_{h,z} b_D(h) \widehat{P}(z \mid q, h, D) \big[L(g(q, z), h; w) + c_v(q, z, D) \big]. \tag{6}$$

For q_0, verification costs zero. The feasible set $\mathcal{Q}_D$ contains only q_0 after target recovery, episode termination, or insufficient verification budget; otherwise it also contains type-specific and skill-wide proposals.

For each feasible test, the selector enumerates branches $\omega = (y, \ell, s')$ with feedback, action count, and public successor state:

$$\widehat{V}_v(a \mid D) = R_v(D) - \sum_\omega \widehat{P}(\omega \mid a, D) R_v(D_{a,\omega}) - \lambda \widehat{C}(a \mid D). \tag{7}$$

We set $\mu = 0$ and include navigation, hand clearing, and releases in C. Task completion captures the remaining effects of state changes. Successor risks use the remaining budget and observed state. The highest positive score selects the next test. Otherwise acquisition stops, and the minimizing proposal in Equation 6 determines whether to run the final check.

3.5 Contract verification and execution

Both controllers revisit the target type before storing a proposed contract m. A fresh unexplained failure commits $M^+ = M \cup \{m\}$; otherwise $M^+ = M$. Success cancels the proposal and returns the pick outcome to the planner. An explained or unavailable check defers commitment. A disruption that survives verification can still produce a persistent restriction, as represented by g in Equation 6.

Both controllers use a guard that blocks picks matching a stored contract and returns feedback for replanning. Two blocks terminate the episode without certifying infeasibility. Every fourth proposal matching a body contract is executed as a recheck, and success clears that contract. Local contracts persist.

Diagnostic adapters use simulator metadata for reference types and hand state. All executed diagnostic actions enter the interaction budget and reported costs.

4 EXPERIMENTS

We evaluate failure reuse at three levels: the complete reference controller, reference acquisition with scope inference and execution rules held fixed, and numerical selection of diagnostic tests.

4.1 EXPERIMENTAL SETUP

We use EB-ALFRED and EB-Habitat from EmbodiedBench (Yang et al., 2025), with a frozen Qwen3-VL-32B-Instruct planner (Bai et al., 2025), official visual prompts, ten demonstrations, and temperature zero. Each episode has a 30-action budget. The environment and planner reset between tasks, while memory persists.

Four conditions vary failure scope and duration. Body disables every pick throughout the stream. World disables picks of one type: apple in ALFRED and plate in Habitat. Transient blocks the first pick of each episode, after which actions execute normally. Clean leaves the environment unchanged. All injected failures return the same generic feedback.

Unexplained failed picks U include diagnosis and rechecks. Completed tasks S and total actions A measure progress and interaction cost. Failures with explicit precondition explanations are excluded from U, but their actions count toward A. We compare No update, Mem0, EmbodiSkill, ExpeL (Zhao et al., 2024), AWM (Wang et al., 2025), PhysMem (Li et al., 2026), WorldMind (Ren et al., 2026), WIMW, and Oracle scope. All memory baselines update online from empty memory using the same visual planner. Oracle scope receives the true persistent restriction and shares WIMW's guard.

The main comparison uses all 300 tasks in official order, one stream per method and condition. Follow-ups use two distinct 60-task sets per environment, balanced across the six official subsets and including at least six World-target tasks. The first compares complete-manual EmbodiSkill; the second compares caches and matched acquisition. Both are drawn from the 300-task pool using fixed seeds and metadata, independently of method outcomes. Selection, exclusions, and adapter configurations appear in Appendix B.

The complete-manual and cache comparisons use two fixed within-subset orders and report mean $\pm$ sample SD across orders of the same task set. Matched acquisition adds five executor seeds per order, yielding ten paired streams per arm and condition. Its SDs describe between-stream variation. Shared controller rules appear in Appendix B.4.

4.2 REDUCING FAILED ATTEMPTS UNDER PERSISTENT FAULTS

Body tests whether an agent can reuse a skill-wide limitation across targets. World requires restricting the affected type while preserving the skill elsewhere (Table 1).

Under Body, WIMW reduces U by 95.0% in ALFRED and 96.0% in Habitat relative to No update. Global contracts block later proposals across types, leaving diagnosis and periodic rechecks as the remaining failed attempts. ExpeL, AWM, PhysMem, and WorldMind incur 2,205–2,486 failures in ALFRED and 2,247–2,767 in Habitat, where their text memories leave the planner to apply stored advice. With picking disabled, all methods complete zero ALFRED tasks and 9–10 Habitat tasks. WIMW avoids spending actions on the unavailable skill, reducing A by 45.6% and 88.0%, respectively.

Under World, WIMW incurs three unexplained failed picks per environment: the initial failure, target retry, and verification. A successful reference pick supports a type-specific contract. WIMW

Table 1: Persistent faults (32B; one 300-task stream in official order per cell). U: unexplained failed picks, including diagnosis and rechecks; Success: completed tasks; A: all actions. †: truncated memory; adapter configurations in Appendix B.3.

	Body			World		
Method	$U \downarrow$	Success ↑	$A \downarrow$	$U \downarrow$	Success ↑	$A \downarrow$
EB-ALFRED						
No update	2,472	0	6,500	177	121	5,306
Mem0	2,381	0	6,549	168	106	5,264
EmbodiSkill†	2,358	0	6,624	181	120	5,336
ExpeL	2,482	0	6,466	178	137	5,065
AWM	2,486	0	6,436	172	142	4,929
PhysMem	2,470	0	6,454	162	115	5,262
WorldMind	2,205	0	6,970	155	125	5,127
WIMW	123	0	3,539	3	139	4,748
Oracle scope	119	0	3,447	0	124	4,730
EB-Habitat						
No update	2,721	10	5,967	113	132	4,383
Mem0	2,777	9	5,865	108	125	4,520
EmbodiSkill†	2,776	10	6,059	142	134	4,636
ExpeL	2,247	10	5,421	108	151	4,457
AWM	2,747	10	6,037	124	135	4,479
PhysMem	2,716	10	6,109	139	133	4,704
WorldMind	2,767	10	6,084	114	142	4,343
WIMW	108	10	719	3	154	4,316
Oracle scope	104	10	696	0	137	4,242

completes 139 ALFRED tasks and 154 Habitat tasks, compared with 121 and 132 for No update. ExpeL also improves completion (137 and 151), but incurs 178 and 108 unexplained failures.

4.3 COMPARISON WITH SKILL MEMORY

Recovery is useful when failures are temporary. Under Habitat Transient, WIMW completes 179 of 300 tasks using 4,024 actions, the highest completion and lowest action count among the nine methods in this condition. WorldMind completes 156 tasks with 4,574 actions, and ExpeL completes 151 with 4,652. Successful target recovery resumes execution without a persistent restriction. Table 5 in Appendix C.1 reports Transient and Clean in both environments; the matched study below separates acquisition from this shared recovery routine.

The initial Mem0 and EmbodiSkill interfaces supply at most four 200-character lines, leaving EmbodiSkill at most two truncated rules and no execution notes. We therefore also compare against its complete retrieved manual, retaining the internal 40-line/4,000-character budget and online updates. Both methods start with empty memory on the first 60-task set.

Table 2: Complete-manual EmbodiSkill (ES) versus reference WIMW (W), without outer retrieval truncation (32B, 60 tasks per environment, two orders; mean ± sample SD). U: unexplained failed picks; S: completed tasks; A: all actions. Transient/Clean results are in Appendix C.2.

	$U \downarrow$		$S \uparrow$		$A \downarrow$	
Condition	ES	W	ES	W	ES	W
EB-ALFRED						
Body	521.5 ± 10.6	28.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	1391.0 ± 25.5	604.5 ± 10.6
World	40.0 ± 0.0	3.0 ± 0.0	20.0 ± 1.4	26.0 ± 1.4	1059.5 ± 6.4	947.5 ± 9.2
EB-Habitat						
Body	553.0 ± 11.3	25.0 ± 1.4	1.0 ± 0.0	1.0 ± 0.0	1182.5 ± 6.4	143.5 ± 19.1
World	39.0 ± 2.8	3.0 ± 0.0	23.5 ± 4.9	22.5 ± 3.5	975.0 ± 11.3	915.5 ± 17.7

The failure gap persists with complete manuals (Table 2). Under Body, WIMW reduces U by 94.6% in ALFRED and 95.5% in Habitat with unchanged completion; actions fall by 56.5% and 87.9%. EmbodiSkill still relies on the planner to follow its manual, whereas WIMW enforces restrictions before execution. Under World, WIMW also incurs fewer failures. Appendix C.2 gives the complementary Transient and Clean results.

4.4 WHEN DOES ACTIVE ACQUISITION PAY OFF?

Episodic and Persistent cache verify an unexplained failure before blocking its type through the same guard as WIMW. They differ in whether restrictions survive task boundaries; neither tests references or permits a skill-wide contract. Table 3(a) and Figure 3 compare them on the second 60-task set.

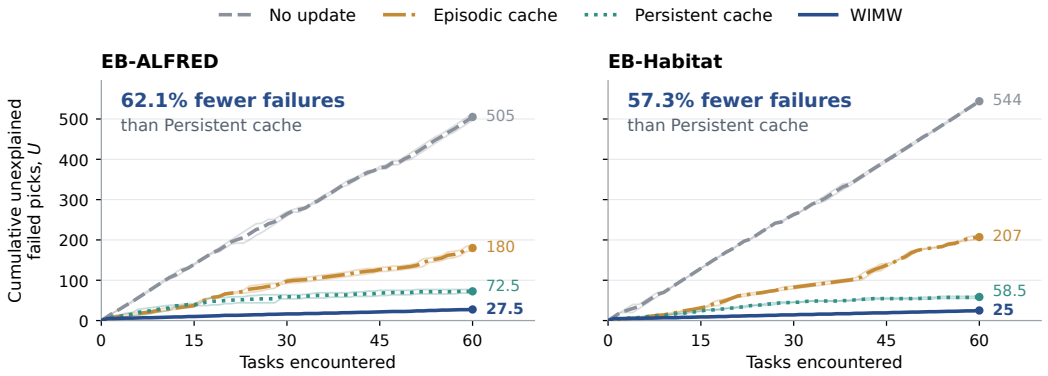


Figure 3: **A persistent type cache still rediscovers a skill-wide limitation.** Cumulative unexplained failed picks under Body (32B, 60 tasks), including diagnosis and rechecks. Thin traces show the two task orders; bold curves show their mean. Table 3(a) gives completion and action costs.

Table 3: Executable-memory controls (32B, 60 tasks per environment; mean ± sample SD). (a) Reference WIMW and caches, two orders. (b) Matched WIMW and Passive global + R (Passive + R), ten streams, with shared recovery, verification, guard, rechecks, and termination.

		EB-ALFRED			EB-Habitat		
Condition	Method	$U \downarrow$	$S/60 \uparrow$	$A \downarrow$	$U \downarrow$	$S/60 \uparrow$	$A \downarrow$
(a) Complete controller comparison							
Body	No update	505.0 ± 11.3	0.0 ± 0.0	1307.0 ± 0.0	544.0 ± 2.8	2.0 ± 0.0	1165.5 ± 17.7
	Episodic cache	180.0 ± 8.5	0.0 ± 0.0	772.5 ± 23.3	207.0 ± 7.1	2.0 ± 0.0	522.5 ± 16.3
	Persistent cache	72.5 ± 6.4	0.0 ± 0.0	467.5 ± 13.4	58.5 ± 4.9	2.0 ± 0.0	249.5 ± 12.0
	WIMW	27.5 ± 0.7	0.0 ± 0.0	725.0 ± 7.1	25.0 ± 0.0	2.0 ± 0.0	137.5 ± 4.9
World	Persistent cache	2.0 ± 0.0	24.0 ± 0.0	981.0 ± 52.3	2.0 ± 0.0	22.5 ± 0.7	824.5 ± 13.4
	WIMW	3.0 ± 0.0	22.0 ± 2.8	983.5 ± 62.9	3.0 ± 0.0	24.5 ± 0.7	848.5 ± 9.2
(b) Matched acquisition comparison							
Body	Passive + R	31.7 ± 1.6	0.0 ± 0.0	689.4 ± 31.8	29.3 ± 1.4	2.0 ± 0.0	164.7 ± 13.6
	WIMW	27.8 ± 1.0	0.0 ± 0.0	707.1 ± 35.2	25.4 ± 0.7	2.0 ± 0.0	141.9 ± 9.8
World	Passive + R	3.0 ± 0.0	23.4 ± 2.4	970.6 ± 42.7	3.0 ± 0.0	23.7 ± 1.8	837.4 ± 29.5
	WIMW	3.0 ± 0.0	23.2 ± 2.5	974.8 ± 41.6	3.0 ± 0.0	23.8 ± 2.0	841.1 ± 31.7
Transient	Passive + R	55.6 ± 0.8	27.2 ± 2.5	1094.3 ± 46.9	58.0 ± 0.0	34.1 ± 2.2	837.6 ± 28.1
	WIMW	55.5 ± 0.7	27.4 ± 2.4	1090.8 ± 45.1	58.0 ± 0.0	34.2 ± 2.1	835.9 ± 27.7
Clean	Passive + R	0.0 ± 0.0	26.4 ± 2.3	1091.7 ± 43.8	0.0 ± 0.0	26.7 ± 2.5	868.4 ± 35.6
	WIMW	0.0 ± 0.0	26.5 ± 2.4	1089.2 ± 41.7	0.0 ± 0.0	26.6 ± 2.4	870.1 ± 36.3

Under Body, retaining type restrictions reduces U from 180.0 to 72.5 in ALFRED and from 207.0 to 58.5 in Habitat. Reference WIMW further reduces these counts to 27.5 and 25.0 at unchanged

completion, but uses 55.1% more actions than Persistent cache in ALFRED and 44.9% fewer in Habitat. Under World, Persistent cache incurs two failed picks versus three for WIMW. These controllers also differ in retries and rechecks, motivating a comparison with those routines held fixed.

Passive global inference with shared execution rules. Table 3(b) compares *Passive global + R* with matched WIMW. Both infer local or global contracts from a common ledger of public outcomes. Passive global uses outcomes from ordinary execution, including failures on distinct types that can justify a global contract. The active arm additionally tests references when the ledger leaves scope unresolved. Both share target recovery R, fresh verification, memory format, guard, global rechecks, and termination. Thus passive acquisition still includes the common target checks and contract rechecks.

Skill-wide faults provide the clearest acquisition benefit. Under Body, reference acquisition lowers mean U from 31.7 to 27.8 in ALFRED and from 29.3 to 25.4 in Habitat, reductions of 12.3% and 13.3% at unchanged mean completion. Habitat also saves 22.8 actions (13.8%), whereas ALFRED spends 17.7 more (2.6%). Acquiring scope evidence therefore pays for itself in total interactions in Habitat Body; in ALFRED Body it trades extra actions for fewer failures.

Under World, both arms incur 3.0 failed picks per environment, while active acquisition adds 4.2 and 3.7 mean actions. Completion differs by -0.2 and $+0.1$ tasks. Under Transient, the corresponding completion differences are $+0.2$ and $+0.1$, with action savings of 3.5 and 1.7. With recovery shared, additional acquisition yields little observed completion gain in this condition. Clean produces no unexplained failures or eligible acquisition triggers; the small remaining differences occur without active acquisition.

4.5 DOES NUMERICAL PATCH-VOI IMPROVE DIAGNOSIS?

We compare numerical Patch-VOI with verification-only, fixed-order, and information-gain acquisition. All share the observation model, terminal patch decision, recovery, verification, and guard. Reference WIMW supplies an end-to-end comparison. Each policy runs eight 12-task blocks in two orders under all four conditions, resetting memory per block and order. Fixed-order and information-gain acquisition use development-selected caps of one and two tests, respectively, within a shared diagnostic action allowance.

Table 4: Numerical Patch-VOI on eight 12-task blocks per environment, each run in two orders. Means average equally over four conditions, blocks, and orders; memory resets for each stream. J combines realized scope loss and diagnostic action cost ($c_{\text{under}} = 1$, $c_{\text{over}} = 5$, $\lambda = 0.1$). Fixed-order and Information gain use caps of one and two tests.

	EB-ALFRED				EB-Habitat			
Policy	$U \downarrow$	$A \downarrow$	$S/12 \uparrow$	$J \downarrow$	$U \downarrow$	$A \downarrow$	$S/12 \uparrow$	$J \downarrow$
WIMW reference	5.86	194.72	4.20	1.94	5.59	135.84	4.70	1.77
Verification only	9.23	184.34	4.19	4.28	8.64	149.22	4.72	3.77
Fixed-order acquisition	5.55	193.08	4.23	1.86	5.41	136.45	4.75	1.68
Information gain	5.80	197.91	4.17	2.03	5.53	140.17	4.69	1.84
Numerical Patch-VOI	5.42	187.66	4.22	1.69	5.27	133.59	4.73	1.55

Numerical Patch-VOI achieves the lowest mean J and U in both environments (Table 4). Relative to fixed-order acquisition, mean scope loss plus diagnostic cost J falls from 1.86 to 1.69 in ALFRED and from 1.68 to 1.55 in Habitat, reductions of 9.1% and 7.7%. Actions fall by 2.8% and 2.1%, with slightly lower mean completion: 4.22 versus 4.23 ALFRED tasks per block and 4.73 versus 4.75 Habitat tasks.

Against verification-only, numerical selection lowers U and J in both environments. It uses more ALFRED actions (187.66 versus 184.34) and fewer Habitat actions (133.59 versus 149.22). Appendix A.2 defines task-entry scoring and aggregation across the four conditions.

5 Related work

Memory and skill revision. Agents reuse execution reflections (Shinn et al., 2023), extracted information (Chhikara et al., 2025), linked notes (Xu et al., 2025), and structured manuals (Chen et al., 2024). ExpeL combines retrieved experiences with extracted insights (Zhao et al., 2024). EmbodiSkill distinguishes skill defects from execution lapses (Ju et al., 2026); ASPIRE distills validated repairs into reusable skills (Lu et al., 2026). Capability contracts instead encode where future skill use should be withheld and enforce that decision during execution.

Failure diagnosis and interaction. ReAct revises plans through interaction (Yao et al., 2022); SayCan grounds skill selection in learned affordances (Ahn et al., 2022). REFLECT uses multisensory execution summaries to explain failures and guide corrective planning (Liu et al., 2023). PhysMem tests physical hypotheses before storing conditional principles (Li et al., 2026). WorldMind combines prediction errors and successful trajectories into reusable world knowledge (Ren et al., 2026). WIMW selects evidence for the scope of an action restriction and carries the resulting contract across tasks.

Environment-grounded memory. Concurrent work on environment-probing curation uses read-only tools after task completion to check, scope, and refresh candidate memories (Suresh et al., 2026). In our setting, diagnostic skill executions consume the current task's action budget and can change its state. The resulting contracts constrain later action proposals through an execution guard.

Embodied evaluation. ALFRED, Habitat, and ALFWorld support instruction-following and interactive learning (Shridhar et al., 2019; Savva et al., 2019; Shridhar et al., 2020). EmbodiedBench evaluates visual planners across simulators (Yang et al., 2025). RobustNav includes motor failures that disable rotation throughout an episode (Chattopadhyay et al., 2021). Our skill-interface interventions vary failure scope and duration while memory persists across tasks.

6 Limitations

Our experiments use injected pick failures in two simulators. Diagnosis relies on simulator metadata to identify reference types and track hand state, so the results assume that these inputs are available. On a physical robot, errors in recognizing a reference object or estimating the hand state would also affect which tests can be attempted and how their outcomes are interpreted. Transfer therefore requires evaluating diagnosis with the robot's own perceptual estimates and naturally occurring failures.

Stored contracts can outlast the limitation that justified them. Target verification confirms that a failure recurs, but a temporary disruption may persist through that check and still lead to a contract. Global contracts receive periodic rechecks, whereas local contracts currently have no revocation rule. A type-specific restriction can therefore continue to block useful actions after the underlying constraint disappears. Deployments with changing local constraints would need a policy for revisiting these contracts.

7 Conclusion

WIMW shows how a failure can guide action across tasks once its scope is recorded and enforced. The value of this memory depends on both the attempts it prevents and the evidence required to justify it. The matched experiments show that reference evidence reduces failed picks under skill-wide faults in both environments, beyond what passive global inference achieves. Numerical Patch-VOI incorporates the cost of acquiring this evidence into test selection by considering the restriction that will remain after verification. For an agent facing a sequence of tasks, the cost of learning a limitation belongs in the same decision as the cost of encountering it again.

AI USE STATEMENT

Generative AI tools were used in four ways. (1) Writing: to aid and polish the prose; all scientific claims, numbers, and interpretations were written or verified by the authors. (2) Literature retrieval and discovery, such as finding related work; every cited paper was read and checked by the authors. (3) Implementation: parts of the evaluation and analysis code were drafted with AI assistance and then reviewed, tested, and corrected by the authors. (4) Data organization: AI assistance was used to collate and format experimental records, such as aggregating saved predictions into tables; all reported results come from the authors' runs, and every number in the paper was checked against the source records by the authors.

REPRODUCIBILITY STATEMENT

Section 3 defines the controllers. Appendices A and A.1 give the inference equations and numerical settings; Appendix B specifies task selection, baseline adapters, action accounting, and the matched acquisition protocol. Complementary conditions appear in Appendices C.1 and C.2. Code and archived records are available in the anonymous repository; its documentation specifies which records and reconstruction scripts are included.

ETHICS STATEMENT

The experiments use simulated embodied agents and controlled skill failures. No human participants or physical robots are involved.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. *arXiv preprint arXiv:2204.01691*, 2022. URL https://arxiv.org/abs/2204.01691.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Prithvijit Chattopadhyay, Judy Hoffman, Roozbeh Mottaghi, and Aniruddha Kembhavi. RobustNav: Towards Benchmarking Robustness in Embodied Navigation. *arXiv preprint arXiv:2106.04531*, 2021. URL https://arxiv.org/abs/2106.04531.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. AutoManual: Constructing Instruction Manuals by LLM Agents via Interactive Environmental Learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/0142921fad7ef9192bd87229cdafa9d4-Abstract-Conference.html.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. *arXiv preprint arXiv:2504.19413*, 2025. URL https://arxiv.org/abs/2504.19413.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966. doi: 10.1109/TSSC.1966.300074. URL https://ieeexplore.ieee.org/document/4082064/.

Ruofei Ju, Xinrui Wang, Xin Ding, Yifan Yang, Hao Wu, Shiqi Jiang, Qianxi Zhang, Hao Wen, Xiangyu Li, Weijun Wang, Kun Li, Yunxin Liu, Haipeng Dai, Wei Wang, and Ting Cao. EmbodiSkill: Skill-Aware Reflection for Self-Evolving Embodied Agents. *arXiv preprint arXiv:2605.10332*, 2026. URL https://arxiv.org/abs/2605.10332.

Haoyang Li, Yang You, Hao Su, and Leonidas Guibas. PhysMem: Scaling Test-Time Memory for Embodied Physical Reasoning. *arXiv preprint arXiv:2602.20323*, 2026. URL https://arxiv.org/abs/2602.20323.

Zeyi Liu, Arpit Bahety, and Shuran Song. REFLECT: Summarizing robot experiences for failure explanation and correction. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3468–3484. PMLR, 2023. URL https://proceedings.mlr.press/v229/liu23g.html.

Runyu Lu, Yubo Wu, Ethan Kou, Letian Fu, Wenli Xiao, Ajay Mandlekar, Yinzhen Xu, Guanya Shi, Ken Goldberg, Ang Chen, Mosharaf Chowdhury, Yuke Zhu, Linxi "Jim" Fan, and Guanzhi Wang. ASPIRE: Agentic /Skills Discovery for Robotics. *arXiv preprint arXiv:2607.00272*, 2026. URL https://arxiv.org/abs/2607.00272.

Baochang Ren, Yunzhi Yao, Rui Sun, Shuofei Qiao, Ningyu Zhang, and Huajun Chen. Aligning Agentic World Models via Knowledgeable Experience Learning. *arXiv preprint arXiv:2601.13247*, 2026. URL https://arxiv.org/abs/2601.13247.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. *arXiv preprint arXiv:1904.01201*, 2019. URL https://arxiv.org/abs/1904.01201.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. *arXiv preprint arXiv:2303.11366*, 2023. URL https://arxiv.org/abs/2303.11366.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. *arXiv preprint arXiv:1912.01734*, 2019. URL https://arxiv.org/abs/1912.01734.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. *arXiv preprint arXiv:2010.03768*, 2020. URL https://arxiv.org/abs/2010.03768.

Susheel Suresh, Hazel Mak, Sahil Bhatnagar, Chhaya Methani, and Alejandro Gutierrez Munoz. Grounding agent memory: Environment-probing curation for enterprise agents. *arXiv preprint arXiv:2609.11060*, 2026. URL https://arxiv.org/abs/2609.11060.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent Workflow Memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic Memory for LLM Agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. EmbodiedBench: Comprehensive Benchmarking Multi-modal Large Language Models for Vision-Driven Embodied Agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70576–70631. PMLR, 2025. URL https://proceedings.mlr.press/v267/yang25f.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629*, 2022. URL https://arxiv.org/abs/2210.03629.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

A DECISION DETAILS

Assume finite hypothesis and patch sets, bounded loss, and fixed reference demand w. Each hypothesis supplies a disabled set $\mathcal{R}_h$ and an observation model that may depend on history. For observations with positive predictive probability,

$$b_{D_{a,y}}(h) = \frac{P(y \mid a, h, D) b_D(h)}{\sum_{h'} P(y \mid a, h', D) b_D(h')}, \qquad D_{a,y} = D \cup \{(a, y)\}. \tag{8}$$

With w, L, and the patch set fixed during acquisition, the one-step rule is

$$a^*(D) \in \arg\max_{a \in \mathcal{A}(D)} \mathrm{PatchVOI}(a \mid D). \tag{9}$$

Acquisition stops when no feasible test has positive value. Demand comes from calibration and preceding requests. The objective values evidence for scope decisions, while task completion measures recovery and progress.

Proposition 1 (Indistinguishable target evidence)**.** *Suppose h_b, h_w have positive priors and $P(y \mid a, h_b, D) = P(y \mid a, h_w, D)$ for every reachable history and allowed test on the original target. A common history-dependent target-test policy leaves their posterior odds unchanged for any observation sequence with positive likelihood under h_b (equivalently h_w).*

Proof. Each update multiplies the odds by $P(y \mid a, h_b, D)/P(y \mid a, h_w, D) = 1$; action-selection probabilities also cancel under the common policy. Apply this identity successively. A reference test can change the odds when its likelihoods differ. □

Proposition 2 (Decision value of evidence)**.** *For Equation 4, $V(a \mid D) = R(D) - \mathbb{E}_y R(D_{a,y}) \geq 0$. If one patch is optimal after every positive-probability outcome, $V(a \mid D) = 0$.*

Proof. Let q^* minimize risk at D. Posterior optimization gives

$$\mathbb{E}_y R(D_{a,y}) \leq \mathbb{E}_y \mathbb{E}_{h \mid D_{a,y}} L(q^*, h; w) = \mathbb{E}_{h \mid D} L(q^*, h; w) = R(D). \tag{10}$$

Averaging the posterior inequalities shows that a patch optimal after every outcome is also optimal before the test, giving equality. This result assumes a fixed feasible patch set. Equations 6–7 additionally account for verification and remaining budgets. □

A.1 NUMERICAL IMPLEMENTATION

Table 4 evaluates the selector in Section 3.4. Each diagnosis starts with a uniform belief over skill-wide fault B, triggering-type fault W_c, and no persistent fault N. The triggering failure and shared retry each update this belief once. The type c stays fixed, and candidate updates account for existing contracts.

Candidates comprise the target and at most six distinct reference types from simulator metadata, with fixed seeds for ordering and tie breaking. Tests record success (S), unexplained failure (F), explained failure (E), or unavailability (X). The retry costs one action; subsequent test macros allow four and verification three. Each test must fit its cap and verification reserve within the twelve diagnostic actions (including retry and verification) and the 30-action episode limit. Budgets and termination are checked before every action. A macro that reaches its cap without a pick records X. Navigation, hand clearing, picks, and releases count individually. Target recovery returns the object and ends diagnosis; a successful reference test releases the object before another test. Acquisition occurs at most once per episode.

Likelihood features f encode test role, relation to the target, public navigation and hand state, and previous unexplained attempts binned $0, 1, 2, \geq 3$. Calibration equally weights hypotheses and, within N, clean execution and bursts lasting one, two, or three attempts. With $n_{f,y} = \sum_h n_{h,f,y}$, define

$$\begin{aligned} r_{f,y} &= \frac{n_{f,y} + 1}{\sum_{y'} n_{f,y'} + 4}, \quad y \in \{E, X\}, \\ \widehat{p}_{h,f}(S) &= (1 - r_{f,E} - r_{f,X}) \frac{n_{h,f,S} + 1}{n_{h,f,S} + n_{h,f,F} + 2}, \\ \widehat{p}_{h,f}(F) &= 1 - r_{f,E} - r_{f,X} - \widehat{p}_{h,f}(S), \quad \widehat{p}_{h,f}(E) = r_{f,E}, \quad \widehat{p}_{h,f}(X) = r_{f,X}. \end{aligned} \tag{11}$$

Algorithm 1 Numerical Patch-VOI with verified commitment

1	Initialize belief, incorporate the triggering failure, and run the shared retry. Return on recovery; otherwise freeze demand.
2	Enumerate verification branches for terminal proposals (Equation 6) and feasible test branches (Equation 7). Execute the highest positive-scoring test and repeat, unless recovery ends diagnosis.
3	Otherwise choose the minimizing proposal: retain memory for q_0, or run fresh target verification and apply $g(q, z)$.

If any hypothesis has fewer than ten records, all hypotheses back off to bins defined by role and attempt history, then to hypothesis-level counts, retaining the pseudocounts. Outcomes E and X preserve scope odds. Fault labels are used for offline calibration; deployment uses public feedback and fixed model weights.

Demand counts n_u come from calibration requests and preceding instructions, extracted before applying memory. For public vocabulary $\mathcal{U}$,

$$w(u) = H \frac{n_u + 1}{\sum_{u'} n_{u'} + |\mathcal{U}|}, \qquad H = 12. \tag{12}$$

Weights freeze during diagnosis; H represents reference demand, not hidden future tasks. Settings are $c_{\text{under}} = 1$, $c_{\text{over}} = 5$, $\lambda = 0.1$, $\mu = 0$.

Interaction costs are conditional means of complete macro action counts. The pooled kernel $\widehat{T}(\ell, s' \mid y, a, f)$ models action counts and terminal public state, giving branch probabilities $\widehat{p}_{h,f}(y)\widehat{T}(\ell, s' \mid y, a, f)$. Because $\widehat{T}$ is hypothesis-independent, only y updates the belief. Each branch contributes its own remaining budget to the expected risk; unsupported kernels make tests unavailable. Verification uses the same model, with z recording feedback, cost, and terminal state. It charges $c_v = \lambda \widehat{C}_v$ once for $q \neq q_0$ and zero otherwise. Target recovery or task completion permits only q_0. Ties favor no update, then narrower scope.

A.2 Evaluation protocol

Fitting, validation, development, and evaluation used disjoint task IDs, fixed along with estimator and controller settings. Each environment used eight nonoverlapping 12-task blocks, with two tasks per official subset. Each block ran in two fixed within-subset orders under Body, World, Transient, and Clean, resetting memory for every stream. Metadata and calibration rollouts stratified benchmark tasks by type diversity and navigation cost.

Selectors share the retry, belief, loss, verification, guard, rechecks, and budgets. Verification-only stops acquisition after retry and minimizes Equation 6. Fixed-order uses seeded ordering; information gain maximizes expected entropy reduction. Development selected caps of one additional test for fixed-order and two for information gain from $\{0, 1, 2\}$ by mean J, subject to one-percentage-point completion-loss and feasible-demand-restriction margins against verification-only. Ties favor fewer tests, with zero as fallback. The WIMW reference controller was rerun with the common diagnostic allowance and its original terminal proposal rule.

For workload requests u_t, let $\widehat{\mathcal{R}}_t$ denote restrictions at task entry, before actions:

$$J = \sum_t \Big(c_{\text{over}} \mathbb{I}[u_t \in \widehat{\mathcal{R}}_t \setminus \mathcal{R}^*] + c_{\text{under}} \mathbb{I}[u_t \in \mathcal{R}^* \setminus \widehat{\mathcal{R}}_t] \Big) + \lambda A^{\text{diag}}. \tag{13}$$

Requests contain each required skill–type pair once per task, extracted from specifications and initial predicates independently of trajectories, with conditional branches resolved at reset. Tasks with ambiguous request lists remain in U, A, S but are excluded from scope metrics, which require extraction-coverage reporting. Blocked requests count. Future requests and ground-truth scope are used for evaluation only. A^{diag} counts retry, acquisition, cleanup, and verification once each. Demand loss approximates this task-entry scoring. Feasible-demand restriction and uncovered disabled-demand rates divide pooled, equally weighted error counts by exposures; zero denominators are undefined. Target annotations do not certify whole-task feasibility. Count metrics weight conditions, blocks, and orders equally. The primary contrast is paired J against fixed-order acquisition; the reported aggregate means describe observed outcomes rather than tests of significance or quality margins.

B IMPLEMENTATION AND EVALUATION DETAILS

B.1 REFERENCE CONTROLLER

The ledger records successful picks and unexplained failures with their type, episode, and diagnostic role. Explained failures supply no negative scope evidence. At most once per episode, the reference routine retries the target, then considers up to six distinct reference types in seeded order, skipping unavailable actions and using best-effort navigation.

Algorithm 2 Reference WIMW after an unexplained failure on type x

1	Retry the target; retain the object and resume the task on success.
2	Test references. One success proposes type-x scope; two unexplained failures on distinct other types propose skill-wide scope. Insufficient evidence leaves memory unchanged.
3	Revisit type x, clear the hand if needed, and check again. Commit only after unexplained failure; success resumes the task, while explained or unavailable checks defer the update.

Successful reference tests release the object, whereas successful target tests retain it. Duplicate contracts leave memory unchanged, and global contracts take precedence over local ones. Every fourth proposal matching a body contract executes a recheck, counted across episodes; success clears the contract. Other matches are blocked with replanning feedback, and two blocks terminate the episode. Local contracts persist.

B.2 INTERFACES, TASK SELECTION, AND ACCOUNTING

The visual planner receives images, instructions, skills, feedback, and ten official demonstrations. Diagnosis additionally uses simulator metadata for present pickable types, navigation targets, and hand state. Faults return generic pick-failure feedback without revealing the fault condition. Full streams order the six subsets as base, common sense, complex instruction, spatial, visual appearance, and long horizon, with 50 tasks each. World targets are apple in ALFRED and plate in Habitat.

The first 60-task set (Table 2) uses sampling seed 2701 and order seeds 2702 and 2703, excluding 30 development tasks selected with seed 1701. Sampling is balanced across subsets, with at least six World-target tasks for evaluation and three for development. The second 60-task set (Table 3) uses sampling seed 3701 and order seeds 3702 and 3703. It excludes both preceding sets and contains ten tasks per subset, including six World-target tasks overall. Both evaluation sets come from the previously evaluated 300-task pool. Memory starts empty and persists within each stream; the planner and environment reset per task.

The original five methods in the 300-task tables retain their initial deployment; ExpeL, AWM, PhysMem, and WorldMind use v8 with vLLM 0.13.0. The initial runner checked episode boundaries outside an already-started diagnostic routine. Version v4 and later check before each diagnostic action and propagate recovery to the episode controller. The complete-manual study uses v5, the cache study v7, and matched acquisition uses one common frozen deployment for both arms. All executed actions, including diagnosis, releases, and rechecks, enter A; U counts all unexplained pick failures. Timeouts and failed calls retain their observed outcomes and costs. Initially satisfied goals and successful alternative branches count as completion. Two empty plans terminate an episode, as do the 30-action limit, planner-round limits, and simulator termination.

B.3 BASELINE ADAPTERS

All memory baselines update online from empty memory using the shared visual planner. Mem0 uses Qwen3-VL extraction (temperature zero, top-p one, 1,024 output tokens), all-MiniLM-L6-v2 embeddings, and Qdrant retrieval of four memories. Its input contains the task, outcome, and last ten feedback entries; the planner receives at most four 200-character lines.

EmbodiSkill maintains Planning, Search, Execution, Failure Avoidance, and execution notes, updating after each episode rather than using the original evolve-then-evaluate schedule. Its initial outer four-line cap leaves at most two truncated rules after headings. Table 2 removes that cap while retaining online updates and the internal budget of 40 lines and 4,000 characters. This repair also delivers the rules through a skill-manual wrapper.

ExpeL retains insight updates and retrieval of two successful trajectories; an unmatched failure is paired with the nearest successful task. AWM induces workflows from successful trajectories. WorldMind retains goal and process memory, prediction, and feedback verification, using the common visual planner in place of its custom planner, action repair, and metadata summaries. PhysMem retains hypothesis generation, passive verification, and promotion to principles, consolidating every five episodes and supplying at most five principles. These four adapters use a 20,000-character memory-injection cap. Oracle scope receives the persistent restriction and shares the guard and rechecks. In Clean and Transient, it starts without a contract or an added recovery routine.

B.4 Matched acquisition protocol

Table 3(b) compares Passive global+R with matched WIMW. Both allow type-local and skill-wide contracts and share recovery, verification, interception, rechecks, and termination. They differ in whether they deliberately acquire reference evidence.

Runs. Each environment uses the second 60-task set (sampling seed 3701), two task orders (seeds 3702 and 3703), and five executor seeds per order, giving ten paired streams per condition. Memory starts empty. Both arms use Qwen3-VL-32B with the same runner revision, serving configuration, fault targets, prompt templates, planner, and budgets.

Scope inference. Within an evidence epoch, the ledger retains each type's latest conclusive pick outcome across tasks: success or unexplained failure. Explained failure and unavailability leave it unchanged. A proposal requires the target's latest outcome to be an unexplained failure. Success on any other type takes precedence and yields a local candidate. Otherwise, failures on at least two distinct non-target types yield a global candidate, and fewer yield a local candidate.

Acquisition. The first eligible unexplained task pick failure triggers diagnosis, at most once per episode. After unsuccessful recovery, Passive global+R applies the scope rule immediately. Matched WIMW acquires references while scope remains unresolved: no other-type success and fewer than two non-target failures. It considers at most six other types in seeded order, stopping at the first reference success, the second distinct non-target failure, or when no test fits the budget. Both arms then apply the same proposal rule. Later failures update the ledger without restarting diagnosis.

Recovery, verification, and budgets. Recovery immediately retries the target. A recognized precondition failure permits repair (release or navigation) and one further retry. Success resumes the task. A nonempty proposal receives a fresh target check: unexplained failure commits it, success cancels it, and explained failure or unavailability defers it without changing memory. Both arms allow 30 executed actions per episode and twelve diagnostic actions, including recovery, references, cleanup, and verification. Three actions are reserved for verification. Each reference macro has a four-action cap and starts only when its cap and the reserve fit both budgets. Limits are checked before every action.

Execution and reporting. Every fourth proposal matching a body contract executes a recheck; success clears the contract and starts a new evidence epoch. Rechecks precede interception, and two actual blocks terminate an episode. Local contracts persist. Reported $S/60$, unexplained failed picks U, and total executed actions A are means and sample SDs over the ten streams, using the accounting in Appendix B.2. Paired-effect confidence intervals are not reported.

C Complementary condition results

C.1 Full-stream Transient and Clean

Table 5 completes the four-condition comparison in Table 1, using one official-order 300-task stream per cell. Clean has no eligible unexplained failure and therefore measures ordinary planning and memory use.

Table 5: Full-stream Transient and Clean results (32B; 300 tasks in official order, one stream per cell). S: completed tasks; A: all actions. Interfaces follow Table 1.

	EB-ALFRED				EB-Habitat			
	Transient		Clean		Transient		Clean	
Method	$S\uparrow$	$A\downarrow$	$S\uparrow$	$A\downarrow$	$S\uparrow$	$A\downarrow$	$S\uparrow$	$A\downarrow$
No update	136	5,558	145	5,175	145	4,733	151	4,299
Mem0	117	5,638	124	5,292	119	4,836	128	4,662
EmbodiSkill†	127	5,618	127	5,265	136	4,836	151	4,504
ExpeL	138	5,531	156	5,099	151	4,652	166	4,378
AWM	148	5,299	151	5,183	142	4,762	142	4,281
PhysMem	136	5,574	131	5,128	143	4,558	150	4,446
WorldMind	129	5,567	139	5,149	156	4,574	145	4,516
WIMW	139	5,310	144	5,113	179	4,024	143	4,363
Oracle scope	129	5,598	146	5,078	127	4,814	142	4,361

C.2 COMPLETE-MANUAL TRANSIENT AND CLEAN

Table 6 complements the persistent-fault results in Table 2. Appendices B.2 and B.3 specify sampling and the repaired delivery interface. Standard deviations describe the two orders of the same task set.

Table 6: Complete-manual EmbodiSkill (ES) and reference WIMW (W) under Transient and Clean (32B; 60 tasks, two orders; mean ± sample SD). Metrics follow Table 2; all diagnostic actions and failures are counted.

	$U\downarrow$		Success / 60 ↑		All actions ↓	
Condition	ES	W	ES	W	ES	W
EB-ALFRED						
Transient	59.0 ± 0.0	60.0 ± 0.0	26.0 ± 4.2	29.0 ± 1.4	1176.0 ± 82.0	994.5 ± 12.0
Clean	0.0 ± 0.0	0.0 ± 0.0	25.5 ± 2.1	27.5 ± 3.5	1108.5 ± 33.2	1037.5 ± 10.6
EB-Habitat						
Transient	59.0 ± 0.0	59.0 ± 0.0	26.0 ± 1.4	31.5 ± 0.7	1014.0 ± 48.1	861.0 ± 24.0
Clean	0.0 ± 0.0	0.0 ± 0.0	28.5 ± 2.1	27.5 ± 0.7	898.0 ± 32.5	849.5 ± 13.4