ROBOQUEST: GENERALIST PHYSICAL AGENTS THAT SEARCH, INSPECT AND TEST

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in multimodal foundation models have them capable generalist physical agents for a range of manipulation tasks. However, successful operation in an unfamiliar environment may require an agent to seek task-relevant information through interaction should it be absent in the observations. It may need to determine where a relevant object is, inspect an unobserved property, discover the effect of an unfamiliar tool, or manipulate the environment to make the required information accessible. We thus introduce a benchmark for goal-directed embodied exploration, where agents must actively acquire task-relevant information through physical interaction, use the resulting evidence to adapt subsequent actions, and autonomously decide when to commit to task completion. The benchmark centers on three forms of uncertainty—search, manipulation-based inspection, and interactive testing—while allowing viewpoint changes and preparatory manipulation to emerge as task-dependent strategies rather than prescribed procedures. Our evaluation mainly focuses on three frontier multimodal agents that operate through a common visuomotor interface and are evaluated by physical task outcomes, interaction efficiency, and explicit task submission. We additionally provide full-episode trajectories to support learning-based methods and evaluate a trained policy $\pi_{0.5}$, allowing zero-shot evaluation of out-of-the-box frontier robotic agents and trained vision-language-action policies under the same task framework. Our benchmark aims to study not only whether a robot can execute an action, but whether it can determine what it needs to learn from the environment, physically obtain that information, and use it to complete a goal. This also allows teasing out the failure points in the trajectory of task execution.

1 INTRODUCTION

Recent advances in generalist robotic agents have enabled increasingly complex manipulation, yet such strong execution may not necessarily translate into successful task completion without ready access to key task-relevant information in the observation. In such settings, the robot must actively determine what it needs to know, how to obtain that information, and how each new observation should shape what it does next. This may require searching an environment, inspecting objects from informative configurations, or deliberately interacting with the world to reveal its behavior. Current robotic manipulation benchmarks (Nasiriany et al., 2026; Chen et al., 2026; Tao et al., 2024; Li et al., 2026b) do not demand such open-ended information-seeking physical interactions from the agents and policies. With the increasing preponderance of generalist robotic agents, such exploratory benchmarking becomes pivotal to truly evaluate the zero-shot capabilities of such agents that may encounter unseen challenges in new environments.

Consider a robot searching an unfamiliar room in darkness with only a portable light source. It can observe only what it illuminates, yet the same gripper used to carry the light is also needed to retrieve objects. A searched compartment may contain the target, nothing at all, or a key that makes another region accessible, so each observation can change both what the robot knows and what it is able to do next. The robot must therefore decide where to look, how each discovery should redirect its behavior, and when it has learned enough to act. Similar challenges arise beyond search when a robot must determine whether an object has been inspected sufficiently or choose a physical test to probe certain hidden property of an object.

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.

The underlying principle is longstanding. Active and interactive perception study how agents can act to acquire task-relevant information rather than passively consuming fixed observations (**?**Bohg et al., 2017). Yet evaluating this capability requires more than simply hiding information from the initial observation. If every uncertainty has an obvious one-step solution, successful completion primarily tests whether the agent performs that prescribed interaction. A richer evaluation would demand the agent itself organize the investigation: different actions expose different evidence, observations inform next step, information may need to be accumulated across interactions, and acquiring it may affect the future state of the task. The challenge is therefore not merely whether a robot can reveal missing information, but whether it can conduct an effective investigation in service of a physical goal.

This question is becoming increasingly palpable as robotic agents grow stronger and more general. Frontier multimodal agents such as `GPT-6 Astra` can now act directly as robot policies across diverse, previously unseen manipulation tasks without task-specific finetuning. Improved execution shifts the bottleneck from simply performing actions toward deciding what to investigate, how to investigate it, and when enough has been learned. This creates a setting in which the quality of an agent's investigation can be meaningfully evaluated.

Thus, we introduce **RoboQuest**, a benchmark for assessing *goal-directed embodied exploration*, in which a robot must autonomously gather task-relevant evidence through physical interaction, use that evidence to guide subsequent actions, and decide when it knows enough to commit to a solution. RoboQuest contains ten tasks organized into three families: **Search & Explore**, **Object Inspect**, and **Testing**. The tasks are set in full kitchens with a mobile manipulator, so they are mobile manipulation: exploring often means driving through the kitchen as well as manipulating objects precisely. The tasks span uncertainty about where relevant objects are, what properties they possess, and what can be inferred from the outcomes of physical interventions. Rather than prescribing how the robot should investigate, each task specifies an externally verifiable physical goal and leaves the investigation strategy to the agent, from how evidence is gathered to when that evidence is sufficient to stop exploring and commit.

Across the benchmark, exploratory actions can change not only what the robot knows, but also what it can do next. Search may uncover new opportunities for interaction, while inspection and physical testing can reveal evidence that reshapes subsequent decisions. Our evaluations reveal a substantial gap between general manipulation competence and the ability to explore effectively under uncertainty. Strong robotic agents can execute many of the required physical actions, yet performance remains highly uneven when success depends on organizing exploration over time. These results highlight a persistent gap between stronger robotic execution and the ability to autonomously explore, resolve uncertainty, and achieve a given physical goal. Analysis of the execution trajectory also provides closer view at the failure points that need special attention.

Our main contributions are:

- **RoboQuest**, a benchmark for goal-directed embodied exploration with ten carefully designed tasks in which exploration shapes subsequent decisions and final task success.
- **A dataset of scripted demonstrations that include search, inspection, and test steps**, with 3,211 successful episodes (405 hours at 20 Hz) from scripts that can easily generate more, providing supervision for learning both exploratory behavior and task execution.
- **A systematic evaluation of general-purpose and trained robotic agents**, including frontier multimodal agents and a RoboQuest-finetuned $\pi_{0.5}$, together with detailed failure analysis to characterize where exploration under uncertainty breaks down.

2 Related Work

Robotic Manipulation Benchmarks. Standard manipulation benchmarks such as RLBench (James et al., 2020), CALVIN (Mees et al., 2022), LIBERO (Liu et al., 2023), ManiSkill3 (Tao et al., 2024), RoboTwin 2.0 (Chen et al., 2025), BEHAVIOR-1K (Li et al., 2023), and RoboCasa (Nasiriany et al., 2024; 2026) evaluate multi-task learning and long-horizon motor control. However, these benchmarks operate under complete or immediately accessible task observability, making the objective primarily execution-centric. In contrast, RoboQuest focuses on goal-directed embodied exploration: task-critical information is fundamentally absent from initial

Table 1: **Comparison with representative manipulation benchmarks.** ✓: explicit target; ∼: partial; –: not targeted. *Hidden*: task information absent from the current observation.

		Benchmark features				Evidence seeking		
Benchmark	**Focus**	**Long hor.**	**Hidden**	**Memory**	**Mobile**	**Search**	**Inspect**	**Test**
RLBench (James et al., 2020)	Visuomotor skills	∼	∼	–	–	–	–	–
ManiSkill3 (Tao et al., 2024)	Scalable manipulation	∼	–	–	–	–	–	–
RoboTwin 2.0 (Chen et al., 2025)	Bimanual manipulation	∼	–	–	–	–	∼	–
LIBERO (Liu et al., 2023)	Lifelong skill transfer	✓	–	–	–	–	–	–
CALVIN (Mees et al., 2022)	Long-horizon manipulation	✓	∼	–	–	–	–	–
BEHAVIOR-1K (Li et al., 2023)	Household activities	✓	∼	∼	✓	✓	–	–
RoboCasa365 (Nasiriany et al., 2026)	Manipulation in kitchen	✓	–	✓	✓	∼	∼	∼
RMBench (Chen et al., 2026)	Memory-dependent manipulation	✓	✓	✓	–	–	–	✓
RoboQuest	**Evidence-seeking manipulation**	✓	✓	✓	✓	✓	✓	✓

observations and cannot be resolved by passive viewing, requiring physical interaction to gather evidence before goal completion.

Active Perception and Interactive Exploration. Active perception centers on acting deliberately to acquire task-relevant information (Bohg et al., 2017). While recent benchmarks investigate camera repositioning (Liu et al., 2026b;a), visual occluders (Li et al., 2026b), container retrieval (He et al., 2026; Li et al., 2026a), or history retention (Chen et al., 2026), they predominantly restrict active sensing to viewpoint adjustment or single-step un-occlusion. RoboQuest expands physical exploration across three core dimensions: directed search, active inspection, and interactive testing of hidden dynamics, where exploratory interventions carry irreversible physical consequences.

Embodied Reasoning and VLAs under Uncertainty. Robotic foundation models either employ frontier multimodal models as zero-shot high-level planners (Liang et al., 2023; Yang et al., 2025) or train Vision-Language-Action (VLA) models for direct low-level control (Brohan et al., 2023; Kim et al., 2025; Physical Intelligence et al., 2025). Existing evaluations test both paradigms in fully observed scenes. When information must be actively gathered, reasoning agents often prematurely commit or hallucinate unobserved states, while Markovian VLAs struggle with multi-step hypothesis testing. RoboQuest provides an architecture-neutral benchmark to evaluate both paradigms under physical uncertainty

3 RoboQuest Benchmark

RoboQuest is a collection of ten tasks—detailed in Appendix D—requiring information-seeking physical interactions for successful completion. As visualized in Fig. 1 the information seeking interactions constitute three dominant types: search, inspect, and test. In *Search* tasks, the missing information is the location of some object. In *Inspect* tasks, it is some hidden information about an object, and the agent reveals it by interacting with the object: turning, lifting, or opening it. In *Test* tasks, the agent cannot observe the hidden information directly. We discuss the capabilities required to carry out these tasks in §3.3 and from that lens we also analyze the potential failure modes in §3.4.

Comparison with other benchmarks for physical agents. In most manipulation benchmarks, relevant information for task progression and completion is always present in the current observation/state, as compared in Table 1. In RLBench, LIBERO, and CALVIN, the agent must exercise the correct motor skills, but need not explore to uncover some necessary information. In contrast, every RoboQuest task requires information acquisition. Moreover, success still depends on perception and motor execution as well. Aggregate success rates therefore cannot isolate and probe a single capability. Hence, we analyze every episode in full to examine what evidence the agent gathered, retained, and used before it committed to some action (§5).

3.1 Benchmark Setup

Tasks. The ten tasks fall into three families by how the missing information is obtained: three Search tasks, three Inspect tasks, and four Test tasks. Figure 2 shows one demonstration of each task, while Table 7 in appendix lists what each task hides and the main parameters its instances vary, and Appendix D describes every task in full, with a frame sequence for each in Fig. 4.

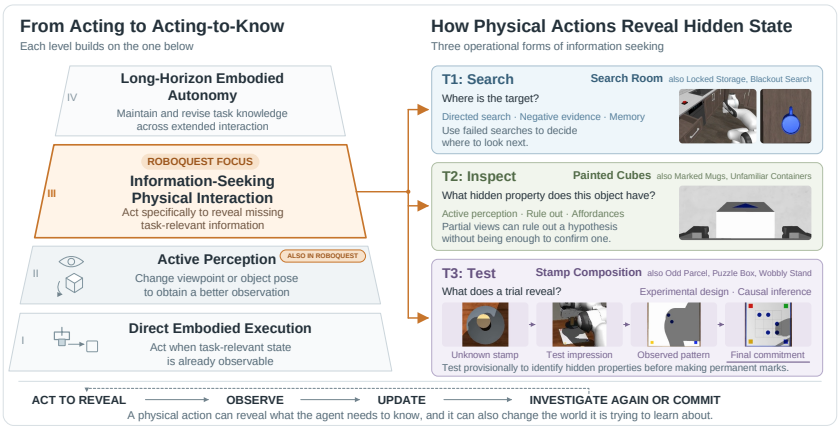

Figure 1: A visual depiction of the salient features of RoboQuest that includes tasks demanding *information-seeking physical interactions* with active perception.

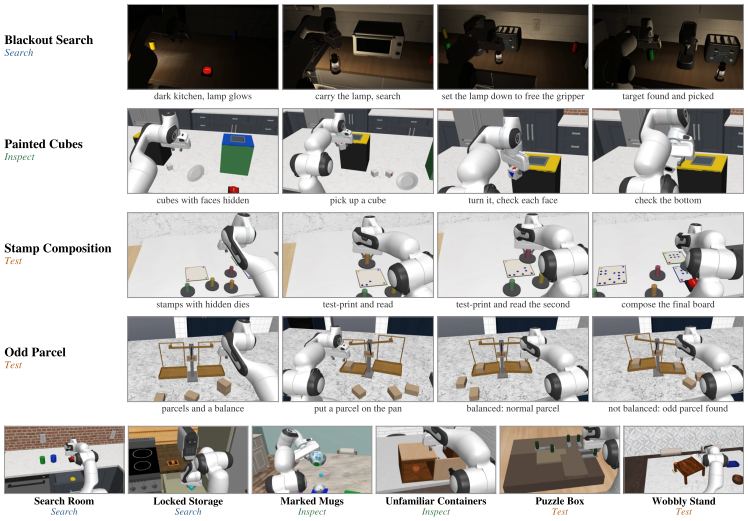

Figure 2: **RoboQuest tasks.** Top: four tasks, from the initial scene through revealing the hidden information to completion. Bottom: the other six tasks. Colours mark families.

Environment. All tasks run in RoboCasa365 kitchens (Nasiriany et al., 2026) simulated in MuJoCo (Todorov et al., 2012), with a Franka Panda arm on a mobile base controlled at 20 Hz and observed by two scene cameras and a wrist camera. Each task places its objects, containers, and mechanisms into a kitchen and states the goal as a natural-language instruction. The tasks require mobile manipulation: the storage places, containers, and objects are spread over the kitchen, so the robot must drive between them, most of all in the Search tasks, and at the Look level of occlusion (below), an object starts out of view and often out of reach from the start pose. The instruction never gives any pertinent hidden information, such as target location, the label under a vessel, which parcel is odd, or the interlocking mechanism of bolts in a box. The ground truth of each instance is kept in a private specification inaccessible to the agent.

Agent interfaces. The two kinds of agents operate in the same environment. The language-model agents act through our custom Inspect Robots (Robocurve, 2026) harness (see Appendix B) by issuing a command, such as moving the gripper to a pose or driving the base, which runs for a duration the agent chooses (see §4). On the other hand, learned policies act at the control rate, sending arm, gripper, and base commands at 20 Hz. Both act on the same physics and are scored in the same way.

Episodes and submission. An episode ends when the robot physically presses a `SUBMIT` button in the scene. The press freezes the score, so only the state at that moment counts, and the agent itself must judge when its evidence is sufficient. Some actions cannot be undone before that point: a cube dropped into a bin stays there, a token shut inside a compartment is locked away, and ink on the final surface is permanent. An episode that ends without a press counts as a failure.

Instances. Each instance is generated from a seed and a setting of the task's main parameters. The seed fixes the kitchen, the object layout, the materials, and the remaining details. Several tasks also vary occlusion at three levels. In *Visible*, every task object is in a camera view at the start. In *Look*, some object starts outside every camera view, elsewhere on the counter, on another surface, or behind a tall opaque object, and the robot must move to another viewpoint to see it. In *Uncover*, some object is hidden under a cover that the robot must remove: a cloche or an inverted bowl that must be lifted, a cup that must be tipped out, or a flat plate, board, or tray that must be pushed aside. We evaluate on 50 instances per task, which cover every level of the task's main parameters and together span 18 kitchen layouts and 5 kitchen styles. All agents play the same instances.

Scoring. An episode succeeds if and only if every goal condition holds at the press. For example, every target on the tray or every cube in its correct bin. Since a single wrong unit fails the whole episode, we also measure progress, a per-task partial credit between 0 and 1 defined in Appendix G.

3.2 Demonstration Dataset

To support learned policies, RoboQuest provides successful trajectories generated by one scripted oracle per task. The oracles read the private specification, so they know where the targets are, which mechanism works, and in which order the locks open. We nevertheless deliberately include scripted search, inspection, and test steps in their routes: the oracles open compartments nearest-first until the targets are seen, lift and tilt each vessel to read its label, turn each cube face by face, print test impressions before the final one, weigh parcels on the balance, and release the ball again after placing a shim (Table 10). An episode is kept only if it succeeds and passes automatic checks of its recording. Every frame records the three camera views, the robot state, the action, and the current subtask. Each scripted skill writes its subtask when it starts, naming objects by their appearance and position in the scene, as in "carry the red mug to the tray" or "the top drawer in the rightmost stack".

The demonstrations share no instance with the evaluation set, and evaluation holds out both the kitchen styles, which set the textures and materials of the cabinets, counters, and walls, and some task configurations, such as some of the target patterns in Stamp Composition and particular bolt-chain configurations in Puzzle Box, so no demonstration uses them. The demonstrations are long-horizon: an episode lasts 7.6 minutes on average, or about 9,100 control steps, from about 3 minutes in Puzzle Box to 16 minutes in Blackout Search, where the oracle carries a lamp from place to place and searches each one in the dark. The dataset contains 3,211 successful episodes, 405 hours of interaction at 20 Hz (Table 10), and because instances are generated from seeds, the scripts can easily generate more.

3.3 Capabilities needed for RoboQuest Tasks

Broadly, RoboQuest benchmark indicates whether an agent has the capabilities, as shown at task level in Table 2, to recognize that some task-relevant information is missing in the current observation, thus acquire it through physical interaction, and use the gathered evidence to decide how and when to act. Thus, it must run a series of trials and identify the relevant cue from the outcomes: an impression, a balance, a rolling ball, or a part that moves. Completing these tasks require execution of both information-seeking and task actions. An information-seeking action reveals information that informs latter actions to advance toward the goal. Whereas, a task action directly drives physical progress toward the goal. These two roles can overlap. For example, opening a container may reveal its mechanism and make retrieval possible. Through these actions, the agents are to exercise the required capabilities to carry out these tasks.

We detail nine of such fine-grained capabilities in Appendix F, that are categorized into four broader types *evidence acquisition*, *evidence use*, *interactive inference*, and *active organization*, as shown

		Evidence acquisition		Evidence use		Interactive inference			Action organization	
Family	Task	DS	AP	EI	M	AD	XI	CI	CA	LP
Search	Search Room	✓	∼	✓	✓	–	–	–	–	∼
	Locked Storage	✓	–	✓	∼	–	–	∼	✓	✓
	Blackout Search	✓	✓	✓	✓	–	–	–	–	∼
Inspect	Painted Cubes	–	✓	✓	∼	–	–	–	∼	∼
	Marked Mugs	–	✓	∼	∼	–	–	–	✓	∼
	Unfamiliar Containers	∼	–	–	∼	✓	∼	–	–	∼
Test	Stamp Composition	–	–	✓	∼	–	✓	–	✓	✓
	Odd Parcel	–	–	✓	✓	–	✓	–	–	∼
	Puzzle Box	–	–	–	∼	∼	∼	✓	∼	✓
	Wobbly Stand	–	–	∼	∼	–	✓	✓	–	∼



Table 2: Task capabilities and failure taxonomy. **Left**: ✓ marks a directly targeted capability, ∼ a supporting requirement, and – no designated requirement. DS: directed search. AP: active perception. EI: evidence integration. M: memory. AD: affordance discovery. XI: experimental identification. CI: physical causal inference. CA: consequential action. LP: long-horizon planning. **Right**: failures before action, during execution, and after feedback. Each failure type is linked to impairment in a likely set of capabilities defined on the left.

in Table 2 (left). It also shows the possible task-specific capability requirements and their links to various failure modes.

3.4 Failure Taxonomy

The workflow an agent generally follows to complete these tasks is (i) acquire relevant evidence, (ii) understand it, (iii) based on it reason and plan a course of action, and (iv) execute next action following the plan. Thereafter from the feedback, the agent (v) detect errors, (vi) repair or recover from them, and move ahead with a revised plan or (vii) stop at completion. Agents may fail at following these steps from lacking in the key capabilities (§3.3) and cause task failure. We broadly identify seven of these failure types: (1) *inadequate evidence gathering*, (2) *misinterpretation of evidence*, (3) *faulty reasoning or planning*, (4) *failed action execution*, (5) *failure to detect errors*, (6) *ineffective error recovery*, and (7) *incorrect termination*. In Appendix E, we detail these failures and discuss their likely relationship with the capabilities as depicted in Table 2 (right). In §5, we further quantify and discuss these failures at task level.

4 Experiments

The primary focus of this work is zero-shot evaluation of the multimodal frontier models to assess their zero-shot agentic capabilities. The trained $\pi_{0.5}$ primarily serves as a baseline to highlight the gap between the generalist and specialist models.

4.1 Experimental Settings

Zero-shot Evaluation. Owing to the impressive multimodal capabilities of the current frontier models and their success in robotic control (Zhi, 2025; Anthropic, 2026), the zero-shot evaluation is primarily focused on three frontier models: `GPT-6 Astra` (medium thinking), `Claude Fable-5.1` (medium thinking), and `Gemini-3.8 Flash` (high thinking). Every model plays the same 50 evaluation instances of each task (§3.1). All models run in the same harness, built on the Inspect Robots agent framework (Robocurve, 2026) and adapted for RoboQuest (Appendix B). At each decision the model receives its proprioception, the elapsed simulated time, and three 512×512 images from two scene cameras and a wrist camera. It replies with exactly one tool call: move the arm to an absolute gripper pose, drive the mobile base, wait, or stop. The model chooses how long each command runs, up to 10 seconds of simulated time, and there are no tools that locate or grasp objects for it. Besides a `SUBMIT` press, an episode ends when the model stops or replies without a tool call, or after 200 decisions. The conversation accumulates for the whole episode, including the model's reasoning from every earlier turn, which is passed back in the provider's native form. Only the images are pruned, to the two most recent observations. Because the history grows with every turn, we use each provider's prompt caching so that a turn reuses the prefix written by the previous

one; about 90% of input tokens are served from the cache (Table 4). The system prompt and tools are given in Appendix A.

VLA Model Evaluation. However, frontier models are often not practical due to cost, latency, and deployablity constraints. Thus, we evaluate a supervised fine-tuned (SFT) $\pi_{0.5}$ model (Physical Intelligence et al., 2025) on a set of held-in environments of the tasks of RoboQuest. The model takes three camera views (left and right base cameras, plus the wrist camera), 16-D robot proprioception, and the task prompt, predicting subtask label and 20-step action chunks (1.0 s at 20 Hz). Training minimizes a combined objective of flow-matching action loss and causal subtask prediction. At test time, the policy executes closed-loop replanning in the simulator until physical `SUBMIT` or task timeout. Training and inference procedures are detailed in §C.

Metrics. To quantify model performance, we use (i) *percentage success rate* (SR%) as the percentage of trajectories that reached the end goal, (ii) *percentage progress* (Prog%) as the percentage of progress made toward the final goal state—calculated as per Appendix G—, and (iii) *wall time* as the time spent from task start till the end. We additionally report *token* and *monetary cost* for `GPT-6 Astra`, `Claude Fable-5.1`, and `Gemini-3.8 Flash`.

4.2 Results and Analyses

Zero-shot Evaluation Results. The results in Table 3 indicates the clear performance advantage of `GPT-6 Astra` on 7/10 tasks, in terms of % success rate (SR). On three tasks, namely Search Room, Blackout Search, and Odd Parcel, the performance of `GPT-6 Astra` is 0%. However, that is not significantly less than the highest SR on Search Room and Odd Parcel of 2% by `Gemini-3.8 Flash`. On Blackout Search, all models failed throughout. On the opposite end, on Puzzle Box task, all models performed their best, leaps and bounds ahead of other task: 96%, 70%, and 12% by `GPT-6 Astra`, `Claude Fable-5.1`, and `Gemini-3.8 Flash`, respectively, 66, 56, and 10% points ahead of the next best task Painted Cubes for all the models. This outlier nature of Puzzle Box task prompts us to also report the average scores without Puzzle Box in the picture for a more balanced assessment. Overall, `GPT-6 Astra` surpasses `Claude Fable-5.1` and `Gemini-3.8 Flash` by 9.4 and 17.4 points, respectively, and without Puzzle Box, the advantage drops to 7.6 and 10 points, respectively. In proportional terms, this gap is huge: $\sim$3.3$\times$ and $\sim$12$\times$ better performance than `Claude Fable-5.1` and `Gemini-3.8 Flash`.

As expected, task progression per model also follows very similar pattern to SR. Across all the tasks, `GPT-6 Astra` also outshines the rest. Even on the three tasks with 0% SR, its mean progression significantly surpasses the rest. Overall, `GPT-6 Astra` progresses past `Claude Fable-5.1` and `Gemini-3.8 Flash` by 15.6 and 35.7 points, respectively. Ignoring Puzzle Box, this gap with `Gemini-3.8 Flash` slightly diminishes to 32.4 points.

Overall, `Claude Fable-5.1` is the second best model that is overall $\sim$3.3$\times$ worse than `GPT-6 Astra` and $\sim$3.7$\times$ better than `Gemini-3.8 Flash` the worst overall performer. At task level, only in 2/10 tasks does `Claude Fable-5.1` touch 10% SR, while `GPT-6 Astra` touches this milestone in 6/10 tasks. `Gemini-3.8 Flash` also poorly touches 10% mark in only Puzzle Box task. Although, surprisingly on Search Room and Odd Parcel `Gemini-3.8 Flash` snags a 2% SR beating the other two models' 0%, despite less mean progress than the others.

Fine-tuned $\pi_{0.5}$ Evaluation Results. Fig. 3 shows the performance of the fine-tuned $\pi_{0.5}$ models on held-out evaluation instances across RoboQuest tasks. Overall, $\pi_{0.5}$ performs substantially worse compared to frontier multimodal agents. On Puzzle Box, the policy achieves a 1.9% success rate with an average task progress of 42.7%, demonstrating that it can occasionally unbolt early sliders but fails to navigate the complete multi-stage dependency to extract the object. Across the remaining tasks, performance collapses entirely, recording 0.0% success and near-zero progress, with most episodes failing due to timeout.

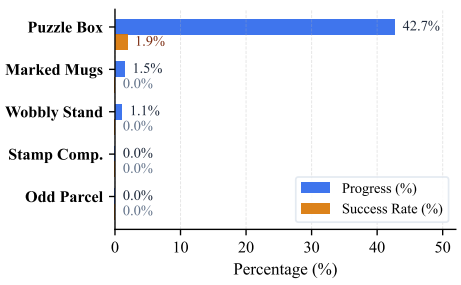

Figure 3: Fine-tuned $\pi_{0.5}$ evaluation results.

Table 3: **Main results.** SR: success rate. Prog: mean progress. Both in %, 50 episodes per cell. Best SR per row in bold. Overall w/o Puzzle Box excludes the easiest task, an outlier on which all three models score far above their other tasks.

		GPT-6 Astra		Claude Fable 5.1		Gemini 3.8 Flash	
Family	Task	SR	Prog	SR	Prog	SR	Prog
Search	Locked Storage	**28.0**	52.9	8.0	21.7	0.0	9.2
	Search Room	0.0	31.0	0.0	19.2	**2.0**	17.5
	Blackout Search	0.0	23.7	0.0	17.1	0.0	4.4
Inspect	Painted Cubes	**30.0**	74.8	14.0	61.5	2.0	11.3
	Marked Mugs	**10.0**	43.0	0.0	16.6	0.0	0.0
	Unfamiliar Containers	**6.0**	32.2	0.0	17.0	0.0	6.3
Test	Puzzle Box	**96.0**	99.6	70.0	81.0	12.0	33.0
	Stamp Composition	**14.0**	55.3	2.0	36.9	0.0	3.4
	Wobbly Stand	**10.0**	18.6	6.0	14.4	2.0	4.6
	Odd Parcel	0.0	18.3	0.0	7.2	**2.0**	2.0
Overall		**19.4**	44.9	10.0	29.3	2.0	9.2
Overall w/o Puzzle Box		**10.9**	38.9	3.3	23.6	0.9	6.5

Table 4: **Cost and effort per episode.** Sim / Wall: simulated / wall-clock minutes. In / Out: tokens in millions. Costs in USD at list prices.

	Effort per episode			Tokens per episode			Cost (USD)	
Model	Decisions	Sim min.	Wall min.	In (M)	Out (M)	Cached (%)	$/episode	$/success
GPT-6 Astra	143.7	10.4	53.8	5.82	0.02	91.8	11.30	58.2
Claude Fable 5.1	148.8	6.1	54.7	14.55	0.11	94.2	19.22	192.2
Gemini 3.8 Flash	160.8	5.1	41.0	19.66	0.13	89.3	3.37	168.3

This failure highlights the challenge of out-of-distribution (OOD) generalisation in goal-directed exploration. The evaluation instances used novel kitchen styles, materials, task mechanisms and target configurations that are not present in the training dataset (§3.2). Thus, the benchmark cannot be solved by simply memorizing state-action mappings or trajectory sequences from training data. Because standard VLA architectures such as $\pi_{0.5}$ lack explicit episodic memory, minor execution errors in OOD scenes quickly compound, leaving the policy stuck in repetitive loops or idling until the decision horizon expires.

Cost Analysis. In Table 4, the model-level comparison reveals substantial differences in economic, compute, token, and inference efficiency. `GPT-6 Astra` exhibits the strongest outcome-adjusted economic efficiency, achieving the lowest cost per successful episode at $58.20, with `Claude Fable-5.1` costing $\sim$3.3$\times$ and `Gemini-3.8 Flash` $\sim$2.9$\times$ as much per successful episode. Although `Gemini-3.8 Flash` has the lowest per-episode cost, `GPT-6 Astra` demonstrates substantially greater token efficiency, with `Claude Fable-5.1` consuming $\sim$2.5$\times$ and `Gemini-3.8 Flash` 3.4$\times$ as many input tokens as `GPT-6 Astra`, while their output-token consumption is $\sim$5.5$\times$ and 6.5$\times$ higher, respectively. In terms of effort, `GPT-6 Astra` requires the fewest model decisions or turns, with `Claude Fable-5.1` requiring 4% and `Gemini-3.8 Flash` 12% more decisions per episode. However, `GPT-6 Astra` incurs the greatest simulated robot time, requiring $\sim$1.7$\times$ that of `Claude Fable-5.1` and 2.0$\times$ that of `Gemini-3.8 Flash`. Conversely, `Gemini-3.8 Flash` exhibits the strongest temporal efficiency, with `GPT-6 Astra` and `Claude Fable-5.1` requiring $\sim$1.31$\times$ and 1.33$\times$ its wall-clock time, respectively. `Claude Fable-5.1` achieves the highest cache hit-rate, $\sim$1.03$\times$ and 1.05$\times$ that of `GPT-6 Astra` and `Gemini-3.8 Flash`, although this modest advantage does not offset its comparatively high token consumption and monetary cost. Overall, the results indicate a clear trade-off between execution speed and outcome-adjusted economic efficiency: `Gemini-3.8 Flash` provides faster and less expensive individual episodes, whereas `GPT-6 Astra` achieves substantially greater token efficiency and a markedly lower cost per successful outcome.

5 Where Does Exploration Break Down?

While §4 demonstrates that both frontier models and VLA policies struggle on goal-directed exploration tasks, aggregate task metrics alone cannot reveal why exploration fails. To understand where

the connection between goals, physical evidence, and motor action breaks down, we attribute every failure of `GPT-6 Astra` and `Claude Fable-5.1` to our failure taxonomy (§3.4) by auditing the final simulator state, hidden specifications, and recorded trajectories per *unmet requirement* (Table 15). We present a detailed empirical analysis, statistical contrasts, and per-task case studies in Appendix K; below, we summarize the primary findings across the core capability dimensions:

Evidence Acquisition is the Dominant Bottleneck. More than half of all failures across both frontier models occur before task solving begins (56% for `GPT-6 Astra` and 58% for `Claude Fable-5.1`). Inadequate evidence gathering is by far the largest failure mode (37% and 42% overall). Rather than misidentifying targets they have already inspected, agents primarily fail by terminating exploration prematurely while relevant search spaces remain unvisited. For instance, in Search Room and Blackout Search, the gripper never approached within 25 cm of over 83% of the missed targets. Similarly, in Painted Cubes, incomplete inspection across cube faces drives a heavy skew toward false negatives over false positives (Fig. 6), showing that models stop checking faces too early.

Workload Degrades Evidence Use and Planning. Increasing task workload (the number of items or dependencies to track) causes a steep decline in success, dropping by 15.8 points for `GPT-6 Astra` and 7.2 points for `Claude Fable-5.1`, far exceeding the drop caused by visual occlusion alone (Table 18). Analyzing failures across the interaction workflow reveals that this workload penalty stems primarily from missing evidence before action and unattempted goals at termination (Table 19), rather than increased execution difficulty. This exposes fundamental limitations in long-horizon planning and context memory when evidence must be accumulated across extended multi-step trials.

Interactive Inference and Action Organization. Under tasks demanding causal hypothesis testing such as covered Puzzle Box, models exhibit stark divergence: `GPT-6 Astra` successfully recovers the hidden bolt dependency through systematic trial actions (96.4% SR), whereas `Claude Fable-5.1` drops from 100% to 34.6% by repeatedly forcing jammed sliders (Table 20). Finally, even when correct evidence is gathered, physical execution errors account for 15-21% of failures. Models diverge sharply in error response: `GPT-6 Astra` attempts more actions but fails to detect dropped objects in 57% of cases, whereas `Claude Fable-5.1` displays excessive caution, terminating or stopping prematurely in 28.6% of episodes (Table 21).

6 Conclusion

The importance of RoboQuest is highlighted by the empirical performance and cost incurred by `GPT-6 Astra`, `Claude Fable-5.1`, and `Gemini-3.8 Flash`. All three generalist frontier physical agents succeed on only a small fraction of episodes: `GPT-6 Astra` achieves the highest success rate at 19.4%, while `Gemini-3.8 Flash` achieves the lowest at 2%; `GPT-6 Astra` is also the most economical of the three. Beyond aggregate performance, our failure analysis reveals where goal-directed exploration breaks down. For both `GPT-6 Astra` and `Claude Fable-5.1`, more than half of all failures occur before task solving begins, with inadequate evidence acquisition emerging as the dominant bottleneck: agents frequently terminate exploration while relevant regions or object states remain uninspected. Increasing task workload further degrades performance, primarily by exacerbating missed evidence and leaving goals unattempted rather than by making physical execution itself more difficult, exposing limitations in long-horizon planning and context memory. Tasks requiring interactive inference additionally reveal substantial differences between agents in their ability to use actions to resolve hidden causal dependencies. Finally, even when sufficient evidence is acquired, execution and error recovery remain significant sources of failure, with agents exhibiting contrasting tendencies toward undetected action failures and premature termination. Together, these results show that progress on embodied exploration requires not only stronger perception and control, but also more systematic evidence acquisition, persistent goal and evidence tracking, causal hypothesis testing, and robust monitoring and recovery during physical interaction.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experiments, assist with method implementation, create or modify scientific figures, summarise existing literature, and draft or edit portions of the manuscript for clarity. We did not use these tools to develop theoretical models, formulate or prove mathematical claims, propose or refine hypotheses, clean or reformat datasets, generate synthetic data, or translate the manuscript. We manually checked LLM-generated code for correctness, verified literature summaries against the cited sources, and reviewed all AI-generated text and figures. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Anthropic. Claude plays robotics. `https://www.anthropic.com/research/claude-plays-robotics`, 2026. Accessed: 2026-09-26.

Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, et al. Rmbench: Memory-dependent robotic manipulation benchmark with insights into policy design. *arXiv preprint arXiv:2603.01229*, 2026.

Yuxin He, Ruihao Zhang, Tianao Shen, Cheng Liu, and Qiang Nie. Towards exploratory and focused manipulation with bimanual active perception: A new problem, benchmark and strategy. *arXiv preprint arXiv:2602.01939*, 2026.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pp. 80–93. PMLR, 2023.

Jialiang Li, Yi Qiao, Yunhan Guo, Changwen Chen, and Wenzhao Lian. Act, sense, act: Learning non-markovian active perception strategies from large-scale egocentric human data. *arXiv preprint arXiv:2602.04600*, 2026a.

Taishan Li, Jiwen Zhang, Siyuan Wang, Xuanjing Huang, and Zhongyu Wei. Libero-occ: Evaluating and improving vision-language-action models under scene-induced occlusion via viewpoint imagination. *arXiv preprint arXiv:2606.10862*, 2026b.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International conference on robotics and automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Mengzhen Liu, Enshen Zhou, Cheng Chi, Yi Han, Shanyu Rong, Liming Chen, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Sapave: Towards active perception and manipulation in vision-language-action models for robotics. *arXiv preprint arXiv:2603.12193*, 2026a.

Zhenyang Liu, Yongchong Gu, Yikai Wang, Xiangyang Xue, and Yanwei Fu. Activevla: Injecting active perception into vision-language-action models for precise 3d robotic manipulation. *arXiv preprint arXiv:2601.08325*, 2026b.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations (ICLR)*, 2026.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Robocurve. Inspect robots: The open-source evaluation framework for physical ai, 2026. URL `https://github.com/robocurve/inspect-robots`.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, et al. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. *arXiv preprint arXiv:2502.09560*, 2025.

et al. Zhi. Closed-loop open-vocabulary mobile manipulation with gpt-4v. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025.

A System Prompt and Tools

Every model receives the same system prompt, sent once at the start of the episode. Only the goal differs between tasks; the goal text is identical across a task's occlusion variants. The prompt for Puzzle Box reads:

> You control a simulated mobile manipulation robot. Use only the provided camera images, robot proprioception and action/budget feedback. Determine what the goal requires from the scene. Choose exactly one arm, base, wait or stop tool per response. There are no semantic object-location or grasp tools. Images are upright RGB from three fixed cameras; new views require physical robot or object motion. The arm tool controls the right gripper site in world metres, with optional world orientation as quaternion xyzw. World z is up. Gripper -1 opens, +1 closes and 0 keeps its current setpoint. The base tool takes normalized body-frame forward, left and yaw velocity commands; positive yaw turns left. The arm stays relative to the base during base motion. The torso is held. A tick is 0.05 seconds.

> Commands consume up to 200 ticks each, ending early on the first physical `SUBMIT` button press. Check observed outcomes: commanded targets are not guaranteed to be reached. Physically press `SUBMIT` to commit the result; the first press freezes the score and ends the episode immediately. Wrong submission, timeout without submission, or calling stop without submission fails. The stop tool abandons the episode without advancing time. Invalid or multiple tool calls perform no physics and still consume this model response. No intermediate success is awarded. Goal: Open the wooden puzzle box on the counter, take out the tangerine inside, and stand it inside the tray. The lid slides sideways but is locked by sliding bolts with round black handles: a part moves only when nothing blocks it, so work out the order. Any order that works is fine. Once the tangerine is inside the tray and you have let go of everything, press the red `SUBMIT` button. Your first submission ends the episode; a wrong submission or no submission fails. Control contract: mobile-native-control-v1.

Table 5 lists the four tools. After each call the model receives a short result with three fields: whether the command was accepted, the current tick count, and an error that is either empty or the fixed message "Invalid action; check schema and remaining budget." The result never contains simulator exceptions, scores, or task state.

Table 5: **Action tools.** Positions are in world metres; a tick is 0.05 s of simulated time. Every tool except `stop` accepts a tick count from 1 to 200.

Tool	Arguments	Default ticks
`arm`	gripper-site position (required); orientation as a quaternion; gripper command in $[-1, 1]$, where -1 opens and $+1$ closes	120
`base`	body-frame forward, left, and yaw velocity, each in $[-0.5, 0.5]$	20
`wait`	optional gripper command; holds the arm and base	20
`stop`	none; abandons the episode, which fails	–

B Evaluation Harness

Lineage. The harness builds on the Inspect Robots agent plugin (plugin version 0.26.0, MIT license), the framework used for the public `GPT-6 Astra` real-robot runs. We reuse its provider clients for the OpenAI Responses and Anthropic Messages interfaces, their message formats, native reasoning continuation (encrypted reasoning items and thinking signatures), its rule for dropping old images, and its pattern of one tool call per turn. The simulator (RoboCasa kitchens in MuJoCo, with a Panda arm on a mobile base), the observation filter, the action tools and their execution, the submission protocol, the system prompt, the budget rules, retry and restart handling, prompt caching, a native Gemini client, and all logging and scoring are ours.

Observations. Each decision adds one user message. Its text part gives the end-effector and base poses, the 13 joint positions and velocities rounded to millimetres and milliradians, the elapsed ticks and simulated seconds, and the number of model responses left, counting down from 200. Its image part holds three labelled 512×512 RGB images from the left and right scene cameras and the wrist camera, without overlays. A positive allowlist builds every observation, so no task specification, seed, instance identifier, or hidden ground truth can reach the model.

Actions and execution. The model must reply with exactly one tool call (Table 5) and chooses its duration in ticks. An arm command runs a proportional servo on the measured gripper pose through the robot's operational-space controller for the requested number of ticks, a base command applies the velocity each tick with the arm held relative to the base, and a wait holds both. A command runs for exactly its requested ticks, with no added settling time, and stops early only when the `SUBMIT` button is pressed. A reply with no tool call, several tool calls, or arguments outside the schema runs no physics but still uses up the decision; a reply without any tool call also ends the episode.

Memory. Every episode starts a fresh conversation, with no memory across episodes and no notes from earlier runs. Within an episode the conversation only grows: the system prompt, then for every decision the observation, the model's full reply including its native reasoning content, and

the tool result. Text is kept for all decisions. Images are removed from all but the two most recent observations, so six images are in context at a time, while the text of older observations, and with it the full proprioceptive history, stays. Nothing is summarized. To keep costs manageable, the Anthropic and OpenAI clients mark cache breakpoints on the newest observations whose images have been dropped, so each turn reads the prefix the previous turn wrote; the Gemini client replays the full history with every turn's thought signature. If a provider rejects a reasoning continuation, the harness archives the conversation and restarts it from the system prompt and the current observation, at most twice per episode and without replaying any physics.

Budgets and termination. An episode allows 200 model decisions and at most 200 ticks per command. Each request may be retried up to six times, and retries do not count as decisions. The tasks also define a budget in ticks, but it is neither shown to the model nor enforced in these runs, so the decision limit is the binding one. An episode ends at the first `SUBMIT` press, after 200 decisions, or when the model calls `stop` or replies without a tool call. Only the state at the press is scored, and the scene freezes that score when the button is pressed. Episodes interrupted by the provider or the runtime are left unscored rather than counted as failures, and each reply's model identifier is checked before its action runs.

Changes from Inspect Robots. The published `GPT-6 Astra` configuration of Inspect Robots allows 20 model calls with images on every turn and computes motion durations from speed limits. We change it in the following ways.

1. **Physical submission replaces `done` and `give_up`.** Upstream scores success when the model declares it or when time runs out, so a policy can reach a correct state by trying several answers. Pressing a physical button commits a single answer, which is what makes information gathering meaningful: the policy has to decide when it knows enough.
2. **The model chooses durations, and nothing settles automatically.** Every tick is charged, and elapsed time is visible in each observation, so managing time is part of the policy's task.
3. **A new tool set.** Upstream's generated tools only support its own rotation format. We use absolute world poses with quaternions, add base and wait tools, and remove the mandatory note, the picture request, and the `done` and `give_up` tools. There are no semantic helpers, so every view has to be earned by moving.
4. **The goal is stated once.** The goal and the control contract are in the system prompt rather than repeated in every observation, and proprioception is rounded, which keeps each turn short and the prompt cacheable.
5. **The decision budget is visible.** The model sees how many responses remain but no countdown of physical time, since a countdown would change when a policy chooses to submit.
6. **Observations only.** Upstream supports operator feedback, hindsight notes, and learnings carried between trials. We leave all of them out, so the benchmark measures what a policy does within one episode.
7. **Accounting and robustness.** One accepted response counts as one decision, retries are counted separately, truncated or invalid replies never execute a partial action, continuation failures are recovered by restarting the conversation, and token usage and cost are recorded for every provider.
8. **Prompt caching.** With 200 decisions of three images each, re-sending the conversation dominates the cost, so cache breakpoints let each turn reuse the previous prefix.
9. **Scale.** Episodes allow 200 decisions instead of 20, with 512×512 images from three cameras, including the wrist.

Every episode in the release keeps its system prompt, tool schemas, instruction, configuration, full and archived conversations, per-decision images, a tick-level physics trace, a video, and its token usage and cost.

C $\pi_{0.5}$ FINE-TUNING AND EVALUATION DETAILS

In addition to evaluating zero-shot frontier language-vision models (§B), we evaluate a supervised fine-tuned $\pi_{0.5}$ model. This section summarises the observation mapping, action representations, training objective, and hyperparameters used.

Observation Mapping. At each decision step, the model ingests:

- **Visual inputs:** Three RGB camera streams: left and right mobile base views and a wrist camera view, each resized to 224×224 pixels.
- **Proprioceptive state:** A 16-dimensional continuous robot state vector.
- **Language prompts:** The high-level task goal instruction provided in the task specification.

Action Space and Horizon. The action expert generates an action chunk of length $H = 20$ time steps, which corresponds to 1.0 second of continuous execution at the simulator's native 20 Hz control frequency. Each action predicts a 12-dimensional hybrid mobile manipulation command: 6-D end-effector delta pose, 1-D continuous gripper action (± 1), 3-D mobile base velocities (forward, lateral, and yaw), and 2-D torso pose.

Training Objective and Joint Modeling. In the joint formulation, training simultaneously optimizes an autoregressive language objective for high-level subtask prediction and a conditional flow-matching objective for low-level action chunking:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{subtask}} + 10 \cdot \mathcal{L}_{\text{flow}}, \tag{1}$$

where $\mathcal{L}_{\text{subtask}}$ is standard cross-entropy loss over causal subtask tokens, and $\mathcal{L}_{\text{flow}}$ is the flow-matching loss conditioned on vision, state, prompt, and the teacher-forced subtask.

Hyperparameters. Table 6 summarizes the key training and inference hyperparameters. Fine-tuning uses the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 1×10^{-4}, and gradient norm clipping at 1.0. The learning rate follows a cosine decay schedule with a linear warmup of 1000 steps, starting from a peak learning rate of 2.5×10^{-5} down to a minimum of 2.5×10^{-6}. Global batch size is set to 32 using full-parameter training with exponential moving average (EMA) weight tracking enabled.

Table 6: $\pi_{0.5}$ **Training and Inference Hyperparameters.**

Category	Hyperparameter	Value
Training	Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$)
	Weight Decay	1×10^{-4}
	Peak Learning Rate	2.5×10^{-5}
	Learning Rate Schedule	Cosine decay with 1000 warmup steps
	Global Batch Size	32
	Gradient Clipping Norm	1.0
	Flow Loss Multiplier	10.0
	Action Normalization	Quantile (q_{01}, q_{99})
Inference	Replanning Frequency	Once per 20 ticks (1.0 s)
	Flow Denoising Steps	10

During evaluation, the policy runs in closed loop, replanning every 20 ticks (1.0,s) until the episode terminates upon pressing the physical `SUBMIT` button or exceeding the task horizon.

D BENCHMARK TASKS

RoboQuest organizes evaluation tasks into three families according to the dominant information-gathering behavior required during execution: *Search & Explore*, *Object Inspect*, and *Testing*.

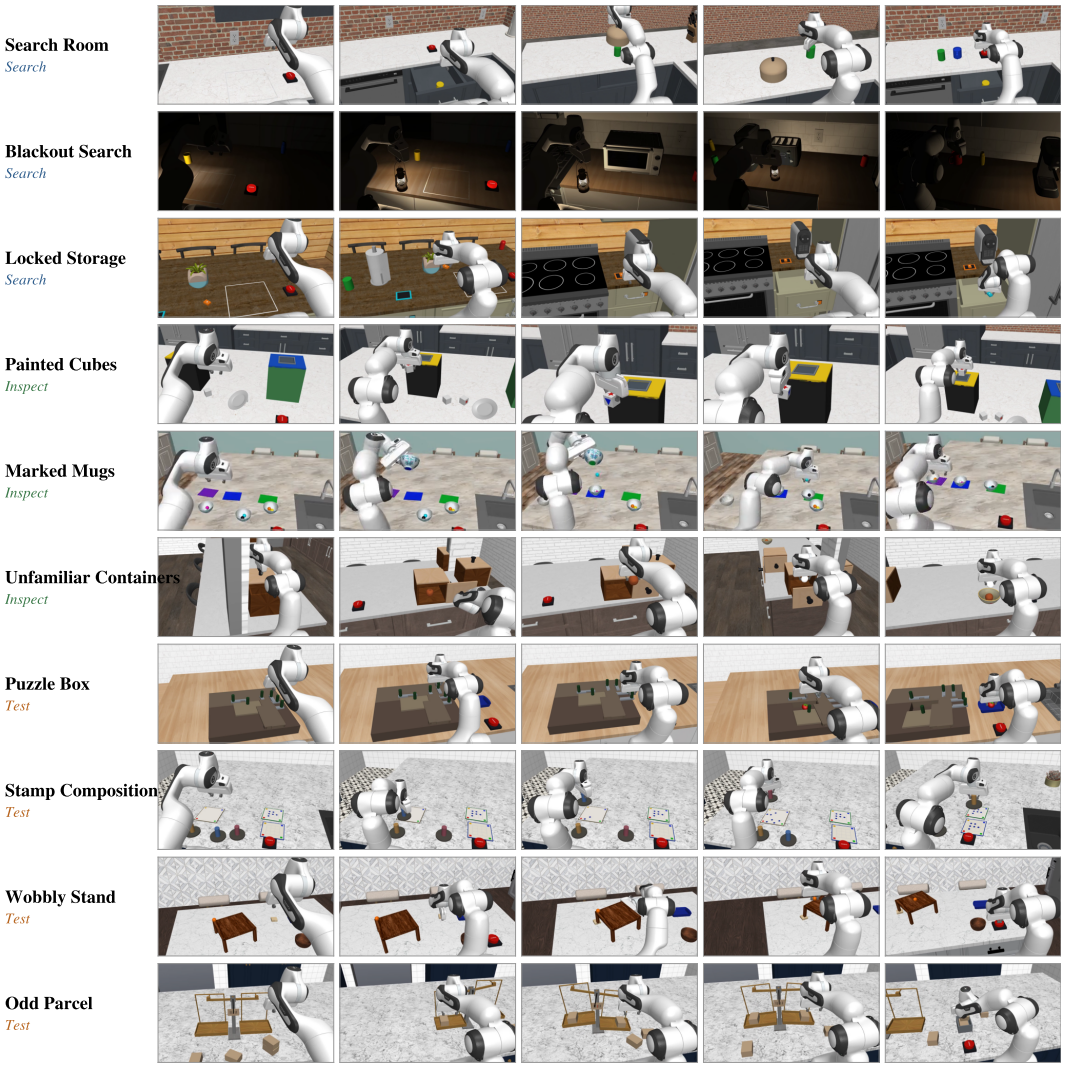

Figure 4: **All ten RoboQuest tasks.** Five frames from one scripted demonstration of each task, from a scene camera, running from the initial scene through the actions that reveal the hidden information to the completed task. Family labels are given in colour.

D.1 Search & Explore

Search Room. A set of target items is stored in open or closed locations around a room, among distractor items of similar kind. The goal is to collect the complete set on the tray and press `SUBMIT`. The locations, the number of storage places, and which storage places are empty are not given. The robot must choose where to search, use negative findings to redirect, remember what it has checked, and stop once the set is complete. Instances vary the set size, item category, room layout, number and type of storage units, and distractors.

Locked Storage. Storage compartments open only when a matching access token rests on their reader. Tokens are either outside the initial observation or hidden inside other storage, and some compartments contain tokens for other compartments, forming a dependency chain. The goal is to retrieve the target item. Finding a token changes which places can be searched next. Closing a compartment with a token still inside locks that token away permanently, so the robot must plan token handling as well as search order. Instances vary the number of locked compartments, chain structure, token hiding places, and dead ends.

Blackout Search. A search task is performed in an unlit room. Only a light source with a glow marker and the SUBMIT button are visible at the start. The robot must acquire the light source, carry it, and search using a wrist camera that sees only what is illuminated. Because one gripper holds the light, retrieving a target requires putting the light down while aiming it at the target, then acting from memory in the dark. Instances vary the target, room, location of the light source, whether a wall switch exists, and whether a timed light turns off after a delay.

D.2 Object Inspect

Marked Mugs. Several identical vessels, all mugs or all bowls in a given instance, each hold loose contents, and a label on the underside of each vessel specifies its destination. The goal is to place every vessel upright at its destination with its original contents inside. Reading a label requires lifting or tilting the vessel, which spills the contents; spilled contents stay nearby and must be returned to the correct vessel before submission. Instances vary the number of vessels, contents, label-to-destination mapping, and initial poses.

Painted Cubes. A collection of similar objects each carries a hidden property distributed over its faces, such as the number of painted faces, with some faces turned toward the table or away from every camera. The goal is to collect every object satisfying a stated property, for example, exactly one painted face. Partial views can reject an object but cannot confirm it, so the robot must decide how much inspection each object needs. Instances vary the number of objects, property rule, initial orientations, and whether the goal asks for one qualifying object or all of them.

Unfamiliar Containers. Several closed containers each open by a different mechanism, such as a hinged lid, sliding lid, drawer, or latched flip-top, and some cannot be opened at all. The target is inside one of them. The robot knows neither where the target is nor how each container opens and must discover each mechanism by interaction. Instances vary the number of containers, mix of mechanisms, target location, and number of decoys.

D.3 Testing

Stamp Composition. Several stamps with unmarked housings each carry a hidden pattern and start at an unknown rotation. A target pattern is shown on a reference. A test surface allows trial impressions, and a final surface must end with exactly the target. The robot must test stamps, read the results, select a subset and their rotations, compose the target, and press SUBMIT. Marks on the final surface are permanent. Instances vary the number of stamps, patterns, target, initial rotations, and size of the test surface.

Odd Parcel. A set of visually identical sealed parcels contains one or two parcels that differ in weight. The only instrument is a two-pan balance whose pans hold several parcels each. The goal is to place the odd parcel alone on the tray. Group weighing identify it in few comparisons, while one-by-one weighing is legal but slower. Instances vary the number of parcels, whether the odd parcel is known to be heavier or could be lighter, pan capacity, and parcel appearance.

Puzzle Box. A container is held closed by a chain of interlocking parts, where each part blocks another until it is moved. All parts are visible with large handles, but the blocking relations are not. The goal is to place the item inside on the tray. The robot must discover the release order by trying moves and observing what moves, and some wrong moves jam another part until reversed. Instances vary the number of parts, chain structure, presence of red-herring parts, and trap moves.

Wobbly Stand. A stand or table has an uneven support, tilting its surface by an amount too small to see directly. A ball placed on the surface rolls toward the low side and reveals the tilt. Shim blocks of different thicknesses are available. The goal is a level surface on which the ball stays put, with everything released. Instances vary the number of legs affected, tilt magnitude, shim set, and whether the tilt is along one axis or two.

Table 7: **Tasks and their main parameters.** Occlusion: Visible (V), Look (L), Uncover (U), defined in §3.1. W, O, C: tasks used for analysis of workload, occlusion, and cover in §5.

Task	Hidden information	Main parameters		W	O	C
Search						
Locked Storage	target behind locks	chain depth 1/2/3		✓		
Search Room	where the targets are	compartments 4/6/8	targets 2/3			
Blackout Search	targets in the dark	compartments 4/6/8	targets 2/3			
Inspect						
Painted Cubes	marks on unseen faces	cubes 4/6/8	occlusion V/L/U	✓	✓	
Marked Mugs	labels under vessels	vessels 3/4/5	occlusion V/L/U	✓	✓	
Unfamiliar Containers	how containers open	items 2/3/4	occlusion V/L	✓		
Test						
Puzzle Box	bolt blocking order	chain 4/5/6	cover no/yes			✓
Stamp Composition	patterns and rotations	stamps 2/3/4	occlusion V/L/U	✓	✓	
Wobbly Stand	short legs and gaps	short legs 1/2	shim choice no/yes			
Odd Parcel	which parcel is odd	parcels 4/6/8	occlusion V/L/U	✓	✓	

E Failure Taxonomy Details and Link to Capabilities

Inadequate evidence gathering concerns missing information where the agent prematurely acts decisively before obtaining key evidence for a reliable decision. For example, it may leave a relevant location unexplored, fail to unveil a distinguishing information, or omit a diagnostic test. In Marked Mugs, placing a vessel before inspecting the label underneath would be an instance of such failure. Lacking in *evidence acquisition* and/or *interactive inference* capabilities curtail agent's capability to gather enough key evidences.

Misinterpretation of evidence acquired in the context of the task may lead to erroneous downstream decisions. For instance, the agent observes the correct object, yet misidentifies it, misreads a mark on it, or misunderstands the result of an interaction with it. In Marked Mugs, misreading the label underneath a vessel would likely lead to a wrong placement of it, precipitating into a task failure; a possible manifestation of the lack of *evidence usage* capabilities. Acquisition and interpretation therefore fail at different points. The former lacks the needed observation and the latter assigns a wrong meaning leading to a task failure.

Faulty reasoning or planning around the gathered evidences may produce an ineffective or erroneous plan. An agent may lose track of a relevant earlier result, use stale information, omit a goal constraint, make incorrect inferences from prior results, or choose an invalid action order. For example, in Odd Parcel task, identifying the odd parcel requires combining the results of balance comparisons before identifying the parcel. On the other hand, correct evidence gathering can still lead to an incorrect plan due to reasoning failure. A lack of *evidence use*, *interactive inference*, and/or *action organization* capabilities may contribute to such failures.

Failed action execution concerns the failure to achieve the intended result from an executed action. For instance, a grasp can miss, an object can fall during transport, or it can be misplaced. This is very likely a fault in *action organization* capabilities.

Failure to detect errors that matter to task completion may keep the agent away from task progress and completion—likely, a combination of faults in *evidence acquisition, use,* and *action organization*. For instance, in Wobbly Stand task, the failure to gauge the tilt of the stand by tracking the ball on top would inexorably lead to task failure.

Ineffective error recovery may also perpetually keep an agent away from progressing toward the goal, that may stem from impaired *interactive inference* and/or *action organization*. For instance, a fallen object of interest that remains on the floor despite repeated retrieval attempts may stop meaningful progression. Recovery requires both detecting the discrepancy and thereafter acting toward a corrected state.

Incorrect termination involves submitting at an end state misjudged as success or prematurely abandoning a goal that remains reachable, possibly a case of a lack of *evidence use* or *action organization*. A dropped tomato followed by unsuccessful retrieval and pressing `SUBMIT` illustrates a chain across execution, recovery, and completion. Reaching the `SUBMIT` button completes that action while leaving the original goal unfinished.

F FINE-GRAINED CAPABILITIES FOR ROBOQUEST

Directed search is the core of the Search family where the agent chooses where to look, track where it has checked, and therefrom determine next likely search zone. An unobserved item is not necessarily absent, so an empty location counts only if the agent has actually checked it. For instance, in Search Room, the agent must judge when the remaining search space is exhausted, because the number of storage places and which of them are empty are not given in the instruction. Locked Storage adds dependency: finding a token changes which places can be searched next, so the search order must follow a chain the agent discovers along the way. Blackout Search removes free observation: the agent sees only what its light source illuminates, and it must put the light down to grasp the target; it then acts from memory in the dark. All three tasks require **evidence integration**, since each observation matters only in combination with what the agent has already seen.

Active perception is the core of the Inspect family where the agent actively scans for pertinent information through interaction, informing next pivotal task actions. In Painted Cubes, the target may be a cube with exactly one painted face. Two painted faces rule a cube out. One painted face does not confirm it while other faces remain hidden. A partial view can therefore reject a candidate, but it cannot confirm one, so the agent memorize the prior observations of the faces. **Memory** is thus a key capability to keep track of the progress and retrieve relevant information from past observations and inferences. In Marked Mugs, the label under each container can be read only by lifting or tilting it and decide next action. In Unfamiliar Containers, the hidden property is the opening mechanism itself. The agent must discover the **affordances** of each container through interaction, and it must recognize that some containers cannot be opened at all.

Experimental identification is the core of the Test family where the agent chooses trials that separate candidate explanations and adapt latter trials to earlier outcomes. In Stamp Composition, trial impressions on the test surface reveal stamp patterns and rotations. In Odd Parcel, a two-pan balance is the only instrument. One-by-one weighing is valid but slow, while group weighing identify the odd parcel in much fewer comparisons. The task therefore measures the quality of experimental design, not only its outcome. **Physical causal inference** links an action to its physical consequences. In Puzzle Box, the agent learns the release order by trying moves and observing which parts move. Puzzle Box is the simplest case of this capability, and the strongest model solves most of its episodes. In Wobbly Stand, the tilt is too small to see. A ball placed on the surface rolls toward the low side and reveals it, and the agent must then relate each shim to the change it produces. These tasks test local inference from physical consequences, not general causal understanding.

Gathering information also has physical consequences. In Marked Mugs, reading a label may spill the contents, and the agent must recollect them before it submits. In Locked Storage, closing a compartment with a token inside locks that token away for good. In Puzzle Box, some wrong moves jam another part until they are reversed. Stamp Composition offers the opposite case. It provides a test surface where trials are cheap, while marks on the final surface are permanent. Taking such **consequential action** requires the agent to weigh the value of more information against its physical cost, and to judge when the evidence is strong enough to act. Every task ends with an explicit `SUBMIT`, so this judgment is part of every episode.

Long-horizon planning is an often essential in agentic setting. In Locked Storage task, for example, a sequence of actions need to thought in advance: finding a token $\rightarrow$ obtaining it without $\rightarrow$ placing at target location $\rightarrow$ opening a new storage $\rightarrow$ obtain new token without closing storage $\rightarrow \ldots$.

G TASK PROGRESS

Every episode ends at the first physical press of the `SUBMIT` button, or when the policy stops or exhausts its budget; the scene state at that moment is frozen and scored. Two quantities are reported per episode: *success* (the task predicate holds at the press) and *progress* $P \in [0, 1]$. P measures the state of the task's own objects, staged as the success predicate is staged; means such as opening a lid or carrying the lantern are never credited. A stage counts only in a passive state: the object released, supported and, where the predicate requires it, untouched by the robot. P is not capped, so $P = 1$ states that the goal configuration is reached; success additionally requires the task's tidiness clause, where one exists, and a valid press. One tidiness clause, the deduction $\delta = 0.1$, appears in three

tasks; every other side condition is recorded as a flag, not as a term. P is clipped to $[0, 1]$ after the deduction.

Two rules are shared by several tasks. *Three-stage object rule* (search room, locked storage, blackout search, unfamiliar containers): for each object i of n, the stage $s_i \in \{0, 1, 2, 3\}$ is 1 once its hiding place has been opened, 2 once the object has been out of its hiding place, 3 once it is placed at its destination, and

$$P = \frac{1}{n} \sum_{i=1}^{n} \frac{s_i}{3}. \tag{2}$$

Stages 1 and 2 are sticky: an object that has been out keeps $s_i \geq 2$ even if a closer shuts it back in or it falls to the floor. A cloche target reaches $s_i = 2$ as soon as the cloche is lifted; an open-counter target has no stage 1. *Chain rule* (puzzle box, locked storage): each lock or bolt of the chain is one share and the object's stages together are one share; once the object is out, the chain counts as fully released.

Search room. Equation equation 2 over the n targets, with stage 1 = the target's compartment opened to at least $15\,\%$ of its joint range (drawers 0.15 m, doors 0.24 rad), stage 2 = out of its hiding place (out of the compartment interior, out from under the cloche, or moved 0.15 m from its open spot), stage 3 = on the tray, supported and upright, less δ if a non-target object overlaps the tray at the press. Flags: target dropped to the floor, closer re-shut a target, objects not still.

Locked storage. Let d be the chain depth, u the number of chain links currently passable ($u = d$ once the target has been out; the dead-end compartment never counts), and s the target's stage as in search room. Then

$$P = \max\Big(0,\ \frac{u + s/3}{d + 1} - \delta\,[\text{a non-target object overlaps the tray}]\Big). \tag{3}$$

Flags: relocks, token on the floor, dead end opened.

Blackout search. As search room, including the deduction; the lantern contributes nothing. Flags: lantern lifted, carried, on the floor; lantern distance at each opening.

Marked mugs. For each of the n vessels let $v_i = 1$ if it stands upright on the pad of its label colour with its own two balls inside, $v_i = 0.5$ if it stands upright on that pad but the balls are not both its own, and $v_i = 0$ otherwise (a wrong pad is 0, not a penalty). Then $P = \frac{1}{n} \sum_i v_i$. Flags: vessels on a wrong pad; other objects released and still.

Painted cubes. With n cubes, c captured in their correct bin and w captured in the wrong bin (captures are irreversible),

$$P = \frac{c}{n}. \tag{4}$$

Flags: wrong captures w, coverage $(c + w)/n$.

Unfamiliar containers. Equation equation 2 over the n items, with stage 1 = the box holding the item opened once, stage 2 = the item out of its box, stage 3 = the item in the bowl. Flags: items released; the box mechanism type, reported against its opening rate.

Stamp composition. Let C be the set of target cells of the reference pattern and, for $c \in C$, let $g_c = 1$ if the cell holds only clean dots (each centre within 9 mm of the cell centre) and no more than the design's k_c dots, $g_c = 0.5$ if a dot crosses the cell edge or the cell holds more than k_c dots, and $g_c = 0$ if the cell is empty. Let m be the number of stray dots, i.e. dots on the paper outside C, plus one if the ink outside the grid exceeds 35 pixels. Then

$$P = \frac{\sum_{c \in C} g_c}{|C| + m}. \tag{5}$$

Dots are located from the recorded impressions with the stamp's die geometry. Flag: every stamp released, still and supported (required for success). An untouched scene reads $P = 0$.

Odd parcel. Let O be the set of odd parcels and B the set of parcels released inside the answer box. Then

$$P = \max\Big(0,\ \frac{|B \cap O|}{|B \cup O|} - \delta\,[\text{a parcel rests on either pan, or the balance was moved}]\Big). \tag{6}$$

An empty box reads 0; the exact odd set with parcels left on the pans reads 0.9.

Puzzle box. Let L be the chain length, r the number of chain bolts currently released (a relocked bolt does not count while the item is inside; $r = L$ once the item has been out; decoy sliders never count), and $s \in \{0, 1, 2, 3\}$ the item stage: 1 once the lid has slid past half its travel, 2 once the item has been out of the cavity, 3 once it stands in the tray, supported and released. Then

$$P = \frac{r + s/3}{L + 1}. \tag{7}$$

Flags: robot contact with any part at the press (required absent for success), relock events, the lock at which a failed episode stalled.

Wobbly stand. Let θ_0 be the initial tilt of the stand and θ its tilt at the press. Let $q = \mathrm{clip}(1 - \theta/\theta_0,\ 0,\ 1)$ while the stand is released, settled and untouched, $q = 1$ once θ is within the level threshold, and $q = 0$ while the robot holds or touches the stand (values below the deadband 0.01 read 0). Let $b = 1$ if the ball is on top of the stand, else 0. Then

$$P = q\,\frac{1 + b}{2}. \tag{8}$$

A level stand without its ball reads 0.5; an untouched scene reads 0. Flags: ball fell off, ball put back.

Task	Units	Stages or terms	Deduction δ
Search room	targets	opened / out / placed	non-target on tray
Locked storage	chain links, target	links + opened / out / placed	non-target on tray
Blackout search	targets	opened / out / placed	non-target on tray
Marked mugs	vessels	$\{0, 0.5, 1\}$ per vessel	–
Painted cubes	cubes	correct captures	–
Unfamiliar containers	items	opened / out / in bowl	–
Stamp composition	target cells	cell grades $\{0, 0.5, 1\}$, strays in the denominator	–
Odd parcel	parcel set	Jaccard of boxed set and odd set	pans loaded
Puzzle box	chain bolts, item	bolts + lid / out / tray	–
Wobbly stand	stand	tilt removed q, ball on top b	–

Table 8: Progress P at a glance. Every stage counts only in a passive state (released, supported, untouched where the predicate requires it); P is uncapped, frozen at the first Submit press and clipped to $[0, 1]$.

Table 9: **Episode outcomes per task.** Percentage of episodes by outcome, with absolute counts shown below in parentheses. Cell shading indicates the percentage within each model–task pair (darker indicates a larger percentage). C: correct submit. W: wrong submit. S: declared stop. A: text answer instead of a submit. T: timeout (out of decisions). Tasks are ordered as in Table 3.

		GPT-6 Astra					Claude Fable 5.1					Gemini 3.8 Flash				
Family	Task	C	W	S	A	T	C	W	S	A	T	C	W	S	A	T
Search	Locked Storage	28% (14)	44% (22)	20% (10)	0% (0)	8% (4)	8% (4)	42% (21)	36% (18)	14% (7)	0% (0)	0% (0)	20% (10)	2% (1)	0% (0)	78% (39)
	Search Room	0% (0)	56% (28)	8% (4)	0% (0)	36% (18)	0% (0)	56% (28)	34% (17)	8% (4)	2% (1)	2% (1)	32% (16)	0% (0)	0% (0)	66% (33)
	Blackout Search	0% (0)	76% (38)	12% (6)	0% (0)	12% (6)	0% (0)	46% (23)	50% (25)	2% (1)	2% (1)	0% (0)	56% (28)	6% (3)	0% (0)	38% (19)
Inspect	Painted Cubes	30% (15)	70% (35)	0% (0)	0% (0)	0% (0)	14% (7)	84% (42)	2% (1)	0% (0)	0% (0)	2% (1)	82% (41)	0% (0)	0% (0)	16% (8)
	Marked Mugs	10% (5)	82% (41)	4% (2)	0% (0)	4% (2)	0% (0)	80% (40)	20% (10)	0% (0)	0% (0)	0% (0)	88% (44)	0% (0)	0% (0)	12% (6)
	Unfamiliar Containers	6% (3)	66% (33)	6% (3)	0% (0)	22% (11)	0% (0)	24% (12)	68% (34)	8% (4)	0% (0)	0% (0)	36% (18)	0% (0)	0% (0)	64% (32)
Test	Puzzle Box	96% (48)	4% (2)	0% (0)	0% (0)	0% (0)	70% (35)	0% (0)	14% (7)	16% (8)	0% (0)	12% (6)	6% (3)	0% (0)	0% (0)	82% (41)
	Stamp Composition	14% (7)	84% (42)	2% (1)	0% (0)	0% (0)	2% (1)	74% (37)	22% (11)	2% (1)	0% (0)	0% (0)	54% (27)	0% (0)	0% (0)	46% (23)
	Wobbly Stand	10% (5)	86% (43)	2% (1)	0% (0)	2% (1)	6% (3)	62% (31)	30% (15)	0% (0)	2% (1)	2% (1)	96% (48)	0% (0)	0% (0)	2% (1)
	Odd Parcel	0% (0)	88% (44)	0% (0)	0% (0)	12% (6)	0% (0)	82% (41)	10% (5)	4% (2)	4% (2)	2% (1)	86% (43)	0% (0)	0% (0)	12% (6)
All tasks		19.4% (97)	65.6% (328)	5.4% (27)	0.0% (0)	9.6% (48)	10.0% (50)	55.0% (275)	28.6% (143)	5.4% (27)	1.0% (5)	2.0% (10)	55.6% (278)	0.8% (4)	0.0% (0)	41.6% (208)

H Demonstration Dataset Details

Table 10: **Demonstration dataset.** Scripted information gathering in each oracle's route, which the oracle performs although it knows the hidden information. Episodes: successful demonstrations. Hours: interaction time at 20 Hz.

Task	Scripted information gathering	Episodes	Hours
Locked Storage	follows the token chain	278	33.2
Search Room	opens compartments nearest-first until the targets are seen	294	46.8
Blackout Search	carries the lamp along a nearest-first search	201	53.0
Painted Cubes	removes covers and turns each cube face by face	276	50.3
Marked Mugs	lifts and tilts each vessel to read its label	312	20.9
Unfamiliar Containers	some episodes make one or two wrong tries before opening	398	76.9
Puzzle Box	tries sliders, one to three of which are blocked	447	21.1
Stamp Composition	prints test impressions before the final one	363	32.6
Wobbly Stand	releases the ball, places a shim, and tests again	381	18.9
Odd Parcel	weighs parcels on the balance	261	51.7
Total		3,211	405.5

I Workload and Occlusion Details

This appendix lists the numbers behind Sections K.1 and K.2. Table 11 gives each level and contrast with its interval, Table 12 breaks the occlusion variants down by task, and Tables 13 and 14 give every design cell. A cell holds five to eleven episodes, so single cells are coarse.

J Failure Attribution Details

Cohort. The attribution uses every instance that both `GPT-6 Astra` and `Claude Fable-5.1` played with an official score, after removing seed-zero runs, superseded registry entries, runs flagged as scene suspects, and runs interrupted by the provider or the runtime. This gives 540 instances per model, between 46 and 63 per task. It differs slightly from the 50 instances per

Table 11: **Factor estimates.** Task-balanced aggregate success rate and progress (%) per factor level, and contrasts between levels, with cell-stratified paired bootstrap 95% confidence intervals (4,000 resamples). Workload uses the six workload tasks; occlusion uses the four occlusion tasks; the shared-task rows recompute the workload contrasts on the four occlusion tasks only, and the last block gives the direct difference between the workload penalty (Low − High) and each occlusion penalty (Visible − Look, Visible − Uncover) on those tasks.

		GPT-6 Astra		Claude Fable 5.1	
Analysis	Quantity	SR	Prog	SR	Prog
Workload (6 tasks)	Low	23.1 [16.1, 30.3]	51.1 [44.9, 57.2]	9.0 [4.9, 13.2]	30.9 [25.3, 36.7]
	Mid	12.6 [7.5, 18.2]	45.7 [40.0, 51.1]	3.0 [0.0, 6.1]	27.0 [22.9, 31.1]
	High	7.7 [3.0, 12.6]	39.0 [33.9, 44.1]	0.0 [0.0, 0.0]	22.4 [18.9, 26.0]
	Mid − Low	-10.5 [-19.1, -1.7]	-5.3 [-13.5, 3.0]	-6.0 [-11.3, -0.8]	-3.9 [-10.8, 3.0]
	High − Low	-15.4 [-24.0, -6.8]	-12.1 [-20.1, -3.9]	-9.0 [-13.2, -4.9]	-8.5 [-15.4, -1.9]
Occlusion (4 tasks)	Visible	16.7 [9.7, 23.6]	51.1 [44.8, 57.5]	5.6 [2.8, 8.3]	34.2 [29.1, 39.3]
	Look	10.8 [4.7, 17.2]	45.6 [39.1, 51.8]	4.7 [0.0, 9.7]	27.6 [22.6, 32.8]
	Uncover	12.8 [6.4, 19.7]	44.5 [38.7, 50.6]	1.4 [0.0, 4.2]	29.5 [24.0, 34.8]
	Look − Visible	-5.8 [-15.0, 3.3]	-5.6 [-14.3, 3.5]	-0.8 [-5.6, 5.3]	-6.6 [-13.8, 0.7]
	Uncover − Visible	-3.9 [-13.9, 5.6]	-6.6 [-15.2, 1.8]	-4.2 [-8.3, 0.0]	-4.7 [-11.9, 2.7]
Workload, shared 4 tasks	Low	21.7 [13.6, 29.4]	53.0 [46.3, 59.8]	7.2 [2.8, 11.7]	31.3 [25.3, 37.4]
	Mid	12.8 [6.9, 18.9]	46.8 [41.0, 52.8]	4.4 [0.0, 9.2]	33.0 [27.6, 38.3]
	High	5.8 [1.4, 11.4]	41.4 [35.4, 47.1]	0.0 [0.0, 0.0]	27.1 [22.8, 31.4]
	Mid − Low	-8.9 [-19.2, 1.4]	-6.2 [-15.1, 2.6]	-2.8 [-8.9, 3.6]	1.7 [-6.4, 9.7]
	High − Low	-15.8 [-25.6, -6.7]	-11.6 [-20.5, -2.8]	-7.2 [-11.7, -2.8]	-4.2 [-11.7, 3.3]
Difference of penalties	Workload − Look penalty	10.0 [-3.1, 23.1]	6.1 [-6.0, 18.9]	6.4 [1.4, 12.5]	-2.4 [-12.0, 7.6]
	Workload − Uncover penalty	11.9 [-0.6, 24.2]	5.0 [-6.9, 16.9]	3.1 [0.0, 7.5]	-0.5 [-10.1, 9.3]

Table 12: **Occlusion by task.** Visible / Look / Uncover on the four tasks that share the occlusion design. Visible means no additional object search or cover removal is required; the deciding information (die faces, parcel mass, bottom labels, cube faces) is still hidden in every mode. Each mode is the mean of its workload cells. SR and Prog in %, Dec: mean decisions, T/O: timeout share (%).

Task	Policy	SR V / L / U	Prog V / L / U	Dec V / L / U	T/O V / L / U
Stamp Composition	GPT-6 Astra	6 / 31 / 7	62 / 63 / 39	110 / 86 / 119	0 / 0 / 0
	Claude Fable 5.1	0 / 6 / 0	39 / 36 / 38	143 / 139 / 143	0 / 0 / 0
Odd Parcel	GPT-6 Astra	0 / 0 / 0	14 / 20 / 14	174 / 186 / 189	13 / 11 / 21
	Claude Fable 5.1	0 / 0 / 0	9 / 4 / 7	181 / 180 / 188	7 / 0 / 5
Marked Mugs	GPT-6 Astra	17 / 0 / 13	47 / 36 / 46	132 / 121 / 126	11 / 0 / 0
	Claude Fable 5.1	0 / 0 / 0	21 / 15 / 15	133 / 139 / 153	0 / 0 / 0
Painted Cubes	GPT-6 Astra	44 / 12 / 31	81 / 63 / 79	114 / 122 / 130	0 / 0 / 0
	Claude Fable 5.1	22 / 13 / 6	68 / 56 / 59	133 / 136 / 143	0 / 0 / 0
Task-balanced mean	GPT-6 Astra	16.7 / 10.8 / 12.8	51.1 / 45.6 / 44.5	133 / 129 / 141	6.1 / 2.9 / 5.4
	Claude Fable 5.1	5.6 / 4.7 / 1.4	34.2 / 27.6 / 29.5	148 / 149 / 157	1.7 / 0.0 / 1.2

task of Table 3, so rates here are not directly comparable to that table. For Stamp Composition we use the dot rule of §G for success. Five `GPT-6 Astra` Odd Parcel episodes and one `Claude Fable-5.1` episode boxed exactly the odd set but left a parcel on a pan. The instruction does not ask for empty pans, so we report these separately and leave them out of the breakdown.

Signals from the action log. Every observation reports the gripper pose and the two finger joints. A close command after which the fingers stay at least 6 mm apart is a grasp, and one after which they close to under 4 mm is a miss. A grip is lost when the fingers collapse below 4 mm while the close command still holds, and an open command while holding is a release. A hold is a *carry* when the gripper lifts by more than 6 cm and moves by more than 15 cm, which separates carrying an object from pulling a handle. A repair attempt is any later grasp or miss within 30 cm of the point where a carried object was lost. We follow objects through carries by matching each grasp to the nearest tracked object within 8 to 11 cm. Thresholds were set on a few episodes and then checked on video frames.

Table 13: **Workload × occlusion cells.** Success rate / progress (%) per design cell for the four tasks that cross both factors, with the number of episodes per cell in parentheses (the same instances for both models). Rows: workload level; columns: occlusion variant.

		GPT-6 Astra			Claude Fable 5.1		
Task	Workload	Visible	Look	Uncover	Visible	Look	Uncover
Stamp Composition	Low (2)	17 / 74 (6)	60 / 84 (5)	20 / 44 (5)	0 / 23 (6)	0 / 20 (5)	0 / 47 (5)
	Mid (3)	0 / 47 (5)	17 / 71 (6)	0 / 50 (6)	0 / 72 (5)	17 / 44 (6)	0 / 19 (6)
	High (4)	0 / 67 (6)	17 / 35 (6)	0 / 22 (5)	0 / 24 (6)	0 / 43 (6)	0 / 47 (5)
Odd Parcel	Low (4)	0 / 26 (7)	0 / 26 (7)	0 / 0 (2)	0 / 13 (7)	0 / 0 (7)	0 / 0 (2)
	Mid (6)	0 / 8 (5)	0 / 24 (5)	0 / 30 (7)	0 / 8 (5)	0 / 8 (5)	0 / 9 (7)
	High (8)	0 / 9 (5)	0 / 9 (5)	0 / 13 (7)	0 / 5 (5)	0 / 5 (5)	0 / 11 (7)
Marked Mugs	Low (3)	17 / 53 (6)	0 / 37 (5)	0 / 50 (5)	0 / 28 (6)	0 / 27 (5)	0 / 30 (5)
	Mid (4)	17 / 42 (6)	0 / 31 (6)	20 / 42 (5)	0 / 27 (6)	0 / 15 (6)	0 / 10 (5)
	High (5)	17 / 47 (6)	0 / 40 (6)	20 / 46 (5)	0 / 7 (6)	0 / 3 (6)	0 / 4 (5)
Painted Cubes	Low (4)	67 / 88 (6)	20 / 65 (5)	60 / 90 (5)	67 / 83 (6)	20 / 50 (5)	0 / 55 (5)
	Mid (6)	67 / 92 (6)	0 / 50 (5)	33 / 75 (6)	0 / 67 (6)	20 / 57 (5)	17 / 61 (6)
	High (8)	0 / 62 (5)	17 / 75 (6)	0 / 71 (6)	0 / 55 (5)	0 / 60 (6)	0 / 60 (6)

Table 14: **Workload cells of the two remaining workload tasks.** Success rate / progress (%) per design cell with episode counts in parentheses. Unfamiliar Containers crosses item count with visible / look; Locked Storage crosses chain depth with the presence of a dead-end link.

		GPT-6 Astra		Claude Fable 5.1	
Task	Workload	Visible	Look	Visible	Look
Unfamiliar Containers	Low (2)	20 / 42 (10)	9 / 39 (11)	0 / 17 (10)	0 / 18 (11)
	Mid (3)	0 / 36 (8)	0 / 21 (7)	0 / 26 (8)	0 / 13 (7)
	High (4)	0 / 27 (7)	0 / 19 (7)	0 / 15 (7)	0 / 11 (7)
Task	Workload	No dead end	Dead end	No dead end	Dead end
Locked Storage	Low (1)	38 / 55 (8)	38 / 53 (8)	38 / 38 (8)	12 / 48 (8)
	Mid (2)	38 / 56 (8)	11 / 62 (9)	0 / 8 (8)	0 / 12 (9)
	High (3)	33 / 58 (9)	12 / 32 (8)	0 / 19 (9)	0 / 6 (8)

Rules. Each unmet requirement gets exactly one category. When a unit fails in execution and a response can be observed, it is attributed to the response: no repair attempt gives failure to detect errors, and a failed repair attempt gives ineffective error recovery. A requirement never attempted in an episode that ended by `SUBMIT`, `Stop`, or a text answer is incorrect termination. In an episode that ran out of decisions it is *out of decisions*. Table 15 lists the task-specific rules.

Validation and limits. The earlier review of the failed episodes' videos coded, for each episode, whether an observed execution failure caused the error and whether a mark was misread. On the first question our attribution agrees on 195 of 217 episodes where the review reached a verdict ($\kappa = 0.79$). On the second it agrees on 43 of 50 episodes ($\kappa = 0.63$); most of the disagreements are targets that the gripper reached but did not collect, which we count as misinterpretation and the review left open. The rules have known biases. Inadequate evidence gathering in the Search family is an upper bound, since a target seen from afar and never approached also counts. Misinterpretation in Painted Cubes is a lower bound, since a missed face explains some errors that were in fact misreads. The Odd Parcel split between IEG and FRP rests on the number of pan placements, not on which parcels were compared. No rule uses the agent's own text, because `GPT-6 Astra` returns no readable reasoning and text would favour the model that writes more. The code and the episode-level attributions are released with the benchmark.

Table 15: **Attribution rules per task.** Unit: what counts as one unmet requirement. The rules are applied in the order listed, and the first matching rule assigns the category. IEG: inadequate evidence gathering; MIS: misinterpretation of evidence; FRP: faulty reasoning or planning; FAE: failed action execution; FDE/IER: after an execution failure, no repair attempt or a failed one; IT: incorrect termination.

Task and unit	Rules
Search Room, Blackout Search; each missing target	On the tray but not upright: FAE. Carried and lost: FDE/IER. Hiding place never within 25 cm: IEG. Three or more grasp attempts at the target without carrying it: IER. Gripper within 15 cm of the target: MIS. Otherwise IEG. Last, a look-alike of the same colour or kind on the tray turns an IEG or MIS unit into MIS; an unpaired non-target on the tray is its own MIS unit.
Locked Storage; the target	Chain progress is read off the progress score. If the target is reachable, as in Search Room. Otherwise, for the token of the first closed link: never approached, IEG; carried and lost, FDE/IER; placed on its reader but the link is closed at the press, FRP.
Unfamiliar Containers; each item not in the bowl	Box never opened: IEG if never approached or probed fewer than three times, FRP if grasped three or more times. Box opened, item still inside: IER after two or more grasp attempts, else IT. Item out of its box: FDE/IER.
Painted Cubes; each misfiled or uncaptured cube	Fell into a bin when the grip was lost: FAE. Wrong bin chosen: IEG if some set of unseen faces explains the choice, else MIS. Hidden and never picked: IEG. In view and never picked, or set down and never binned: IT. Carried and lost: FDE/IER.
Marked Mugs; each failed vessel	Wrong pad: MIS if the vessel was tilted before placement, else IEG. Correct pad with another vessel's balls: FRP. Correct pad with own balls missing: IER if balls were picked up again, else FDE. Tipped over, or set down at its pad without resting on it: FAE. Never moved: IT.
Stamp Composition; each wrong cell	Ink outside the grid, or dots across cell edges: FAE. Blank target cells with fewer final impressions than the solution needs: IT. Other wrong cells: IEG if a stamp reached the final board untested, else MIS.
Odd Parcel; each missing or extra parcel	Extra parcel in the box: FRP. Missing odd parcel: IEG if the agent made fewer pan placements than there are parcels, else FRP.
Puzzle Box; the stall	First unreleased part in the release order: blocked at the press, FRP; never grasped, IEG; grasped three or more times, FRP; else FAE. Lid never opened after all bolts: IEG if never grasped, else FAE. Item out but not in the tray: FAE.
Wobbly Stand; the stand	Tilt above 15°, shim beside a leg, or level without the ball: FAE. Shim under a leg that is not short: MIS. Short legs partly shimmed or wrong thickness: FRP. No shim placed: IT.

K A Detailed Analysis on Exploration Impairment

§3 argues that RoboQuest tasks demand four groups of capabilities (Table 2): evidence acquisition, evidence use, interactive inference, and action organization, and that a weakness in any of them may lead to the failures in our taxonomy (§3.4). §4 showed that the frontier models fail most episodes. In this section, we connect the two. We first assign every failure of `GPT-6 Astra` and `Claude Fable-5.1` to a taxonomy category, and then read the result one capability group at a time.

Attributing failures. A failed episode can fail in several ways at once. A Search Room episode with three targets may end with two of them missing, and a Painted Cubes episode with eight cubes may misfile three. We therefore count failures per *unmet requirement* rather than per episode: every target left out of the tray, every misfiled or unsorted cube, every vessel not correctly placed, every wrong stamp cell, every missing or extra parcel, and, for Puzzle Box and Wobbly Stand, the single unsolved chain or stand (Table 15). We assign each unmet requirement to the first step of the workflow that failed and was not corrected. The assignment uses the final simulator state, the hidden task specification, and the gripper trajectory recorded at every decision. For example, a target whose hiding place the gripper never approached is an inadequate evidence gathering failure, while a target that was picked up and then dropped is a failed action execution. The rules are the same for both models (Appendix J), and on whether an execution error caused the failure, they agree with a manual

Table 16: **Failure breakdown per task.** Share (%) of unmet requirements in each category. Ep.: failed episodes; Req.: unmet requirements. IEG: inadequate evidence gathering; MIS: misinterpretation of evidence; FRP: faulty reasoning or planning; FAE: failed action execution; FDE: failure to detect errors; IER: ineffective error recovery; IT: incorrect termination; OOD: out of decisions.

Task	Policy	Ep.	Req.	IEG	MIS	FRP	FAE	FDE	IER	IT	OOD
Search Room	`GPT-6 Astra`	63	150	59	15	0	6	13	8	0	0
	`Claude Fable-5.1`	63	160	68	16	0	6	5	4	0	0
Locked Storage	`GPT-6 Astra`	43	45	24	29	4	0	22	20	0	0
	`Claude Fable-5.1`	55	58	24	40	2	0	9	26	0	0
Blackout Search	`GPT-6 Astra`	55	131	50	15	0	11	13	11	0	0
	`Claude Fable-5.1`	55	129	65	17	0	5	7	6	0	0
Painted Cubes	`GPT-6 Astra`	37	85	58	15	0	4	1	0	22	0
	`Claude Fable-5.1`	47	132	61	7	0	0	1	2	30	0
Marked Mugs	`GPT-6 Astra`	49	128	2	0	0	70	2	4	22	1
	`Claude Fable-5.1`	54	187	4	1	0	42	1	3	49	1
Unfamiliar Containers	`GPT-6 Astra`	51	118	66	0	14	1	6	4	6	3
	`Claude Fable-5.1`	54	138	72	0	14	1	3	4	6	0
Stamp Composition	`GPT-6 Astra`	42	123	5	27	0	31	0	0	37	0
	`Claude Fable-5.1`	51	170	15	11	0	24	0	0	49	0
Odd Parcel	`GPT-6 Astra`	46	75	41	0	59	0	0	0	0	0
	`Claude Fable-5.1`	46	94	60	0	40	0	0	0	0	0
Puzzle Box	`GPT-6 Astra`	2	2	0	0	0	100	0	0	0	0
	`Claude Fable-5.1`	15	15	0	0	53	47	0	0	0	0
Wobbly Stand	`GPT-6 Astra`	44	44	0	9	11	75	0	0	5	0
	`Claude Fable-5.1`	46	46	0	2	11	67	0	0	20	0

Table 17: **Failure breakdown by taxonomy category,** in % of the unmet requirements of failed episodes. Overall: 95% bootstrap intervals. §J gives the attribution rules.

			Search		Inspect		Test		Overall	
Phase	Category	Linked capabilities	Astra	Fable	Astra	Fable	Astra	Fable	Astra	Fable
Before action	Inadequate evidence gathering	Acquisition, inference	50.6	59.7	39.0	40.9	15.2	25.2	36.7 [34, 40]	42.2 [39, 45]
	Misinterpretation of evidence	Use	16.6	20.5	3.9	2.4	15.2	6.2	11.5 [9, 14]	9.0 [7, 11]
	Faulty reasoning or planning	Use, inference, organization	0.6	0.3	4.8	4.4	20.1	15.7	7.4 [6, 9]	6.4 [5, 8]
During action	Failed action execution	Organization	7.4	4.6	28.1	17.3	29.9	24.3	21.1 [19, 23]	15.4 [13, 17]
After feedback	Failure to detect errors	Use, organization	14.1	6.3	3.3	1.3	0.0	0.0	6.3 [5, 8]	2.5 [2, 3]
	Ineffective error recovery	Inference, organization	10.7	8.6	3.0	3.1	0.0	0.0	5.0 [4, 6]	3.9 [3, 5]
	Incorrect termination	Use, organization	0.0	0.0	16.3	30.4	19.7	28.6	11.3 [8, 15]	20.5 [17, 24]
Failed episodes			161	173	137	155	134	158	432	486
Unmet requirements (denominator)			326	347	331	457	244	325	901	1129

review of the episode videos on 90% of the episodes. Table 17 summarizes the result by task family, with the capability groups linked to each failure type, and Fig. 5 breaks it down per task.

More than half of all failures occur before the agent acts (56% for `GPT-6 Astra` and 58% for `Claude Fable-5.1`), about a fifth during action (21% and 15%), and the rest after feedback (23% and 27%). The two models therefore share the same main weakness, but they differ after that. `GPT-6 Astra` fails more often in execution and in reacting to its own errors, while `Claude Fable-5.1` fails more often by stopping early. We now discuss each capability group.

K.1 Evidence Acquisition

Directed search and **active perception** decide whether the agent obtains the evidence it needs. Their failure, inadequate evidence gathering, is the largest single category for both models (37%

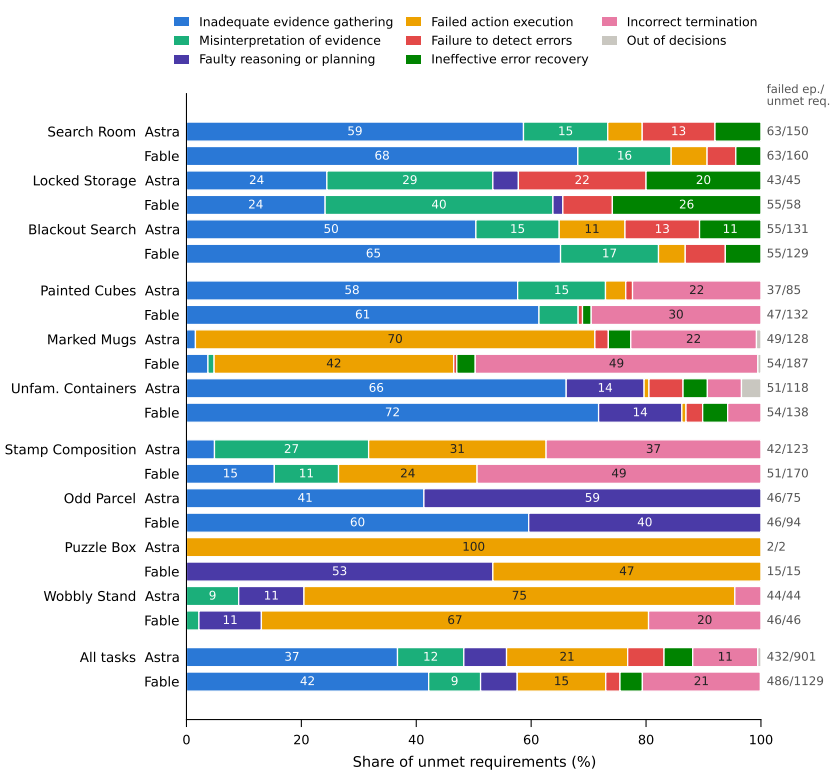

Figure 5: **Failure breakdown per task.** Each bar splits one policy's unmet requirements into the taxonomy categories. Numbers give failed episodes / unmet requirements.

and 42%). In Search Room and Blackout Search, the gripper never came within 25 cm of the hiding place of 83% (`GPT-6 Astra`) and 91% (`Claude Fable-5.1`) of these missing targets. In Unfamiliar Containers, more than half of the missing items stayed in boxes that the robot never approached. Hence, the agents rarely miss a target they have found. Instead, they stop searching while relevant places remain unchecked.

Occlusion. Occlusion adds a search or uncovering step before the task. In *Visible*, the relevant objects are in view; in *Look*, the agent must first find them from another viewpoint; and in *Uncover*, it must first physically uncover them. Four tasks vary it: Stamp Composition, Odd Parcel, Marked Mugs and Painted Cubes. Compared with Visible, Look and Uncover lower success by only 3.3 and 4.0 points on average over the two models, and neither interval excludes zero, while progress falls by 6.1 and 5.6 points (Table 18). Occlusion mainly moves failures into evidence acquisition: Look and Uncover add 0.15 and 0.23 failures before action per episode, and Look removes 0.17 failures during action (Table 19). An object that is never found is never handled, so its failure appears as missing evidence rather than as a failed action.

Painted Cubes. Painted Cubes isolates active perception, since the deciding marks can be on any face of a cube. The agent must put each cube that satisfies a rule into the blue bin and every other cube into the yellow bin. A qualifying cube in the yellow bin is a false negative (FN), and a non-qualifying cube in the blue bin is a false positive (FP). `GPT-6 Astra` makes more false negatives than false positives, with a false-negative rate (FNR) of 21.7% against a false-positive rate (FPR) of 9.8%, and `Claude Fable-5.1` does so far more, 46.1% against 6.5% (Fig. 6). The gap between the two rates is 27.6 points larger for `Claude Fable-5.1` (95% CI $[10.9, 43.4]$). This direction points to incomplete inspection. When the rule asks for a mark, an agent that skips a face can wrongly reject a qualifying cube, but it can wrongly accept a cube only by misreading a mark. Unseen faces can explain 39 of the 48 wrong bin choices of `Claude Fable-5.1`, against 22 of 35 for `GPT-6 Astra`. `Claude Fable-5.1` also left 42 hidden cubes unfound and 27

Table 18: **Workload and occlusion** on the four tasks that vary both, pooled over both models. Brackets: 95% bootstrap intervals. Last block: workload drop minus occlusion drop.

	SR			Prog		
	Astra	Fable	Pooled	Astra	Fable	Pooled
Workload						
Low	21.7	7.2	14.4	53.0	31.3	42.1
Mid	12.8	4.4	8.6	46.8	33.0	39.9
High	5.8	0.0	2.9	41.4	27.1	34.2
High − Low	−15.8 [−25.6, −6.7]	−7.2 [−11.7, −2.8]	−11.5 [−16.8, −6.2]	−11.6 [−20.5, −2.8]	−4.2 [−11.7, 3.3]	−7.9 [−14.0, −1.8]
Occlusion						
Visible	16.7	5.6	11.1	51.1	34.2	42.7
Look	10.8	4.7	7.8	45.6	27.6	36.6
Uncover	12.8	1.4	7.1	44.5	29.5	37.0
Look − Visible	−5.8 [−15.0, 3.3]	−0.8 [−5.6, 5.3]	−3.3 [−8.8, 1.9]	−5.6 [−14.3, 3.5]	−6.6 [−13.8, 0.7]	−6.1 [−11.6, −0.4]
Uncover − Visible	−3.9 [−13.9, 5.6]	−4.2 [−8.3, 0.0]	−4.0 [−9.3, 1.2]	−6.6 [−15.2, 1.8]	−4.7 [−11.9, 2.7]	−5.6 [−11.6, 0.4]
Workload drop minus occlusion drop						
vs. Look	10.0 [−3.1, 23.1]	6.4 [1.4, 12.5]	8.2 [1.5, 15.0]	6.1 [−6.0, 18.9]	−2.4 [−12.0, 7.6]	1.8 [−6.3, 9.8]
vs. Uncover	11.9 [−0.6, 24.2]	3.1 [0.0, 7.5]	7.5 [1.1, 13.8]	5.0 [−6.9, 16.9]	−0.5 [−10.1, 9.3]	2.3 [−5.9, 10.5]

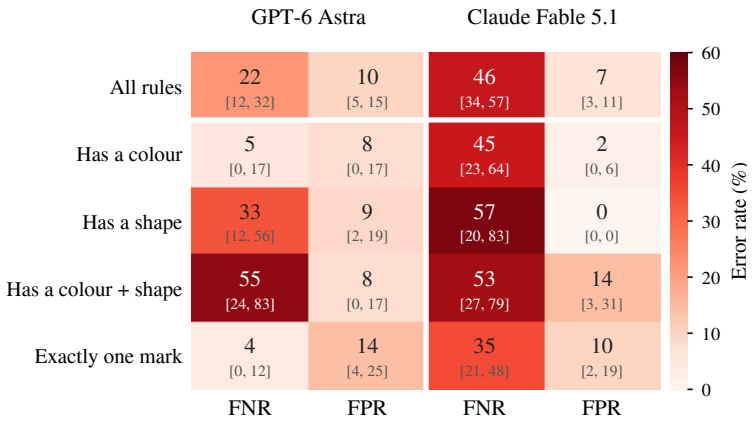

Figure 6: **Painted Cubes: error rates by rule.** False-negative rate (FNR) and false-positive rate (FPR) over committed cubes, in %, with 95% bootstrap intervals over episodes in brackets.

visible cubes untouched, against 27 and 13 for `GPT-6 Astra`. Thus, `Claude Fable-5.1` stops inspecting too early, both on each cube and across the scene.

K.2 Evidence Use

Once the evidence is in view, **evidence integration** and **memory** decide whether it is used correctly. Misinterpretation of evidence accounts for 11.5% and 9.0% of the failures. It is concentrated in tasks with look-alike objects. In Locked Storage, it accounts for 29% and 40% of the failures, mostly a similar object placed on the tray instead of the target. Faulty reasoning is most common in the Test family (20% and 16%). In Odd Parcel, it accounts for 59% and 40% of the failures: the agents placed parcels on the balance at least as many times as there are parcels, yet boxed a wrong or incomplete set, or nothing. The balance results were available, but they were not combined into a conclusion.

Workload. Workload is the number of units a task requires: 2, 3 or 4 stamps, 4, 6 or 8 parcels, 3, 4 or 5 vessels, and 4, 6 or 8 cubes (Low, Mid and High). More units mean more evidence to integrate and remember, and a longer plan. From Low to High workload, success falls by 15.8

Table 19: **Where the extra failures come from.** Unmet requirements per episode by failure phase, on the tasks and instances of Table 18. Brackets: 95% bootstrap intervals.

	Before action	During action	After feedback	All
Workload				
Low	0.66	0.48	0.49	1.63
Mid	1.01	0.61	0.72	2.36
High	1.42	0.67	1.10	3.19
High − Low	0.75 [0.53, 0.96]	0.19 [0.05, 0.33]	0.61 [0.29, 0.95]	1.55 [1.23, 1.89]
Occlusion				
Visible	0.91	0.65	0.64	2.22
Look	1.06	0.49	0.90	2.44
Uncover	1.15	0.61	0.77	2.53
Look − Visible	0.15 [−0.08, 0.36]	−0.17 [−0.30, −0.03]	0.25 [−0.05, 0.55]	0.22 [−0.11, 0.61]
Uncover − Visible	0.23 [−0.01, 0.46]	−0.04 [−0.17, 0.09]	0.13 [−0.16, 0.48]	0.31 [−0.07, 0.68]

points for `GPT-6 Astra` (21.7% to 5.8%) and by 7.2 points for `Claude Fable-5.1` (7.2% to 0.0%), with both intervals excluding zero (Table 18). The decline is not caused by running out of decisions, since timeouts stay below 6% at every level. Adding Unfamiliar Containers and Locked Storage, which also vary workload, gives the same trend: over all six tasks, success falls from Low to High by 15.4 points for `GPT-6 Astra` and 9.0 points for `Claude Fable-5.1` (Table 22). The workload drop exceeds the occlusion drops by 8.2 and 7.5 points, as success requires every unit to be right. The taxonomy shows where the extra failures arise (Table 19). From Low to High, the number of unmet requirements per episode rises by 1.55. Most of the rise occurs before action (0.75) and at termination (0.61), whereas failed execution rises by only 0.19. Hence, the additional units are not lost because they are harder to handle. They are lost because the agent does not inspect, test or attempt them before it stops, which points to limits of evidence integration, memory and **long-horizon planning**.

K.3 Interactive Inference

Affordance discovery, **experimental identification** and **physical causal inference** require the agent to learn from the outcome of its own actions. In Unfamiliar Containers, 14% of the failures of both models are boxes that the robot grasped three or more times without opening them, repeating a move that brought no progress. In Wobbly Stand, on the other hand, the ball test mostly works: a shim under the wrong leg, which would indicate a misread roll, occurs only 4 times and once.

Puzzle Box. Puzzle Box isolates physical causal inference. A chain of sliding bolts locks the lid, and each bolt moves only after the part blocking it is out of the way. A partial or full opaque cover hides some or all bolt engagements, so the agent must infer the blocking order from how the parts move. Covered and uncovered episodes are matched in chain length. Without a cover, `GPT-6 Astra` and `Claude Fable-5.1` are equally strong, solving 94.4% and 100% of the episodes (Table 20). Under the cover, `GPT-6 Astra` still solves 96.4%, although its episodes grow from 85 to 120 decisions. `Claude Fable-5.1`, however, drops to 34.6%, and its episodes grow from 107 to 175 decisions. The two cover effects differ by 67.3 points (95% CI $[44.9, 89.8]$). `Gemini-3.8 Flash` solves 27.8% without a cover and none with it, running out of decisions in 95% of the covered episodes. None of the 15 failures of `Claude Fable-5.1` is inadequate evidence gathering, since it always reaches the bolt that has to move next. In 8 episodes, it pulls that bolt three or more times without releasing it, which is faulty reasoning. In the other 7, it grasps the right bolt but does not slide it far enough. Thus, `Claude Fable-5.1` finds the right part but fails to infer how it moves, while `GPT-6 Astra` recovers the hidden structure through interaction.

Table 20: **Puzzle Box: effect of hiding the locks,** on chain lengths 4 and 5. Δ: Covered minus No cover, with 95% bootstrap interval. Last row: `Claude Fable-5.1`'s Δ minus `GPT-6 Astra`'s.

Policy	Cover	SR	Prog	Decisions
`GPT-6 Astra`	No cover	94.4	99.6	84.7
	Covered	96.4	99.5	119.5
	Δ	2.0 [−8.7, 16.7]	−0.1 [−1.4, 1.0]	34.8 [14.0, 55.8]
`Claude Fable-5.1`	No cover	100.0	100.0	106.7
	Covered	34.6	59.1	174.6
	Δ	−65.4 [−83.7, −46.2]	−40.9 [−56.3, −26.0]	67.9 [46.8, 86.1]
`Gemini-3.8 Flash`	No cover	27.8	45.2	172.2
	Covered	0.0	25.5	199.8
	Δ	−27.8 [−44.4, −11.1]	−19.7 [−37.9, −1.0]	27.6 [8.5, 48.8]
Difference in Δ		−67.3 [−89.8, −44.9]	−40.8 [−56.2, −26.1]	33.1 [5.0, 63.6]

Table 21: **How episodes end,** in % of 500 episodes per model. Stop, Text answer, and Timeout end without a `SUBMIT` and all fail. Precision: share of submissions that were correct.

	Submitted		No submit				
Model	Correct	Wrong	Stopped	Text answer	Timeout	Submit rate	Precision
GPT-6 Astra	19.4	65.6	5.4	0.0	9.6	85.0	22.8
Claude Fable 5.1	10.0	55.0	28.6	5.4	1.0	65.0	15.4
Gemini 3.8 Flash	2.0	55.6	0.8	0.0	41.6	57.6	3.5

K.4 Action Organization

Even with the right evidence and inference, the agent must carry out **consequential actions** and respond to their outcomes. Failed action execution accounts for 21% of the failures of `GPT-6 Astra` and 15% of those of `Claude Fable-5.1`, mostly in Marked Mugs, Wobbly Stand and Stamp Composition. In Marked Mugs, only 2 and 9 vessels ended on a wrong pad, so the labels were mostly read correctly. The failures come later: vessels set down at their pad without resting on it (71 and 44) or tipped over (18 and 32). In Wobbly Stand, the stand was knocked over in 13 and 14 episodes, and the shim ended beside the leg rather than under it in 13 and 14 more. In Stamp Composition, 30 and 35 episodes left ink outside the final grid. In these tasks, the agents gather the right evidence but fail to act on it precisely.

Responding to feedback. The two models differ most in how they respond to the results of their actions. After a carried object was lost, `GPT-6 Astra` never went back for it in 57% of the cases (54 of 94), against 42% (27 of 65) for `Claude Fable-5.1`, so failure to detect errors is more frequent for `GPT-6 Astra` (6.3% against 2.5%). `Claude Fable-5.1` instead fails more often through incorrect termination: 20.5% of its failures, against 11.3% for `GPT-6 Astra`, a gap of 9.2 points (95% CI $[4, 14]$). These are requirements it never attempted before stopping or submitting. For example, it left 81 vessels untouched in Marked Mugs, in line with its high stop rate in Table 21.

In summary, both models fail mostly at evidence acquisition. Beyond that, `GPT-6 Astra` acts more and breaks more, and often leaves its own errors uncorrected, whereas `Claude Fable-5.1` gathers less evidence and ends more episodes with work left undone.

L Additional Analysis Tables

Table 22: **Workload by task.** Each cell gives Low / Mid / High. SR, Prog, and T/O (share of episodes that ran out of decisions) are in %; Dec is the mean number of decisions per episode. Workload levels are listed in Sections K.1 and K.2. Each level has about 17 episodes per model, so single-task trends are noisy.

Task	Policy	SR L / M / H	Prog L / M / H	Dec L / M / H	T/O L / M / H
Stamp Composition	GPT-6 Astra	32 / 6 / 6	67 / 56 / 41	104 / 113 / 98	0 / 0 / 0
	Claude Fable 5.1	0 / 6 / 0	30 / 45 / 38	138 / 142 / 146	0 / 0 / 0
Odd Parcel	GPT-6 Astra	0 / 0 / 0	17 / 21 / 11	171 / 186 / 192	21 / 7 / 18
	Claude Fable 5.1	0 / 0 / 0	4 / 8 / 7	179 / 180 / 190	0 / 0 / 11
Marked Mugs	GPT-6 Astra	6 / 12 / 12	46 / 38 / 44	111 / 132 / 136	0 / 6 / 6
	Claude Fable 5.1	0 / 0 / 0	28 / 17 / 5	136 / 150 / 139	0 / 0 / 0
Painted Cubes	GPT-6 Astra	49 / 33 / 6	81 / 72 / 69	95 / 129 / 142	0 / 0 / 0
	Claude Fable 5.1	29 / 12 / 0	63 / 61 / 59	121 / 133 / 158	0 / 0 / 0
Unfamiliar Containers	GPT-6 Astra	15 / 0 / 0	41 / 28 / 23	183 / 189 / 151	10 / 41 / 21
	Claude Fable 5.1	0 / 0 / 0	17 / 20 / 13	170 / 181 / 150	0 / 0 / 0
Locked Storage	GPT-6 Astra	38 / 24 / 23	54 / 59 / 45	140 / 162 / 149	6 / 11 / 6
	Claude Fable 5.1	25 / 0 / 0	43 / 10 / 13	113 / 135 / 119	0 / 0 / 0
Task-balanced mean	GPT-6 Astra	23.1 / 12.6 / 7.7	51.1 / 45.7 / 39.0	134 / 152 / 145	6.2 / 10.7 / 8.4
	Claude Fable 5.1	9.0 / 3.0 / 0.0	30.9 / 27.0 / 22.4	143 / 153 / 150	0.0 / 0.0 / 1.9

Table 23: **Puzzle Box: how far episodes get.** Share of episodes (%) that reach each stage, on chain lengths 4 and 5, weighted as in Table 20. All bolts: every chain bolt was released at some point. Extracted: the item left the box. Gave up: the episode ended with a declared `Stop` or a text answer. Timeout: the episode ran out of decisions.

Policy	Cover	All bolts	Extracted	Solved	Gave up	Timeout
`GPT-6 Astra`	No cover	100	100	94	0	0
	Covered	100	96	96	0	0
`Claude Fable-5.1`	No cover	100	100	100	0	0
	Covered	47	40	35	65	0
`Gemini-3.8 Flash`	No cover	39	39	28	0	67
	Covered	13	0	0	0	95

Table 24: **Painted Cubes: direction of wrong-bin commitments.** Cube counts by final outcome. A positive cube satisfies the rule and belongs in the blue bin. FN: positive in yellow; FP: negative in blue. FNR = FN/(TP+FN) and FPR = FP/(TN+FP) over committed cubes, in %, with 95% bootstrap intervals over episodes. Uncaptured cubes (positive / negative) are counted separately.

Policy	Rule	TP	FN	FNR	TN	FP	FPR	Uncaptured
`GPT-6 Astra`	Has a colour	21	1	4.5 [0.0, 16.7]	49	4	7.5 [0.0, 17.0]	7 / 4
	Has a shape	10	5	33.3 [12.5, 55.6]	30	3	9.1 [2.5, 19.4]	2 / 2
	Has a colour + shape	9	11	55.0 [23.8, 82.6]	36	3	7.7 [0.0, 17.1]	2 / 7
	Exactly one mark	25	1	3.8 [0.0, 12.0]	42	7	14.3 [4.3, 25.0]	7 / 14
	All rules	65	18	21.7 [11.9, 31.7]	157	17	9.8 [5.5, 14.7]	18 / 27
`Claude Fable-5.1`	Has a colour	11	9	45.0 [23.1, 64.3]	50	1	2.0 [0.0, 5.8]	9 / 6
	Has a shape	6	8	57.1 [20.0, 83.3]	26	0	0.0 [0.0, 0.0]	3 / 9
	Has a colour + shape	9	10	52.6 [27.3, 78.6]	30	5	14.3 [2.7, 30.6]	3 / 11
	Exactly one mark	15	8	34.8 [21.1, 48.0]	37	4	9.8 [2.4, 18.9]	10 / 22
	All rules	41	35	46.1 [34.2, 57.3]	143	10	6.5 [2.9, 10.7]	25 / 48