PERCEPTARENA: BENCHMARKING PERCEPTION AS A DECISION IN WEB AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Computer-use benchmarks have proven sharp instruments for measuring an agent's execution ability: given an observation, can it select the right actions? Yet nearly all of them share a single harness design: the environment is rendered and fed to the agent at every step, so the agent itself never decides what to observe. We introduce **PerceptArena**, which returns that control over perception to the agent itself: how much to observe, at what fidelity, and when to stop observing. PerceptArena is a controllable, self-hosted web environment comprising nine scenarios and a cross-scenario layer with 652 tasks in total; each feasible task places its decisive evidence at particular locations and information carriers, making perception allocation part of the task rather than a fixed input. We evaluate 18 vision-language models spanning frontier and mid-tier APIs under both the mainstream **H**arness-**S**upplied **O**bservation protocol (**HSO**) and our proposed **A**gent-**C**ontrolled **P**erception protocol (**ACP**). Seventeen of the eighteen models attain a higher cost-success AUC under **ACP**, and sixteen spend less metered tokens; the median model spends 59% of its **HSO** metered tokens, average success is essentially unchanged, moving 2.1 points field-wide and 0.9 points with the two degenerate baselines excluded, and evidence recall across the field rises from 60.0% to 64.7%. Nevertheless, **ACP** also exposes a shared weakness of frontier models in perception self-management: models struggle to target the decisive evidence, track what they already hold, and stop once the evidence suffices. These observations remain hidden under the traditional **HSO** protocol. PerceptArena thereby makes perception allocation a measured behavior, and turns what to look at, what to keep, and when to stop into explicit, trainable targets. The benchmark, all 652 task definitions, and the analysis code are available at https://anonymous.4open.science/r/PerceptArena/.

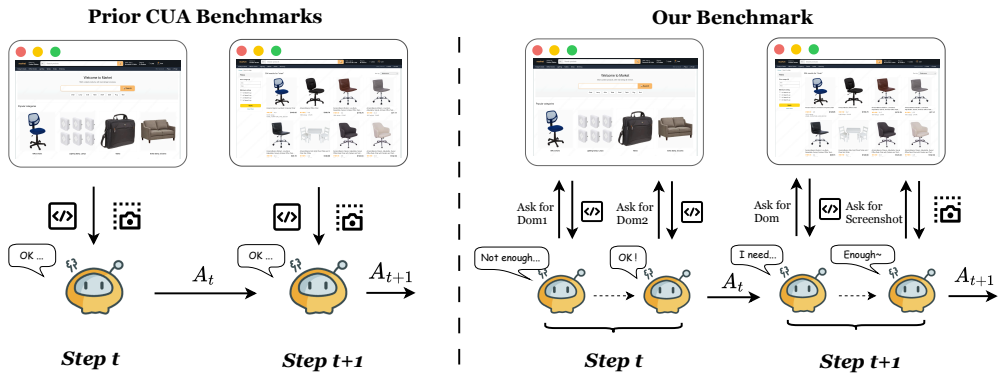


Figure 1: Supplied-observation benchmarks fix perception in the harness; PerceptArena makes it part of the agent policy.

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.

1 INTRODUCTION

Consider a web agent asked to find a stay with an ocean view and book it. Dozens of listings match the text filters, and no description states which window actually faces the sea, so the decisive fact lives only in the photographs. Opening the next listing changes the environment state; taking a screenshot of the right listing page, or zooming a cropped visual read onto its window, instead changes what the agent can observe. The first kind of choice is an ***action***; the second is a ***perception operation***, acquired to change the agent's information rather than the world. For an autonomous agent, deciding whether another observation is necessary, which one to acquire, and when the gathered evidence is sufficient to commit an irreversible booking is therefore part of the decision process itself, not a preprocessing step performed by the infrastructure.

In common evaluation setups for computer-use agents, the harness determines when and how observations are rendered and passed to the model, so the observation policy is fixed by the benchmark; we refer to this mainstream design as **H**arness-**S**upplied **O**bservation (**HSO**), sketched on the left of Figure 1. An agent under such a harness can always decide where to look, which link to follow or which page to open, but the harness decides how to look, at what fidelity and how often the page is rendered, so these setups more directly answer *what should the agent do given an observation* than *what should the agent observe before acting*. This paper proposes the complementary design, **A**gent-**C**ontrolled **P**erception (**ACP**): the agent itself chooses each observation, reads only as much as it needs, and stops once the evidence suffices to commit. The distinction matters for three reasons: repeated large observations carry a real context cost; some decisive facts live in pixels or rendered structures that a page-level text observation does not carry; and an agent that controls its own perception must also decide when to stop. PerceptArena provides the control that answering this question requires: a self-hosted web ecosystem of ten sites and 652 tasks whose decisive evidence is hidden in location and carrier, a programmatic evaluator, and the measured token footprint of every observation.

We evaluate 18 vision-language models spanning frontier and mid-tier APIs under both protocols on the same 652 tasks. The protocol choice changes what the benchmark measures; the findings are stated once in the contributions below and dissected in Section 5.

This paper makes three contributions.

- We formulate perception allocation in web agents as an explicit, measured decision process covering what to look at, what the agent already holds, and when to stop, with every decision made by the agent itself.
- We build PerceptArena, a 652-task web benchmark spanning nine scenarios and a cross-scenario layer, with diverse decisive-evidence carriers and programmatic evaluation, and we implement two harnesses over the same environment and task set: an **HSO** harness representing the mainstream computer-use paradigm and an **ACP** harness realizing our active-perception paradigm.
- We evaluate 18 vision-language models under both harnesses and find that seventeen of them attain a higher cost-success AUC under **ACP** while sixteen spend less metered tokens, yet under **ACP** frontier models remain poor managers of their own perception: they cannot aim their perception at the decisive evidence, do not track what they already hold, and cannot tell when the evidence suffices. PerceptArena turns these three decisions into measured behavior, giving the next generation of agents explicit, trainable targets for managing perception rather than merely consuming it.

2 RELATED WORK

Web and computer-use agent benchmarks. Existing benchmarks advance along environment realism and task breadth: WebArena (Zhou et al., 2024), VisualWebArena (Koh et al., 2024), WorkArena (Drouin et al., 2024) and BrowserGym (de Chezelles et al., 2025), OSWorld (Xie et al., 2024), Mind2Web (Deng et al., 2023) and Online-Mind2Web (Xue et al., 2025), and BrowseComp (Wei et al., 2025), with multimodal successors such as MM-BrowseComp (Li et al., 2025) and BrowseComp-V^3 (Zhang et al., 2026b) carrying the evidence in images. To our knowledge they share one harness design: the environment renders each observation and injects it into the model context, so the observation policy is fixed by the benchmark or the scaffold and what gets

Table 1: Positioning PerceptArena against representative benchmarks; column criteria are defined in the text. ● = yes, ◑ = partial, ○ = no.

Benchmark	Interactive	Pixel evidence	Agent-chosen obs.	Metered cost	Per-obs. attrib.	Protocol comp.
Web and computer-use benchmarks						
WebArena	●	○	○	○	○	○
VisualWebArena	●	●	○	○	○	◑
WorkArena	●	○	○	○	○	○
OSWorld	●	◑	○	○	○	◑
Online-Mind2Web	●	◑	○	○	○	○
BrowseComp	◑	○	◑	○	○	○
OSWorld-MCP	●	◑	◑	○	○	○
Active perception and cost-aware evaluation						
DeepEyes	○	●	◑	○	○	○
OSWorld-Human	●	◑	○	◑	◑	○
PerceptArena (ours)	●	●	●	●	●	●

measured is how to act given an observation. AgentOccam (Yang et al., 2025) shows that merely aligning the observation and action spaces lifts success substantially on WebArena; across all of them, however, the observation format is an engineering choice made by the evaluator before the run, not a behavior of the model being evaluated. Recent concurrent work has begun to expose individual observation controls, which Appendix A discusses one by one; these approaches control particular components of perception, whereas PerceptArena places acquisition, fidelity, retention, and stopping jointly under the evaluated model policy, and measures their consequences under a paired supplied-versus-agent-controlled protocol.

Closest to ours, OSWorld-MCP (Jia et al., 2025) also hands part of the choice back to the model, but over the action channel, letting it pick between GUI operations and MCP tools, while observations remain harness-supplied; BrowseComp-style benchmarks let the agent pick which pages to open, yet their evidence is carried in text and the answer itself is the deliverable. In all of these the agent can decide where to look, but the harness still decides how to look. PerceptArena instead places the decisive evidence on different carriers, photographs, tables, scanned attachments, and 3D renders, so the perception decisions sit on the critical path to task success.

Active perception and cost-aware evaluation. Active perception is the classical idea that the perceiver itself decides what to look at, how, and when, and that one more observation is worth taking only if what it adds outweighs what it costs. Recent multimodal work picks the idea up inside a single image: DeepEyes (Zheng et al., 2025) rewards models that crop and zoom during reasoning, but perception there changes no environment state, involves no cross-page evidence gathering or stopping decision, and meters no cost. A second line attends to cost: OSWorld-Human (Abhyankar et al., 2025) measures the time and steps agents waste against human reference trajectories, treating cost as an end-to-end total, and it does not contrast observation protocols on the same models over the same tasks.

Table 1 consolidates the comparison along six criteria. A benchmark counts as interactive when actions change a live environment state, as carrying pixel evidence when some decisive facts live only in pixels, and as agent-chosen when the model itself decides whether, what, and at which fidelity to observe. It meters cost when token or monetary cost is part of the reported metric, attributes per observation when cost is traced to individual perception decisions, and compares protocols when the same models run under different observation policies on the same tasks. PerceptArena is the only entry that satisfies all six: its self-hosted environment attributes every metered token to its source, and its paired **HSO** and **ACP** protocols make what to look at, at what fidelity, and when to stop a measured, trainable model behavior.

3 PERCEPTARENA

PerceptArena is a self-hosted web ecosystem of ten sites, the nine single-site scenarios plus the shared Plaza hub every episode starts from, and a 652-task suite. The sites run on real data at scale:

2,885 emails, 8,672 local-business stores with 82,694 photographs, 10,123 stay listings, 1,983 forum questions with 10,000 answers, 34,840 spreadsheet cells, and 91,473 products, with the per-site sources listed in Appendix B.2. What we build on top of that data is the evidence placement, the timed state changes, and the reward routing, because studying perception requires control over the observation interface, the environment state, and the cost accounting, and no live website will hold them still.

3.1 HARNESS DESIGN

PerceptArena benchmarks perception allocation, and the harness is built so that the allocation is posed, not solved. An episode is a sequence of turns over an environment state s_t: an observation operation p_t returns an observation o_t without changing the controlled state s_t, while exogenous state changes still arrive on their own clock; an action a_t changes the state; and a distinguished action $a_\perp$ commits the answer and ends the episode. The operations gathered up to the stop form the observation strategy

$$\pi_{\text{obs}} \;=\; \big(p_1, p_2, \ldots, p_M\big), \qquad I_T \;=\; \mathcal{I}\big(o_1, \ldots, o_M\big), \tag{1}$$

where I_T is the information the model holds at the commit, computed from the observation outputs o_i as retained after the memory operations applied to them, T the stopping time, and $M \le T$ the number of perception operations among those turns; committing before the reads are done is a premature stop, and observing without ever committing an overdue one. What a protocol fixes is how observations are acquired and when acquisition ends, and the two protocols of PerceptArena instantiate that control differently.

Agent-Controlled Perception (ACP). Under **ACP** the agent owns both controls. It starts from nothing: it receives only the task instruction, holds no observation of the environment, and must build its entire understanding itself. At every turn it chooses one operation from the instrument menu $\mathcal{P}$, which offers an accessibility read at adjustable depth, a screenshot, a one-question visual read on a cropped element, and a server-side 3D render at a chosen view angle, so the same decisive fact is readable at several footprints and fidelity tiers. Both controls are outputs of the model's policy π_θ:

$$u_t \;\sim\; \pi_\theta\big(\,\cdot \,\big|\, I_{t-1}\big), \qquad u_t \in \mathcal{P} \cup \mathcal{A} \cup \mathcal{M} \cup \{a_\perp\}, \qquad T \;=\; \min\big\{\, t \,:\, u_t = a_\perp \,\big\}. \tag{2}$$

Here I_{t-1} is the information the model holds entering turn t, each turn emits one control u_t, a perception operation, an action, a memory operation, or the commit $a_\perp$, so the strategy and the stop are the model's own. The agent-managed memory tools that summarize or drop held content sit alongside the instrument menu, and their tokens are in the ledger.

Harness-Supplied Observation (HSO). Under **HSO** the observation policy and its schedule move to the harness. At every step the harness renders the current page through a fixed rule h into a full-page accessibility read, capped at 16k characters, together with a full-viewport screenshot, and injects both into the model context before the model decides, so the model can no longer choose whether to acquire another observation; it can end the acquisition process only by committing. The injected observation persists in the conversation, the same agent-managed memory tools are available to compress it, and the perception instruments are disabled. Writing τ_{commit} for the time of the model's voluntary commit, $\min\{t : a_t = a_\perp\}$, and K for the step cap at which the harness force-submits the agent's final answer, the protocol is the pair of constants (h, K):

$$o_t \;=\; h(s_t), \qquad T \;=\; \min\big\{\, \tau_{\text{commit}},\, K \,\big\}. \tag{3}$$

This is the convention of existing computer-use benchmarks. Equations 1 to 3 use one shared state, observation, and information notation; only who sets the controls changes, the policy π_θ under **ACP** and the pair (h, K) under **HSO**.

The two protocols share the environment, task set, scorer, memory policy, and step cap, and the metering rule is the same for both: every model call is billed at its reported tokens. An observation enters the model context at its footprint $c(p_t)$ and is processed again by every later call. Under a unit re-processing cost, the measured cost therefore accumulates over the whole strategy rather than the single call,

$$C_{\text{obs}}(\pi_{\text{obs}}) \;=\; \sum_{i=1}^{M} c(p_i)\,\big(1 + r_i\big), \tag{4}$$

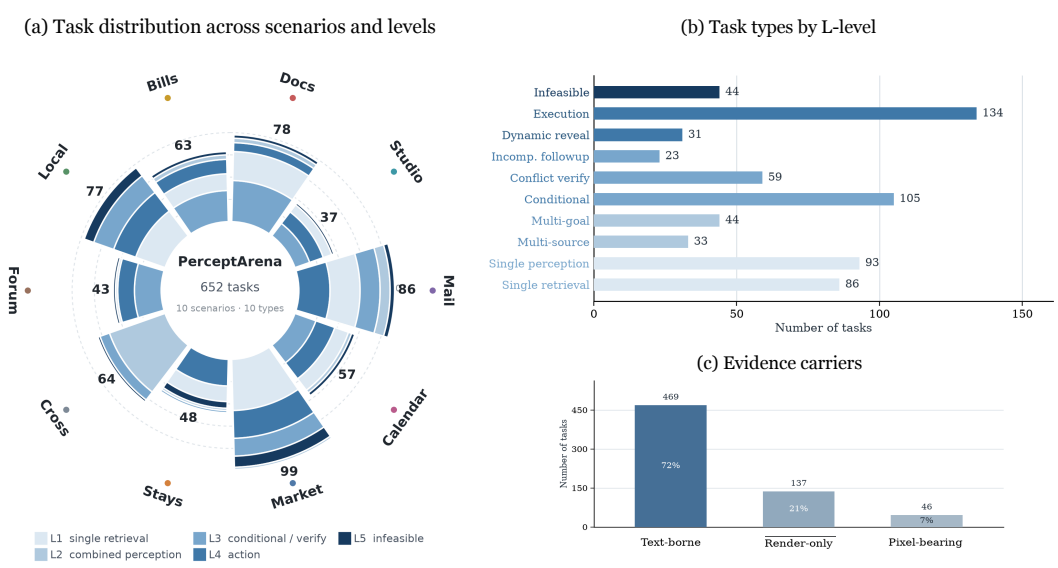


Figure 2: The 652-task suite across scenarios, diagnostic levels, and evidence carriers.

where r_i is the number of later model calls that process o_i again; under **HSO** the summand is the injected observation, under **ACP** the chosen one. This is a schematic decomposition; the exact accounting behind the reported numbers is the token ledger of Section 4.1. An observation is therefore never free and never one-off: a repeated broad read costs more than a targeted one, and a page never looked at costs nothing. Figure 1 walks one task through the **ACP** loop, and the complete tool contract and protocol inventory are in Appendix C.

3.2 Benchmark Composition

Each task states a first-person goal in domain language and ends with a commit; in between, the agent must find the decisive evidence itself, and a ten-item admission review verifies on every task that finding it genuinely requires explicit information acquisition rather than luck or guessing (Appendix B). The suite is described along three axes, shown together in Figure 2.

Scenarios. PerceptArena comprises nine single-site scenarios: `shop`, `stay`, and `local` probe pixel-based information discovery; `docs` and `mail` probe structured evidence; `bills` and `calendar` probe irreversible decision execution and consistency checking across information sources; `design_studio` and `forumhub` widen the perception axis. A cross-scenario extension layer measures long-horizon perception and processing across scenarios. Per-scenario data sources, depth mechanisms, and verified task counts are detailed in Appendix B.2.

Task types. Tasks are classified by an information-topology axis into ten types that consolidate to five diagnostic levels, tabulated with their counts in Appendix B.3. The five levels of Figure 2 grade single-source retrieval and perception upward through cross-source combination, conditional perception and conflict verification, dynamic state changes and irreversible execution, and finally the infeasible tasks, where no valid commitment exists and the correct behavior is to recognize the impossibility and decline rather than guess.

Modality. Each task's decisive fact is carried in rendered text, in image pixels, or in an on-demand 3D render: 465 tasks are text-borne, counting plain text, attached files, and the cells of rendered tables and spreadsheets; 145 are pixel-bearing, counting photographs, charts, and the image half of text-plus-image tasks; and 42 are reachable only through the render instrument. A DOM-leakage audit enforces the intended modality separation: no task whose decisive fact lives in pixels carries its answer string in the rendered page text, so the decisive fact is unavailable from rendered text alone, although chance or prior-knowledge success remains possible (Appendix B).

Table 2: Task success and token cost under **ACP**; cost in thousand metered tokens per episode. †/‡: degenerate **HSO** cells.

	Task success			*Token cost (k)*			*vs. **HSO***	
Model	Avg@2	Pass@2	Pass^2	Total	Perc	Other	ΔAvg@2	ΔCost
Non-thinking models								
GPT-4o-mini‡	0.073	0.095	0.051	721k	634k	87k	+0.022	-10.7M
Qwen3-VL-Plus (w/o thinking)	0.242	0.348	0.137	222k	184k	37k	+0.028	+65k
Gemini-2.5-Flash (w/o thinking)†	0.252	0.330	0.173	563k	481k	82k	+0.216	+553k
GPT-5.4-mini (w/o thinking)	0.275	0.342	0.207	220k	194k	26k	-0.010	**-667k**
GPT-4.1	0.299	0.376	0.222	408k	363k	45k	**+0.123**	-655k
GLM-5V-Turbo (w/o thinking)	0.357	0.428	0.285	657k	578k	79k	+0.030	-643k
Thinking models								
Gemini-2.5-Pro (w/ thinking)	0.296	0.377	0.215	355k	306k	49k	+0.110	-14k
GPT-5.1 (w/ thinking)	0.331	0.400	0.261	**115k**	**96k**	**19k**	-0.002	-65k
Gemini-3-Flash (w/ thinking)	0.381	0.462	0.301	704k	491k	214k	+0.003	-511k
MiniMax-M3 (w/ thinking)	0.386	0.459	0.313	554k	480k	74k	-0.095	-430k
Kimi-K2.6 (w/ thinking)	0.400	0.494	0.307	454k	402k	52k	-0.093	-492k
Gemini-3.6-Flash (w/ thinking)	0.430	0.483	0.377	509k	454k	54k	-0.047	-275k
Qwen3.5 (w/ thinking)	0.437	0.506	0.368	840k	739k	100k	+0.024	-476k
Qwen3.5-Plus (w/ thinking)	0.447	0.488	0.406	895k	803k	92k	-0.007	-610k
GPT-5.2 (w/ thinking)	0.450	0.517	0.383	254k	226k	28k	-0.010	-145k
GPT-5.4 (w/ thinking)	0.480	0.538	0.422	226k	201k	25k	+0.010	-53k
Claude-Opus-4.7 (w/ thinking)	0.482	0.532	0.431	683k	549k	135k	+0.094	-494k
GPT-5.6-terra (w/ thinking)	**0.541**	**0.594**	**0.488**	264k	186k	78k	-0.011	-231k
Field	0.364	0.432	0.297	480k	409k	71k	+0.021	-880k

4 EXPERIMENTS

We evaluate 18 vision-language models, all accessed through APIs, on the full 652-task suite under both harnesses of Section 3.1, with two runs per task under each protocol. Every episode is metered and programmatically scored, and under either harness every model receives the same instruction, scaffold, and memory policy.

4.1 METRICS AND SETUP

Across the two runs per task we report Avg@2, Pass@2, Pass^2, and the metered token cost, defined as the reported token usage under a fixed accounting rule. The headline object is the cost-success frontier: the cumulative fraction of tasks solved within each budget cap, whose area under the curve summarizes the success a model attains per unit of token usage. The cost is also decomposed by a token ledger that attributes every token to its cause and closes on the raw totals; Table 7 of Appendix E defines the suite.

Every episode starts on the Plaza hub with no hint about which application holds the answer, and the system prompt carries no strategy hints. The step cap is fifty steps on single-site tasks and grows with the number of sites a task spans; at the cap the harness force-submits the agent's final answer, so every episode ends with a commit. The verbatim prompt and the protocol and run settings are in Appendices G and C.

4.2 RESULTS

The ACP leaderboard. Table 2 places the eighteen models under **ACP**, the protocol they control. Average success spans a sevenfold range, from 7.3% for GPT-4o-mini to 54.1% for GPT-5.6-terra, and the models spread along the trade-off the frontier exposes. Token usage is mostly perception: observation content takes roughly 70% to 90% of every model's metered tokens, although each of those observations was chosen by the model itself.

ACP vs HSO. The last two columns of Table 2 hold the protocol comparison: the full sweep was rerun under **HSO**, the supplied-observation convention in its full-context form, and the columns

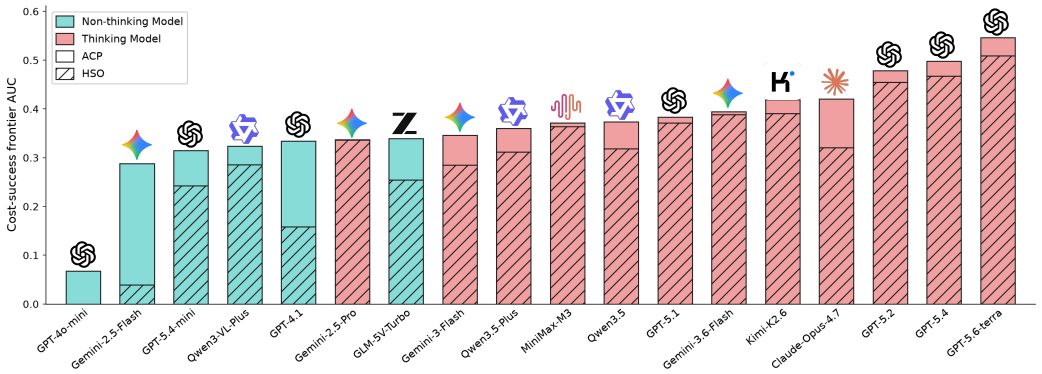


Figure 3: Cost-success AUC by model; the hatched segment is the **HSO** AUC, the solid segment the further gain to **ACP**.

report **ACP** minus **HSO** on the same tasks. Success moves little: the field mean changes by $+2.1$ points, eight of the eighteen models decline, and with the two degenerate **HSO** cells excluded the change is $+0.9$ points. Cost moves far more: sixteen of the eighteen models cut their token consumption, and the median model spends 59% of its **HSO** tokens under **ACP**. Figure 3 folds the two axes into the cost-success frontier AUC, which rewards solving more of the suite at smaller budgets; the per-model values are in Table 8 of Appendix E. Seventeen of the eighteen models attain a higher area under **ACP**, GPT-5.6-terra rising from 0.509 to 0.546, with MiniMax-M3 the single exception. Under a paired-task bootstrap with 1,000 resamples, thirteen of the eighteen AUC gains have 95% confidence intervals entirely above zero and none entirely below, and fifteen of the eighteen cost ratios have intervals below one; two further robustness checks, excluding the 42 render-only tasks reachable only under **ACP** and the `ocr` sub-call's billing, are reported with the AUC table in Appendix E.

Where the gain comes from. The cost advantage is not primarily explained by broader evidence coverage: field-wide evidence recall rises from 60.0% under **HSO** to 64.7% under **ACP**, but the rise is concentrated at the bottom of the field, with the eight lowest-**HSO**-recall models all improving and eight of the ten strongest declining by at most 2.4 points (Table 12). What handing back the control buys is a cheaper cost structure: comparable coverage at well under the tokens.

The perception decision now sits with the models, and how well they use it is what bounds the outcome; Section 5 reads the paired trajectories of the same sweep to locate where the next improvement should come from.

5 What are models still missing?

Section 4.2 located much of the remaining headroom under **ACP** in the models. Reading the paired trajectories of the same sweep traces that headroom to its source, and the trajectories show three recurring behavioral signatures of one underlying weakness: frontier models remain poor managers of their own perception. They do not know what to look at; they do not know what they already hold; and they do not know when to stop. We take the three failures in turn.

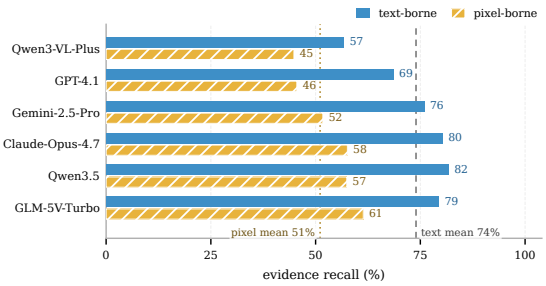


Figure 4: Evidence recall under **ACP**; solid bars are text-borne, hatched pixel-borne.

5.1 Deciding what to look at

When perception is a choice, models choose by habit rather than by aim, and the habit does not match the task.

The first miss is in the aim itself. We measure aim with evidence recall: the share of a task's minimum evidence set, annotated per task, that the trajectory actually visited. Across all **ACP** episodes the models cover 64.7% of that set, and no model covers more than 74.2% of its own minimum evidence set. The miss runs through every carrier. On the 465 text-borne tasks, the easiest case, the field still misses more than a quarter of the evidence, with mean recall 74.0%. On the 145 pixel-bearing tasks it covers 51.1%, and on the 107 text-plus-image tasks, where a picture carries part of the decisive facts alongside text, coverage collapses to 40.5%. The 3D views are almost never read: across all 3D episodes the render tool is called 0.59 times per episode on average, and four out of five 3D episodes never call it once; Figure 4 shows the miss as paired bars, with the pixel-borne recall of the six lowest-recall non-degenerate models sitting well below their text-borne recall.

The second miss is in the instrument itself. The perception menu offers a whole-tree structural read, a whole-page screenshot, a targeted fine read, and an on-demand 3D render, each at a different price and resolution, and choosing among them is the decision the benchmark hands over. The field's selection, however, collapses to the two broad instrument families: structural reads take 67.9% of all perception calls and screenshots another 28.4%, and every single model leans on them for 89.7% to 98.7% of its calls. The requests are page-wide as well: 56% of structural reads request the full tree and 32% a depth-pruned one, leaving 12% for targeted element subtrees, and 78% of screenshots request the full viewport, so page-wide output is requested for 82% of all perception calls even though every instrument permits targeted acquisition. The targeted fine read, the one instrument that resolves a small region at high fidelity, takes 3.4% of calls across the field, and on the tasks whose evidence is printed inside a picture it still takes only 6.0%. The full instrument mix is in Figure 21 of Appendix F.3, and no model breaks the pattern; the two models that lean on targeted output the most, GPT-4.1 with 10% fine reads and Qwen3-VL-Plus with 7%, request page-wide output for only 45% and 58% of their calls, so the menu is genuinely usable when a model chooses to use it. At the end of this slope, the scanned attachments in `mail`, whose decisive value is printed inside the scan itself, receive fewer than one targeted read per task and are solved 4.0% of the time.

5.2 Knowing what is already in hand

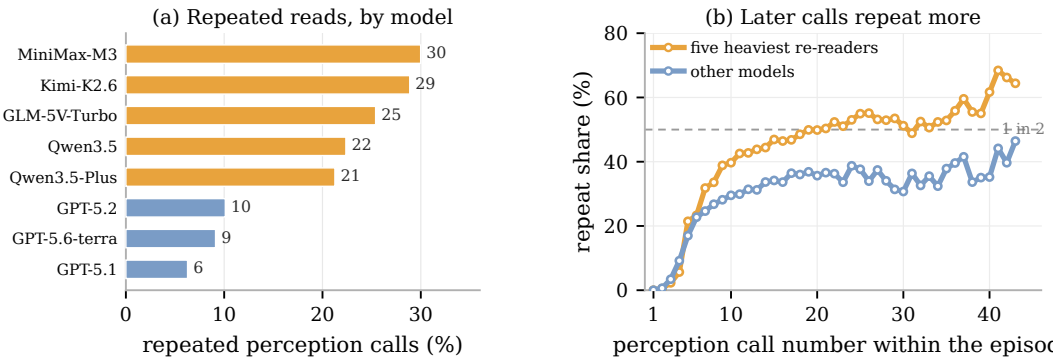


Figure 5: Re-reading under **ACP**. (a) Repeat share for representative models. (b) Repeat share by call position.

No model keeps a reliable account of what it already holds.

Figure 5a is the direct measurement: a perception call counts as a repeat when the same tool, target, and URL signature has already appeared in the episode, and repeats span 6.3% for GPT-5.1 to 30.0% for MiniMax-M3 of all perception calls. Figure 5b conditions on the position of the call: among the five heaviest re-readers the repeat share climbs as the episode ages, passes one call in two around the twentieth call, and past the fortieth passes two in three; the rest of the field climbs the same slope more slowly.

Repeated acquisition is strongly associated with higher perception cost: because the conversation keeps every observation, each repeat is metered again on every later model call. The consequence shows up directly in token usage: GPT-5.1, the lightest re-reader, spends 115k metered tokens per episode, while the five heaviest re-readers spend 450k to 900k on the same suite; the heavier the repeated reading, the higher the token cost tends to be.

5.3 KNOWING WHEN TO STOP

Models broadly do not know when to stop, and the failure takes two forms: stopping too early and stopping too late.

The first form is stopping too early. On feasible tasks, false verdicts of infeasibility run from 2.6% for GPT-5.6-terra to 85.7% for GPT-4o-mini, and the median episode behind such a verdict has covered none of the annotated evidence. The model gives up without ever looking. These verdicts arrive late, after a median of forty steps, not as a prompt reaction to the budget note. On the genuinely infeasible tasks the field declines correctly only 68.5% of the time.

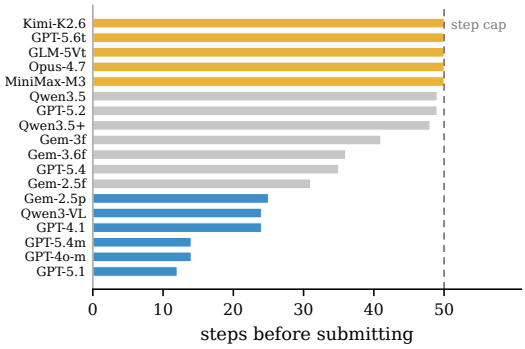


Figure 6: Steps used before submitting on one `mail` task under **ACP**; amber bars reach the 50-step cap.

The second form is stopping too late. 78.7% of the episodes MiniMax-M3 loses end in a forced submission at the step cap, with 62.0% for Kimi-K2.6, 53.5% for GLM-5V-Turbo, and about half for both Qwen3.5 variants; these five are the heaviest re-readers of Section 5.2. The evidence was already in hand: of the **ACP** episodes that run into the step cap, 53% hold the complete evidence set when the cap arrives, and convert it into success less than one time in five. Figure 6 shows both forms on one `mail` task whose decisive fact sits inside an attachment: six models submit within 25 steps having gathered none of the evidence, five ride to the cap with all of it in hand. What constitutes sufficiency, how much evidence a commit deserves, is a judgment the evaluated models make poorly: they excel at answering with what they are given, but remain unreliable judges of when enough has been acquired.

6 CONCLUSION AND LIMITATIONS

PerceptArena began from a simple observation: computer-use benchmarks hand the agent its observations, so the most consequential perceptual decisions, what to acquire, at what fidelity, and when to stop, are made by the harness rather than the model and leave no behavior to measure. We built a self-hosted web ecosystem of ten sites and 652 tasks whose decisive evidence hides in particular locations and carriers, metered every observation in tokens, and returned the perception decision to the agent. Evaluating 18 frontier and mid-tier models under both the supplied-observation convention and the agent-controlled design showed that the return pays: seventeen of the eighteen models attain a higher cost-success AUC under agent control, sixteen spend fewer tokens, and average success is essentially unchanged. More revealing than the comparison itself is what the agent-controlled trajectories expose. Models aim badly, covering under two thirds of the annotated evidence on average; models lose track of what they hold, repeating up to 30% of their perception calls; and models cannot tell when to stop, giving up before any evidence is gathered and riding to the step cap with the evidence already in hand. These three failures, invisible under a harness that supplies every observation, are the measured shape of weak perception self-management, and PerceptArena turns them into explicit, trainable targets: what to look at, what to keep, and when to stop. The protocols differ in several dimensions at once, so the comparison is not a causal ablation; the evaluated **HSO** protocol is the full-context form, and both share the same memory tools. The suite leaves long-horizon planning and multi-user state out of scope; run-to-run variance is quantified by the Pass@2 gap.

AI USE STATEMENT

Generative AI tools were used in the preparation of this work for improving the readability and consistency of the manuscript, assisting with literature search, and providing feedback on the presentation of the methodology, the mathematical formulation, and the interpretation of results. Consistency-checking and analysis scripts were written with AI assistance. Visual evidence annotations for image-bearing tasks were produced with vision-language-model assistance and verified by human inspection. Generative AI tools were not used to formulate the research questions, design the

benchmark tasks and reward rules, or execute the reported evaluation runs, which were carried out by the benchmark harness on the recorded model APIs. All AI-assisted outputs were reviewed and verified by the authors: citations and mathematical claims were checked against primary sources and derivations, recomputed numbers were checked against the recorded experiment logs, and the vision-language-model annotations were individually verified by human inspection. The authors take full responsibility for the final content of the paper, including any text, claims, or artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

Everything needed to rerun the benchmark is available or located in the paper. The complete harness, the ten scenario servers, all 652 task definitions, and the programmatic graders are in the anonymized code repository at https://anonymous.4open.science/r/PerceptArena/, together with sanitized episode-level traces sufficient to regenerate every reported table and figure. The benchmark design and the two protocols are specified in Section 3.1, the verbatim agent prompts in Appendix G, the tool contract and per-episode counters in Appendix C, and the metric definitions in Appendix E. The raw source datasets behind the scenario databases are not redistributed in the repository: they are multi-gigabyte and partly carry upstream licenses, so the repository ships the code, the task definitions, and per-source data-acquisition instructions instead, and obtaining each public corpus under its own terms and running the provided scripts reconstructs every site. No component of the evaluation itself is closed.

ETHICS STATEMENT

PerceptArena combines synthetic environments with existing research datasets spanning product, lodging, local-business, forum, document, email, and design domains. We do not recruit human participants or collect new private user data. Some source datasets contain real-world user-generated or human-authored content; we use such data only for constructing research tasks and do not perform re-identification, sensitive-attribute inference, or individual-level profiling.

Importantly, we do not redistribute third-party source datasets as part of PerceptArena. The public release contains the benchmark implementation and reconstruction utilities, while users must obtain the required source datasets independently from their original providers under the corresponding licenses and terms of use. Dataset provenance, applicable licenses or usage restrictions, and acquisition instructions are documented in the accompanying materials. In particular, Yelp-dependent components require users to obtain the Yelp Open Dataset separately under Yelp's applicable terms, and no Yelp records or images are redistributed by us. We follow the applicable requirements governing the research use and publication of all third-party datasets.

REFERENCES

Reyna Abhyankar, Qi Qi, and Yiying Zhang. Osworld-human: Benchmarking the efficiency of computer-use agents, 2025. URL https://arxiv.org/abs/2506.16042.

Chayan Banerjee and Ethan Goan. Breaking the epistemic trap: Active perception under compound uncertainty, 2026. URL https://arxiv.org/abs/2605.26627.

Chen Chen, Jiawei Shao, Dakuan Lu, Haoyi Hu, Xiangcheng Liu, Hantao Yao, and Wu Liu. Gui-eyes: Tool-augmented perception for visual grounding in GUI agents. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 29350–29358. AAAI Press, 2026a. doi: 10.1609/AAAI.V40I35.40175. URL https://doi.org/10.1609/aaai.v40i35.40175.

Yuxin Chen, Yi Zhang, Zhengzhou Cai, Yaorui Shi, Zhiyuan Yao, Chenhang Cui, Jingnan Zheng, Yaqi Huo, Xi Su, Qi Gu, Xunliang Cai, Xiang Wang, An Zhang, and Tat-Seng Chua. Vitabench 2.0: Evaluating personalized and proactive agents in long-term user interactions, 2026b. URL https://arxiv.org/abs/2605.27141.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F. Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21126–21136, June 2022.

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Massimo Caccia, Alexandre Drouin, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Graham Neubig, Quentin Cappart, Russ Salakhutdinov, and Nicolas Chapados. The browsergym ecosystem for web agent research. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=5298fKGmv3. Expert Certification.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 28091–28114. Curran Associates, Inc., 2023. doi: 10.52202/075280-1220. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/5950bf290a1570ea401bf98882128160-Paper-Datasets_and_Benchmarks.pdf.

Mahir Labib Dihan, Tanzima Hashem, Mohammed Eunus Ali, and Md Rizwan Parvez. Weboperator: Action-aware tree search for autonomous agents in web environment, 2025. URL https://arxiv.org/abs/2512.12692.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. WorkArena: How capable are web agents at solving common knowledge work tasks? In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11642–11662. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/drouin24a.html.

Jingxuan Han, Wei Liu, Mingyang Zhu, Youpeng Wang, Ziwen Wang, Lin Qiu, Xuezhi Cao, Xunliang Cai, Zheren Fu, Licheng Zhang, and Zhendong Mao. Dailyreport: An open-ended benchmark for evaluating search agents on daily search tasks, 2026. URL https://arxiv.org/abs/2606.12871.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, Man Gao, Xi Su, Xiaodong Cai, Xunliang Cai, Yu Yang, and Yunke Zhao. Vitabench: Benchmarking llm agents with versatile interactive tasks in real-world applications, 2025. URL https://arxiv.org/abs/2509.26490.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. FUNSD: A dataset for form understanding in noisy scanned documents. *CoRR*, abs/1905.13538, 2019. URL http://arxiv.org/abs/1905.13538.

Hongrui Jia, Jitong Liao, Xi Zhang, Haiyang Xu, Tianbao Xie, Chaoya Jiang, Ming Yan, Si Liu, Wei Ye, and Fei Huang. Osworld-mcp: Benchmarking mcp tool invocation in computer-use agents, 2025. URL https://arxiv.org/abs/2510.24563.

Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi (eds.), *Machine Learning: ECML 2004*, pp. 217–226, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-30115-8.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905, Bangkok, Thailand, August

2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.50. URL https://aclanthology.org/2024.acl-long.50/.

Bojie Li and Noah Shi. Agent-computer observation interfaces enable dynamic computer use, 2026. URL https://arxiv.org/abs/2606.29472.

Jialian Li, Yuchen Cao, Junhong Liu, Weiran Guo, Xutao Wang, Jiaming Song, Jiahao Zhang, and Jie Chen. Engagement process: Rethinking the temporal interface of action and observation, 2026a. URL https://arxiv.org/abs/2605.11484.

Shilong Li, Xingyuan Bu, Wenjie Wang, Jiaheng Liu, Jun Dong, Haoyang He, Hao Lu, Haozhe Zhang, Chenchen Jing, Zhen Li, Chuanhao Li, Jiayi Tian, Chenchen Zhang, Tianhao Peng, Yancheng He, Jihao Gu, Hui Huang, Donghao Zhou, Yuanxing Zhang, Jian Yang, Ge Zhang, Wenhao Huang, Zhaoxiang Zhang, Qiangpeng Yang, and Shilei Wen. Mm-browsecomp: A comprehensive benchmark for multimodal browsing agents, 2025. URL https://arxiv.org/abs/2508.13186.

Wanli Li, Bowen Zhou, Yunyao Yu, Zhou Xu, Yifan Yang, Dongsheng Li, and Caihua Shan. Weavebench: A long-horizon, real-world benchmark for computer-use agents with hybrid interfaces, 2026b. URL https://arxiv.org/abs/2606.09426.

Enqi Liu, Liyuan Pan, Zhi Gao, Yan Yang, Chenrui Shi, Yang Liu, Jingrong Wu, and Qing Li. Benchmarking and improving gui agents in high-dynamic environments, 2026. URL https://arxiv.org/abs/2604.25380.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/25ae35b5b1738d80f1f03a8713e405ec-Paper-Conference.pdf.

Andrei Cristian Nica, Akshaya Vishnu Kudlu Shanbhogue, Harshil Shah, Aleix Cambray, Tudor Berariu, Lucas Maystre, and David Barber. Toward autonomous ui exploration: The uiexplorer benchmark, 2025. URL https://arxiv.org/abs/2506.17779.

Jiaqi Shao, Yuxiang Lin, Munish Prasad Lohani, Yufeng Miao, and Bing Luo. Do llm agents know how to ground, recover, and assess? a benchmark for epistemic competence in information-seeking agents, 2025. URL https://arxiv.org/abs/2509.22391.

Ningning Wang, Xavier Hu, Pai Liu, He Zhu, Yue Hou, Heyuan Huang, Shengyu Zhang, Jian Yang, Jiaheng Liu, Ge Zhang, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. Efficient agents: Building effective agents while reducing cost, 2025. URL https://arxiv.org/abs/2508.02694.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL https://arxiv.org/abs/2504.12516.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. Widesearch: Benchmarking agentic broad info-seeking, 2025. URL https://arxiv.org/abs/2508.07999.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094. Curran Associates, Inc., 2024. doi: 10.52202/079017-1650. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/5d413e48f84dc61244b6be550f1cd8f5-Paper-Datasets_and_Benchmarks_Track.pdf.

Zequn Xie, Junjie Wang, Dan Yang, Jie Feng, Yue Shen, Jian Wang, and Jinjie Gu. Slimsearcher: Training efficiency-aware web agents via adaptive reward gating, 2026. URL https://arxiv.org/abs/2606.07074.

Zhenghao Xing, Ruiyang Xu, Yuxuan Wang, Jinzheng He, Ziyang Ma, Qize Yang, Yunfei Chu, Jin Xu, Junyang Lin, Chi-Wing Fu, and Pheng-Ann Heng. Native active perception as reasoning for omni-modal understanding, 2026. URL https://arxiv.org/abs/2606.19341.

Ke Xu, Yuhao Wang, Ziyang Cheng, Hongcheng Liu, Yanfeng Wang, and Yu Wang. Agentic active omni-modal perception for multi-hop audio-visual reasoning, 2026. URL https://arxiv.org/abs/2605.28192.

Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. An illusion of progress? assessing the current state of web agents. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=6jZi4HSs6o.

Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik A Chaudhari, George Karypis, and Huzefa Rangwala. Agentoccam: A simple yet strong baseline for llm-based web agents. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 97533–97565, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/f2c6e459b95694a24ac69c469a4ee746-Paper-Conference.pdf.

Jiashu Yao, Heyan Huang, Daiqing Wu, Wangke Chen, Huaxi Ai, Haoyu Wen, Zeming Liu, and Yuhang Guo. Benchmarking living-screen-native gui agents on short-video platforms, 2026. URL https://arxiv.org/abs/2606.04701.

Ziang Ye, Wentao Shi, Yuxin Liu, Yu Wang, Zhengzhou Cai, Yaorui Shi, Qi Gu, Xunliang Cai, and Fuli Feng. Look before you leap: Autonomous exploration for llm agents, 2026. URL https://arxiv.org/abs/2605.16143.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. AssistantBench: Can web agents solve realistic and time-consuming tasks? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8938–8968, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.505. URL https://aclanthology.org/2024.emnlp-main.505/.

Guijia Zhang, Yuxun Chen, Yuheng Qi, and Harry Yang. Do gui agents believe their eyes? diagnosing state-belief reliance on pixels versus structure, 2026a. URL https://arxiv.org/abs/2607.04334.

Huanyao Zhang, Jiepeng Zhou, Bo Li, Bowen Zhou, Yanzhe Shan, Haishan Lu, Zhiyong Cao, Jiaoyang Chen, Yuqian Han, Zinan Sheng, Zhengwei Tao, Hao Liang, Jialong Wu, Yang Shi, Yuanpeng He, Jiaye Lin, Qintong Zhang, Guochen Yan, Runhao Zhao, Zhengpin Li, Xiaohan Yu, Lang Mei, Chong Chen, Wentao Zhang, and Bin Cui. Browsecomp-v^3: A visual, vertical, and verifiable benchmark for multimodal browsing agents, 2026b. URL https://arxiv.org/abs/2602.12876.

Jiarui Zhang, Muzi Tao, Shangshang Wang, Ollie Liu, Xuezhe Ma, and Willie Neiswanger. An exam for active observers, 2026c. URL https://arxiv.org/abs/2607.16165.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning, 2025. URL https://arxiv.org/abs/2505.14362.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/4410c0711e9154a7a2d26f9b3816d1ef-Paper-Conference.pdf.

APPENDIX CONTENTS

A Related Work: Extended Notes 15

B Benchmark Reference 16

B.1 Task schema 16
B.2 Scenarios 17
B.3 Task types 19
B.4 Self-check 19

C Evaluation Protocols 20

D Formal Analysis: Perception as a Cost-Aware POMDP 23

E Metric Suite 27

F Detailed Experimental Results 30

F.1 Results under **HSO** 30
F.2 Cost anatomy 31
F.3 Perception behavior 34
F.4 Success stratifications 36

G Evaluation Prompts 39

H Case Studies 43

A Related work: extended notes

Section 2 positions PerceptArena against the web and computer-use benchmarks, the active-perception line, and the cost-aware evaluation line; Table 1 consolidates the comparison. This appendix adds the notes that space excludes from the main text.

Bare-question benchmarks. A second family begins from a bare question rather than a forced observation, which brings it closest to our task-only start on the surface. GAIA (Mialon et al., 2024) poses general-assistant questions that require tool use, AssistantBench (Yoran et al., 2024) tests web agents on realistic tasks, and WideSearch (Wong et al., 2025) tests broad information seeking. These works do not hand the agent a starting page, but they all define the task itself as an information need, where the correct answer is the deliverable and the action space is the fixed paradigm of query, read, and synthesize. None treats the building of environmental understanding as a capability distinct from retrieval, and none measures the token footprint of each perception call.

Active perception for LLM agents, and concurrent benchmarks. A recent wave revisits active perception for LLM-driven agents (Xing et al., 2026; Xu et al., 2026; Banerjee & Goan, 2026; Ye et al., 2026; Shao et al., 2025; Nica et al., 2025; Han et al., 2026), but these works build a single system evaluated against different baseline systems, not a benchmark-level, within-model comparison of perception regimes on a shared task set and environment. Zhang et al. (2026a) diagnose that today's GUI agents are unreliable precisely in how they form state beliefs from pixels versus structure. On the efficiency side, systems such as SlimSearcher (Xie et al., 2026), Efficient Agents (Wang et al., 2025), and WebOperator (Dihan et al., 2025) reduce end-to-end cost through training or search, but none of them prices the individual observation. Concurrent to this work, VitaBench (He et al., 2025; Chen et al., 2026b) poses long state-mutating tasks with rubric scoring, and WeaveBench (Li et al., 2026b) evaluates long-horizon trajectories under a trajectory-aware judge, but neither attaches a metered cost to perception calls, and neither adopts the task-only, metered-cost, web-perception setting we study.

Agent-controlled perception and concurrent observation interfaces. Concurrent work has begun to expose individual components of observation control, but these approaches stop short of making the perception policy itself the object of end-to-end agent evaluation. GUI-Eyes learns whether and how to invoke crop and zoom tools for visual grounding, concentrating active perception on local refinement of a given GUI view (Chen et al., 2026a). LivingScreen makes temporal viewing controllable on continuously evolving short-video interfaces through an agent-chosen watch duration and frame rate, but ordinary GUI actions still automatically return a screenshot, leaving the base observation loop harness-supplied (Yao et al., 2026). AOI and DynamicGUIBench address a different limitation of fixed snapshots: they place adaptive sensing in an observation interface or perception module that captures keyframes, audio, or salient video frames for the downstream agent, rather than making each acquisition decision part of the evaluated model policy (Li & Shi, 2026; Liu et al., 2026). ActiveVision makes repeated visual inspection measurable, but does so primarily over static-image reasoning tasks rather than a stateful action–perception environment (Zhang et al., 2026c); Engagement Process generalizes the temporal semantics of action and observation into decoupled event streams, but does not turn fine-grained perceptual allocation into a benchmarked model behavior (Li et al., 2026a). PerceptArena goes beyond these partial controls by making the entire perception-allocation policy explicit and measurable. The agent begins from the task alone, receives no environment observation for free, and must decide whether to observe, where to look, which carrier and fidelity to acquire, what information to retain, and when the evidence suffices to stop. These decisions unfold inside a stateful multi-site web environment with heterogeneous evidence carriers, exogenous state changes, and irreversible actions; their acquisition and downstream re-processing costs are attributed through an exact token ledger; and the same models and tasks are evaluated under paired harness-supplied and agent-controlled perception protocols. PerceptArena therefore moves beyond designing a better observation mechanism: it benchmarks whether an agent can manage its own perception.

B BENCHMARK REFERENCE

B.1 TASK SCHEMA

Every PerceptArena task is a JSON object conforming to a single schema. The schema carries the task identifier, the owning scenario, the natural-language goal given to the agent, the oracle answer used by the programmatic checker, an expected instrument sequence, a structured task-data block consumed by the site's reward engine, the annotated minimum-sufficient evidence set used by the evidence-recall metric, the per-task token budget, and the derived information-topology, source-count, perception-depth, action-risk, task-type, and difficulty-category labels used for stratified reporting. Figure 7 shows a representative email-forensics task. The goal asks the agent to reply to a reconciliation email and include the relevant details from its attached spreadsheet; the reward engine requires the reply to carry the recorded Cleared value, $349.16, and the evidence set names the single fact that suffices. The goal deliberately does not name the cell in which the Cleared value lives; that is for the agent to discover.


```
{
  "task_id": "mail_div_xlsx_table_000",
  "scenario": "mail",
  "goal": "I got an email from Solace Ltd with a reconciliation
    spreadsheet attached. Reply to that email and include the
    relevant details from the spreadsheet in your reply.",
  "oracle_answer": "reply_with_fact",
  "oracle_modality_sequence": ["read_a11y",
                               "browser_action",
                               "read_a11y",
                               "screenshot",
                               "ocr",
                               "submit_answer"],
  "task_data": {"oracle_type": "reply_with_fact",
                "oracle_key": "Cleared",
                "oracle_fact": "$349.16",
                "eval_mode": "state"},
  "evidence": [{"fact": "the Cleared value recorded in the
    xlsx_table attachment",
                "sources": ["mail.attachment"],
                "modality": "table",
                "locator": "attachment att_div_xlsx_table_000 ->
    div_xlsx_000.xlsx -> Cleared",
                "essential": true}],
  "budget_tokens": 100000
}
```


Figure 7: A representative task object.

Every site, including the Plaza, implements the same evaluation contract: a health endpoint; a session-creating home route; a task-injection route that stores the task for a session; a reward route that, when requested with a JSON accept header, returns the programmatic oracle score for a submitted answer; a live-state route that the Plaza hub reads to display session state; a reset route that drops the session between episodes; and, for sites with dynamic state, a mutation route that fires the exogenous state change described in Section 3.2. The Plaza implements the same contract for cross-tasks: task injection stores the per-site map, and the reward route splits a composite $A \mid B \mid C$ submission positionally and fans each part out to the corresponding site's own reward route on the same session identifier, returning the logical AND of the per-site rewards, so a cross task counts as solved only when every component is solved. The agent path receives only the goal string; a runtime assertion in the runner enforces that no task-data or oracle field is ever passed to the model, so the oracle stays isolated from the perception loop.

B.2 SCENARIOS

The ten sites named in the main text are the nine single-site applications below plus the Plaza hub every episode starts from. Each description names the site's data source, the decisive evidence it places, and its depth mechanism, the exogenous state change that forces the agent to re-check what it believes.

`shop` is a product-catalogue site built on the ABO dataset (Collins et al., 2022), carrying 99 tasks. Its decisive evidence is pixel-borne: product facts live in photographs rather than in the catalogue text, and a family of tasks requires rotating an on-demand 3D render to read a face the static photo never shows. The depth mechanism is commercial state: multi-object carts recompute totals and stock shifts mid-episode, so a cached belief about a price or an availability can silently expire between the read and the commit. The same fact is readable at two fidelities here: the coarse product card answers cheaply and the photograph answers reliably, so the site rewards a model that knows when the cheap tier is enough.

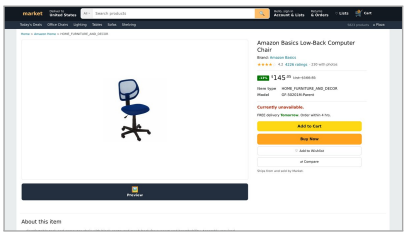


Figure 8: The `shop` site.

`mail` is an email client hosting 86 forensics-style tasks on the Enron corpus (Klimt & Yang, 2004), with document scans from FUNSD (Jaume et al., 2019) among six attachment formats. The decisive evidence is location-unknown: the decisive fact sits inside one attachment or one message of the 2,885 in the box, and the task names no message, so the agent must find where the fact lives before it can read it. The depth mechanism is the revised attachment: a message the agent has already read can gain the decisive document after the first visit. The scans hold their content as pixels, out of reach of any structural read, and folders and search are the only routing aids, so skimming the whole box is exactly the expensive habit the tasks price.

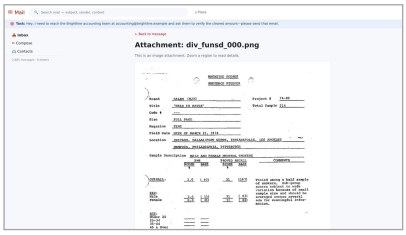


Figure 9: A scanned attachment in `mail`.

`docs` is a work suite of spreadsheets and policy documents, carrying 78 tasks over synthetic workbooks. Its decisive evidence is structural: facts live in formulas, in hidden sheets, or in cells whose displayed value depends on a chain of precedents, so a whole-tree read of the page returns numbers without the relations that decide them. The depth mechanism is the `what_if` mutation: a workbook parameter changes mid-episode and forces recompute, so the value read once is not the value that holds. Policy documents add a second carrier: a clause whose applicability depends on a date or a threshold must be found and read in full, and the surrounding text offers no shortcut to it.

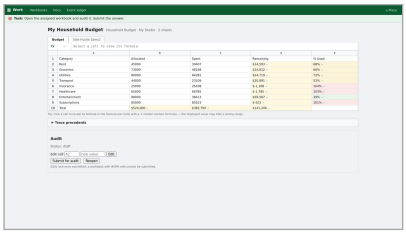


Figure 10: A workbook in `docs`.

`local` is a local-business search site built on Yelp business photographs, carrying 77 tasks over 8,672 stores. Its decisive evidence is pixel-borne ambiance: whether a venue suits the evening is printed in its photographs, the seating, the bar, the light, and not in the review text; reviews and ratings are present and truthful, but the tasks are phrased so they rarely decide, which punishes the habit of reading text first and treating photographs as decoration. The depth mechanism is dynamic availability: a table read as open may already be full when the agent commits. The store pages are photograph-heavy enough that a page-wide skim cannot substitute for the one targeted look the task needs.

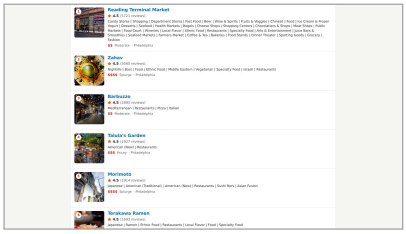


Figure 11: Business results in `local`.

cross is the cross-scenario layer, 64 composite tasks routed through the Plaza hub. A task here has no single home site: the agent receives a goal and must itself discover which of the nine sites hold each part of the evidence, moving between them through the hub, whose profile, notes, and per-site links are the only map it is given. The layer reuses the nine single-site scenarios without modification, so its difficulty comes from routing and integration rather than from new interfaces. A typical goal splits into two or three site-local facts, and the composite reward is the conjunction of the per-site answers, so one missed site fails the whole episode and the cheapest route visits no site twice.

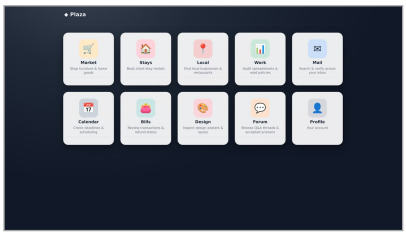


Figure 12: The Plaza hub, the entry of every episode.

bills is a wallet ledger carrying 63 tasks whose transactions derive from the persona's real orders across the ecosystem. Its decisive evidence is cross-source status: whether a refund has landed must be read here, because an order page can show refunded while settlement still lags, and the charge a task asks about can be an orphan tied to no order at all. The depth mechanism is settlement itself, a refund that completes mid-episode, and the dispute is irreversible: disputing the wrong charge is a silent failure. Flagging and disputing are the two verbs that commit, and both sit behind the same evidence gate, which makes the wallet the suite's cleanest test of committing only on facts already in hand.

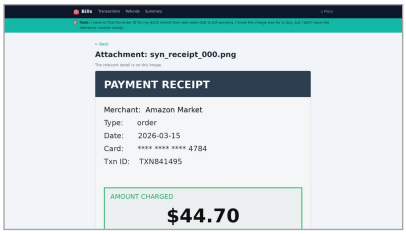


Figure 13: A receipt scan in `bills`.

calendar is a shared timeline of 57 tasks over a synthetic event corpus. Its decisive evidence is recurrence and response state: the RSVP status of a weekly meeting and the date of its next occurrence must be read from the timeline rather than inferred from the goal text, and some facts arrive only as a poster attachment. The depth mechanism is the RSVP window that closes mid-episode, and declining an invitation is irreversible, so an agent that arrives late finds a task that has turned infeasible. Events span personal, work, and shared calendars, so the fact a task asks about can sit in any one of them, and filtering by calendar is itself a perception decision that costs a read.

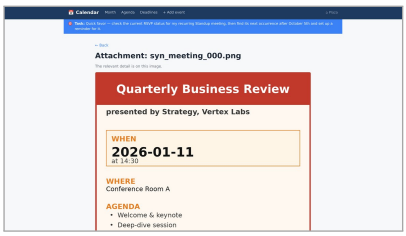


Figure 14: A meeting poster in `calendar`.

stay is an accommodation booking site built on Airbnb listings, carrying 48 tasks over 10,123 stays. Its decisive evidence is pixel-borne quality: the view from the window, the state of the kitchen, the actual bed in the room, are printed in the listing photographs rather than in the amenity checklist. The depth mechanism is inventory: a listing can book mid-episode, so availability read once is not availability held at the commit. The amenity checklist is present and truthful for the facts it states, so the tasks turn on the facts it does not state, the ones only a photograph can show, and host messages extend the carrier set one step further.

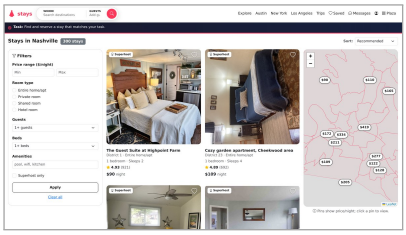


Figure 15: Listing results in `stay`.

forumhub is a question-answering forum built on a real cooking Stack Exchange corpus, carrying 43 tasks over 1,983 questions. Its decisive evidence is thread structure: the accepted answer is rarely the top-scored one, so the agent must read the thread's resolution state instead of stopping at the first plausible reply. A further family of tasks places the decisive fact inside a screenshot attachment of a thread, extending the same structure one level away from the page itself. Votes, comments, and the acceptance mark are all readable structure, and reading them in order is the

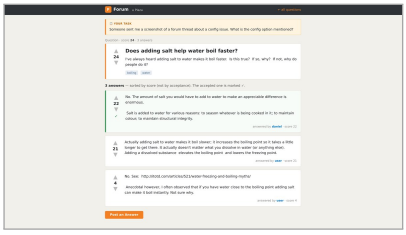


Figure 16: A question thread in `forumhub`.

difference between the first plausible reply and the resolved answer, a distinction the corpus makes rarely enough to be a real trap.

`design_studio` is a layered design editor built on the Crello, LICA, and REIP corpora, carrying 37 tasks. Its decisive evidence is the layer tree: decisive text can sit occluded beneath another layer or carry a watermark, unreadable in the rendered canvas until the stack is inspected layer by layer. The tasks make the agent reconstruct what the flattened image hides, which is the purest form of the benchmark's question, since the whole answer is present on screen but not in any single observation of it. The editor exposes the stack through a layer panel beside the canvas, so the task is legible from the interface, but it rewards the discipline of inspecting the stack rather than viewing the flattened result.



Figure 17: The layer editor in `design_studio`.

The ten site suites hold 652 verified tasks in total.

B.3 Task types

Table 3 lists the ten task types with what each tests, the diagnostic level it maps to, and the verified task count.

Table 3: The ten task types, their diagnostic levels, and verified counts.

Task type	What it tests	Level	n
`single_retrieval`	One source, fixed plan	L1	86
`single_perception`	One focused perception decision	L1	93
`multi_source_retrieval`	Distributed, one carrier	L2	33
`multi_goal_composite`	Distributed, multi-part oracle	L2	44
`conditional_perception`	Next perception depends on prior	L3	105
`conflict_verify`	Two sources disagree; find the truth	L3	59
`incomplete_followup`	A fact is missing; seek the follow-up	L3	23
`dynamic_reveal`	Facts mutate mid-episode; re-read	L4	31
`execution`	Irreversible commit gated on perception	L4	134
`infeasible`	No valid commitment; decline	L5	44
Total			**652**

B.4 Self-check

Every task is admitted only after passing a ten-item self-check, any failed item triggering a redesign. The check operationalizes the design intent of Section 3: full perception must blow the budget by combinatorial breadth; the evidence location must be unknown, with a coarse-to-fine structure; on the pixel-bearing tasks the key fact must not leak into the rendered page text, which the DOM-leakage audit enforces; a blind agent must not be able to guess it; the same fact must be readable at two or more cost-fidelity tiers; a cheap risk cue must precede any irreversible action; a human-verified in-budget oracle path must exist; a programmatic checker must exist; the minimum-sufficient evidence set must be annotated; and the difficulty must come only from decisions, never from noise. The no-LLM regression scripts, namely the DOM-leakage audit, the strategy gate, the render check, the state-isolation test, and the smoke test, are the executable form of this check, and we re-run them after any change to tasks or sites. Beyond the scripts, every admitted task is reviewed by expert annotators for goal phrasing, evidence placement, and oracle validity; annotator agreement on the review labels reaches a Cohen's κ of 0.83.

C EVALUATION PROTOCOLS

ACP and **HSO** differ in how observations are acquired and exposed to the model. The underlying harness, meaning the runtime that executes episodes, accounts cost, and enforces the step cap, is identical across both, as are the model, task set, shared prompt scaffold with protocol-specific perception contracts, and agent-managed memory policy.

C.1 EPISODE SETUP

Every episode, under either protocol, follows task-only initialization: the browser opens on the Plaza rather than the target site, and the agent receives no hint about which application, page, or evidence carrier holds the answer. The system prompt is scenario-agnostic and deliberately carries no strategy hints, because hints such as preferring cheap reads or escalating fidelity would silently pre-solve the perception decision the benchmark tests. The prompt tells the agent only that it will receive a task description, that every perception call costs tokens, and that some actions are irreversible, together with the tool contract and the output format; it is reproduced verbatim in Appendix G.

Each task carries a designed reference budget set by a human-verified efficient solution path; the budget is reported per task and never terminates an episode at runtime, while the cost-success frontier itself uses the fixed global budget grid of Appendix E. Three counters are reported per episode and are not the same unit: a step is one main-loop turn (one agent turn under **ACP**, one injected observation round under **HSO**), a model call is any language-model call including perception subcalls such as `ocr` and `compress_history`, and a tool call is any harness-recorded tool execution, which under **HSO** includes the two automatic injections per step. The step cap is fifty steps on single-site tasks, eighty on two-site tasks, and ninety-five on three-site tasks, and at the cap the harness force-submits a final answer on the agent's behalf so that every episode produces a commit. Cost is the metered token cost, the reported token usage under a fixed accounting rule: the sum of prompt and completion tokens across all calls, with cache tokens at half price; completion tokens include the reasoning tokens emitted by thinking models, so reasoning is inside the ledger, not an unbilled extra. Each model runs two episodes per task under each protocol; every reported success and cost derives from the recorded episodes, and the cost averages follow the mean recorded cost of Equation 12. The per-cell configuration files shipped with the release carry the exact model identifier, code revision, and start time of every cell.

C.2 TOOL DESIGN

Under **ACP** the agent acquires every observation itself. Table 4 lists the tool contract: four perception instruments, the browser verb set, and the commit. Precision is an argument rather than a tool name: each instrument takes a locator, a tree depth, or a view angle that sets how much of the page it returns, so the same decisive fact is readable at several footprints and fidelity tiers, from a full-page accessibility read down to a one-question visual read on a single crop. Every tool output stays in the conversation and is re-billed by each later call (Equation 4), so the footprint of a call follows from the size and form of its output.

Table 4: The **ACP** tool contract; precision is set by the arguments.

Tool	Arguments	Returns	Footprint
		Perception instruments	
`read_a11y`	`bid`, `depth`	The page's accessibility tree as text: one element's subtree, a depth-pruned tree, or the full page.	Text; grows with the region read.
`screenshot`	`locator`	An image: the full viewport, an element crop, or a pixel-coordinate crop.	Image; grows with the crop.
`ocr`	`locator`, `question`	One focused visual question, answered on the crop by the same model in a metered sub-call.	Short text; the sub-call is billed to the episode.
`render_3d_view`	`asin`, `view`, `azimuth`, `elevation`	A server-side render of the product's 3D model at a chosen camera.	Image; on demand, shop only.
		Actions	
`browser_action`	action string	Any browser verb: click, fill, press, select option, scroll, hover, navigate, drag, coordinate click or drag.	Short result; changes state.
`wait`	none	Advances the environment clock by one tick.	Short result; no state change.
`set_slider`	`bid`, `value`	Sets a range slider and fires its input and change events.	Short result; changes state.
		Memory	
`compress_history`	`upto_turn`, `keep_ids`	Summarizes the turns up to `upto_turn`, keeping the ids in `keep_ids`; the summary is generated by the same model in a metered sub-call.	Removes earlier content.
`forget_image` / `downgrade_image`	`turn_id`	Drops an earlier screenshot from the context, or replaces it with its text description.	Removes content from later prompts.
		Commit	
`submit_answer`	`answer`	The programmatic oracle score; ends the episode. `INFEASIBLE` declines the task.	Terminal; irreversible on execution tasks.

C.3 **ACP–HSO** PROTOCOL COMPARISON

Section 3.1 defines the two protocols, so we do not restate them here; Tables 5 and 6 give the complete design inventory instead. Table 5 lists the choices on which the protocols differ, from the observation source to the billing of repeated reads, and Table 6 lists the settings the two share. One structural note belongs in the inventory: under either protocol a single observation covers only the current page, so the agent reaches another site only by navigating there; what the cross-scenario layer tests is holding information across those moves.

The two protocols also share the memory tools verbatim, so the cost comparison is not confounded by a missing ability to compress context under **HSO**. Recorded **HSO** episodes actually use them more often: 21.3% of **HSO** episodes invoke at least one memory tool against 4.7% under **ACP**, and per thousand episodes **HSO** makes about 1,327 memory calls against 128 under **ACP**, yet **HSO** remains the more expensive protocol for sixteen of the eighteen models.

Table 5: Protocol-specific design choices for **HSO** and **ACP**.

Dimension	**HSO**	**ACP**
Observation source	Injected every step: full-page accessibility read, capped at 16k characters, plus a full-viewport screenshot.	None injected; the agent calls perception tools itself.
Perception tools	All four perception instruments are disabled.	All four available, at self-chosen granularity (bid, depth, crop, view angle).
Per-step user message	The injected observation itself.	Only the step index and current URL.
Billing	The injected observation stays in the conversation and is processed again by every later call.	Every call is billed at the reported prompt and completion tokens.

Table 6: Evaluation settings shared by **HSO** and **ACP**.

Dimension	Shared setting
Initialization	Episode opens on the Plaza; task-only start with no scenario hint.
Memory management	Agent-managed under both protocols: the harness never removes a message. The agent may compress its own history into a summary, or drop or downgrade images, and a context-usage notice appears at 80% of the 120k-token context limit.
Action tools	Full browser verb set, slider control, wait, and submit.
Step cap	Fifty steps on single-site tasks, eighty on two-site, ninety-five on three-site; at the cap the harness force-submits the agent's final answer.
Token budget	A per-task designed reference budget, set by a human-verified efficient solution path, reported per task, and never used as a hard cutoff.
System prompt	Same task instruction and shared prompt scaffold; the perception section and tool contract are protocol-specific.
Episode end	A submission in one of the submit, state, or hybrid modes, an INFEASIBLE verdict, or the forced submission at the step cap.

D FORMAL ANALYSIS: PERCEPTION AS A COST-AWARE POMDP

We give one proof spine for the benchmark: an augmented POMDP makes perception, context retention, cost, and stopping part of the same control problem; a policy-class simulation condition then yields simultaneous correctness, frontier, and AUC dominance; and retained observation injection yields the cost lower bound. The simulation condition is explicit because the reported ACP–HSO numbers concern two end-to-end interfaces and are empirical unless such a coupling is established.

POMDP setup. For task j, write the augmented state and control set as

$$x_t = (\xi_j, s_t, e_t, m_t, c_t), \qquad u_t \in \mathcal{U}_t = \mathcal{A}_t \cup \mathcal{P}_t \cup \mathcal{M}_t \cup \mathcal{C}, \tag{5}$$

$$\bar{x}_t \sim \widetilde{T}(\cdot \mid x_t, u_t), \qquad x_{t+1} \sim \mathsf{E}_t(\cdot \mid \bar{x}_t, u_t), \tag{6}$$

$$\widetilde{T}^s(s' \mid x_t, p) = \mathbf{1}\{s' = s_t\} \qquad (p \in \mathcal{P}_t), \tag{7}$$

$$\tau = \min\{\tau_{\text{commit}}, K\}, \qquad b_t(x) = \Pr(x_t = x \mid I_t). \tag{8}$$

Here ξ_j contains task-specific latent facts, e_t the exogenous version process, m_t retained context and cache state, and c_t accumulated metered cost. Thus a perception control preserves the task-relevant web state under $\widetilde{T}$, while it can change context and cost and can be followed by an exogenous state transition. Under the Markov assumption, all continuation laws factor through the belief:

$$\Pr_\pi(S_j, C_j \mid I_{0:t}) = \Pr_\pi(S_j, C_j \mid t, b_t), \qquad b_{t+1} = \mathcal{B}_t(b_t, u_t, o_{t+1}), \tag{9}$$

so an optimal policy may be chosen Markov in (t, b_t). In HSO, $y_t = h(s_t)$ is supplied before the main decision and $\mathcal{P}_t$ is removed; in ACP, the policy chooses $p \in \mathcal{P}_t$. A missing voluntary commit at K is completed by the shared terminal fallback.

For two observation channels $Q(\mathrm{d}y \mid x)$ and $R(\mathrm{d}z \mid x)$ at the same decision state, $Q \succeq_{\mathrm{B}} R$ means that a kernel G satisfies

$$R(\mathrm{d}z \mid x) = \int G(\mathrm{d}z \mid y) Q(\mathrm{d}y \mid x). \tag{10}$$

Thus an observer of $Y \sim Q$ can sample $Z \sim G(\cdot \mid Y)$ and reproduce every R-measurable decision. For a fixed policy π, define the run-level population frontier by

$$\mathcal{F}_\pi(B) = \frac{1}{N} \sum_{j=1}^{N} \Pr_\pi\big(S_j = 1 \ \wedge \ C_j \le B\big), \tag{11}$$

where probability is over model and environment randomness conditional on task j. This is the natural POMDP object: the probability that one episode both succeeds and remains within B.

The reported @2 frontier is the corresponding task-level object,

$$\widehat{\mathcal{F}}^{@2}(B) = \frac{1}{N} \sum_{j=1}^{N} \mathbf{1}\left\{ \frac{1}{2} \sum_{r=1}^{2} S_{jr} \ge \frac{1}{2}, \quad \bar{C}_j \le B \right\}, \tag{12}$$

where $\bar{C}_j$ is the mean recorded cost of task j. Thus all algebra below applies at either the run or task unit; the reported AUC uses Equation 12.

Theorem (simulation dominance). Let Π_{A} and Π_{H} denote ACP and HSO policy classes. Assume there is a map $\iota : \Pi_{\mathrm{H}} \to \Pi_{\mathrm{A}}$ such that each $\pi \in \Pi_{\mathrm{H}}$ can be coupled with $\iota(\pi)$ on the same task and exogenous randomness, with

$$S_j^{\iota(\pi)} = S_j^{\pi}, \qquad C_j^{\iota(\pi)} \le C_j^{\pi} \quad \text{a.s. for every } j. \tag{13}$$

A sufficient construction is a cost-nonincreasing ACP macro-observation that simulates h, preserves the clock, retained context, and terminal fallback; Equation 10 is the information-side simulation condition. Then ACP weakly dominates HSO simultaneously in Bayes correctness, every budget frontier, and discrete AUC.

Proof. For any terminal decision rule δ_R based on a garbled channel R, sampling $Z \sim G(\cdot \mid Y)$ gives

$$\mathbb{E}\big[u(X, \delta_R(Z))\big] = \int u(x, \delta_R(z))\, R(\mathrm{d}z \mid x)\ \Pr(\mathrm{d}x) \tag{14}$$

$$= \iint u(x, \delta_R(z))\, G(\mathrm{d}z \mid y) Q(\mathrm{d}y \mid x)\ \Pr(\mathrm{d}x) \tag{15}$$

$$\le \sup_{\delta_Q} \mathbb{E}\big[u(X, \delta_Q(Y))\big], \tag{16}$$

which gives the correctness claim for $u = \mathbf{1}\{\mathrm{correct}\}$. Under Equation 13, for every j and B,

$$\mathbf{1}\{S_j^{\pi} = 1, C_j^{\pi} \le B\} \le \mathbf{1}\{S_j^{\iota(\pi)} = 1, C_j^{\iota(\pi)} \le B\}, \tag{17}$$

$$\mathcal{F}_{\pi}(B) = \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}\big[\mathbf{1}\{S_j^{\pi} = 1, C_j^{\pi} \le B\}\big] \tag{18}$$

$$\le \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}\Big[\mathbf{1}\{S_j^{\iota(\pi)} = 1, C_j^{\iota(\pi)} \le B\}\Big] = \mathcal{F}_{\iota(\pi)}(B), \tag{19}$$

$$\sup_{\pi \in \Pi_{\mathrm{H}}} \mathcal{F}_{\pi}(B) \le \sup_{\rho \in \Pi_{\mathrm{A}}} \mathcal{F}_{\rho}(B). \tag{20}$$

Because the AUC weights below are nonnegative, the same pointwise inequality proves its AUC inequality. □

The theorem is conditional, and the condition is checkable rather than free: replaying an **HSO** injection under **ACP** costs one extra model call, so the coupling holds whenever that overhead is smaller than the re-billing it saves. The reported comparison is therefore an empirical measurement of two end-to-end interfaces rather than a verification of the condition.

Cost and frontier identities. Let Λ index every billed API call, including main conversation calls and the documented visual-read, memory, and forced-submit subcalls. Under the fixed accounting rule of Appendix C, let P_ℓ, L_ℓ, and Q_ℓ be its prompt, completion, and cached prompt tokens. The metered episode cost is

$$C = \sum_{\ell \in \Lambda} (P_\ell + L_\ell) - \left\lfloor \tfrac{1}{2} \sum_{\ell \in \Lambda} Q_\ell \right\rfloor. \tag{21}$$

For n main conversation calls, let $\Delta_k = P_{k+1} - P_k$. The exact ledger identity follows by rearrangement:

$$\sum_{t=0}^{n-1} P_t = \sum_{t=0}^{n-1} \left(P_0 + \sum_{k=0}^{t-1} \Delta_k \right) \tag{22}$$

$$= nP_0 + \sum_{k=0}^{n-2} \sum_{t=k+1}^{n-1} \Delta_k \tag{23}$$

$$= nP_0 + \sum_{k=0}^{n-2} (n - 1 - k)\Delta_k. \tag{24}$$

For payload i with direct charge $d_i \ge 0$, retained footprint q_i, later re-bills r_i, and effective per-occurrence multiplier at least $\underline{\lambda}_i > 0$,

$$C_{\mathrm{obs}} \ge \sum_i d_i + \sum_i \underline{\lambda}_i q_i (1 + r_i). \tag{25}$$

Hence a fixed schedule that injects one payload of retained footprint at least $q > 0$ before each of n calls obeys

$$C_{\mathrm{obs}} \ge \underline{\lambda} q \, [n + (n-1) + \cdots + 1] \tag{26}$$

$$= \underline{\lambda} q \sum_{r=1}^{n} r = \underline{\lambda} q \, \frac{n(n+1)}{2}. \tag{27}$$

This is the conditional $\Omega(n^2)$ schedule effect behind the HSO cost anatomy: it requires retention and positive marginal billing, and weakens after compression, removal, or effective deduplication.

Frontier AUC. Let $B_0 < \cdots < B_m$ be the fixed budget grid, $D = B_m - B_0$, and $\Delta_i = B_i - B_{i-1}$. The normalized trapezoidal AUC of the run-level population frontier is

$$\mathrm{AUC}_{\mathcal{B}}(\pi) = \frac{1}{D} \sum_{i=1}^{m} \frac{\Delta_i}{2} \left(\mathcal{F}_\pi(B_{i-1}) + \mathcal{F}_\pi(B_i) \right). \tag{28}$$

Set

$$\alpha_0 = \frac{\Delta_1}{2D}, \qquad \alpha_i = \frac{\Delta_i + \Delta_{i+1}}{2D} \ (1 \le i < m), \qquad \alpha_m = \frac{\Delta_m}{2D}. \tag{29}$$

The α_i are nonnegative and sum to one.

Collecting the trapezoid coefficients gives $\alpha_i \ge 0$ and $\sum_i \alpha_i = 1$, hence

$$\mathrm{AUC}_{\mathcal{B}}(\pi) = \sum_{i=0}^{m} \alpha_i \mathcal{F}_\pi(B_i) \tag{30}$$

$$= \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_\pi \left[S_j \sum_{i=0}^{m} \alpha_i \mathbf{1}\{C_j \le B_i\} \right] \tag{31}$$

$$= \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_\pi [S_j \, \psi_{\mathcal{B}}(C_j)], \qquad \psi_{\mathcal{B}}(c) = \sum_{i=0}^{m} \alpha_i \mathbf{1}\{c \le B_i\}, \tag{32}$$

$$\mathcal{F}_\pi(B_0) \le \mathrm{AUC}_{\mathcal{B}}(\pi) \le \mathcal{F}_\pi(B_m) \le \frac{1}{N} \sum_{j=1}^{N} \Pr_\pi(S_j = 1). \tag{33}$$

Thus pointwise frontier dominance in the theorem implies AUC dominance. The converse need not hold because frontiers can cross. Replacing $\mathcal{F}_\pi$ by the reported $\widehat{\mathcal{F}}^{@2}$ of Equation 12 gives the exact algebra of the paper's @2 AUC; the observed 17-of-18 comparison is still empirical.

Stopping and failure modes. For a cost trade-off $\beta \ge 0$, let $V_t^\beta(b)$ be the optimal continuation value and $\kappa_t(b, u)$ the incremental metered cost. Its Bellman comparison is

$$V_t^\beta(b) = \max \left\{ J_t^{\mathrm{stop}}(b), \ \max_{u \in \mathcal{U}_t \setminus \mathcal{C}} J_t^u(b) \right\}, \tag{34}$$

$$J_t^{\mathrm{stop}}(b) = \max_{a \in \mathcal{C}} \mathbb{E}_b[S_j(a)] - \beta \kappa_t(b, a), \tag{35}$$

$$J_t^u(b) = -\beta \kappa_t(b, u) + \mathbb{E}\left[V_{t+1}^\beta(\mathcal{B}_t(b, u, O_{t+1})) \middle| b, u \right], \tag{36}$$

$$\mathrm{NVOI}_t(p; b) = J_t^p(b) - J_t^{u^0}(b). \tag{37}$$

Thus stopping is optimal exactly when the first line selects J_t^{stop}, while perception p is preferred to a clock-matched noninformative continuation u^0 exactly when $\mathrm{NVOI}_t(p; b) \ge 0$. The boundary condition incorporates the forced fallback at K.

Let $\mathcal{E}_j^\star(s_t)$ be the versioned sufficient evidence set and $\mathcal{G}_t$ the retained usable evidence. The three failure modes reduce to

$$D_{\mathrm{aim}} = 1 - \frac{|\mathcal{G}_{\tau^-} \cap \mathcal{E}_j^\star(s_{\tau^-})|}{|\mathcal{E}_j^\star(s_{\tau^-})|}, \tag{38}$$

$$D_{\mathrm{hand}} = \frac{1}{\max\{1, n_P\}} \sum_{i=1}^{n_P} \mathbf{1}\{\sigma_i = \sigma_{i'} \text{ for some } i' < i, \ o_i = o_{i'}, \ o_{i'} \in m_i\}, \tag{39}$$

$$D_{\mathrm{pre}} = \mathbf{1}\{\text{voluntary commit}, \ \mathcal{G}_{\tau^-} \not\supseteq \mathcal{E}_j^\star(s_{\tau^-})\}, \tag{40}$$

$$D_{\mathrm{over}} = \mathbf{1}\{\text{forced fallback}, \ \mathcal{G}_{\tau^-} \supseteq \mathcal{E}_j^\star(s_{\tau^-})\}, \tag{41}$$

where $\sigma_i = (\nu_i, \mathrm{args}_i, \mathrm{URL}_i, \mathrm{version}_i)$; the redundant-perception rate reported in Section 5.2 uses the coarser tool, target, and URL signature, an implementable upper bound on D_{hand}. Under equal controlled and exogenous transitions, assume (i) a policy whose subsequent outputs are invariant to the removal of a repeated message, (ii) retained deterministic output, and (iii) strictly lower incremental billing for a clock-matched no-op than for a repeat; then replacing an in-hand repeat preserves the terminal law and strictly reduces expected cost:

$$\mathcal{L}(S_j^{\mathrm{repeat}}, A_\tau^{\mathrm{repeat}}) = \mathcal{L}(S_j^{\mathrm{skip}}, A_\tau^{\mathrm{skip}}), \qquad \mathbb{E}[C_j^{\mathrm{skip}}] < \mathbb{E}[C_j^{\mathrm{repeat}}]. \tag{42}$$

The observed recall, repeated-call, cap, and forced-submission statistics in Section 5 are measurable proxies for these expressions. They motivate the three diagnoses but, as correlations across evaluated models, do not themselves constitute causal estimates.

E METRIC SUITE

The suite is organized around success, reliability, and cost, supplemented by process-level diagnostics that read the perception loop directly. Table 7 defines the suite. The success and cost aggregates are the headline numbers used in the main results, joined by the protocol-level deltas that carry the **ACP**-versus-**HSO** comparison of Table 2; Table 8 tabulates the per-model frontier AUCs behind Figure 3; the rest are process-level diagnostics, stratified by task type in Appendix F.

Table 7: The PerceptArena metric suite: headline aggregates and frontier AUC first, then process-level diagnostics.

Metric	Definition	Formula / mechanism	What it resists or reads
sr	Binary success of one run, scored by the programmatic grader on the decisive evidence, merging the grader's checks and constraint checks.	$\mathbf{1}[\text{reward} \geq 0.5]$	final-answer correctness
Avg@K	Mean over tasks of the mean per-run success.	$\frac{1}{N}\sum_{t=1}^{N}\frac{1}{K}\sum_{k=1}^{K} s_{t,k}$	honest success point estimate
Pass@K	Fraction of tasks solved on at least one run.	$\frac{1}{N}\sum_t \mathbf{1}[\max_k s_{t,k} \geq 0.5]$	optimistic upper bound
Pass$^\wedge$K	Fraction of tasks solved on all runs.	$\frac{1}{N}\sum_t \mathbf{1}[\min_k s_{t,k} \geq 0.5]$	reliability lower bound
Cost	Mean over tasks of the mean per-run metered tokens.	prompt + completion tokens, cache at half price	metered spend
ΔAvg@2	Change in average success, **ACP** minus **HSO**, in the unit of Avg@2.	per-model difference of Avg@2 between the two protocols on the same tasks	protocol-level payoff
ΔCost	Change in metered token cost, **ACP** minus **HSO**.	per-model difference of Cost, negative meaning **ACP** is cheaper	economy of the returned control
Frontier AUC	Area under the cumulative solved-within-budget curve on the fixed budget grid of eleven caps from 5k to 2M, trapezoid rule, normalized to $[0, 1]$; writing $\bar{s}_t$ and $\bar{c}_t$ for the mean success and mean cost of a task's two runs, a task is solved within B when $\bar{s}_t \geq 0.5$ and $\bar{c}_t \leq B$. The grid top exceeds every non-degenerate cell's mean cost.	$\mathcal{F}(B) = \frac{1}{N}\sum_t \mathbf{1}[\bar{c}_t \leq B \wedge \bar{s}_t \geq 0.5]$	over- and under-perception
Token ledger	Attribution of every metered token to its cause: observation content, scaffold, the model's visible output, memory tools, and action results.	Telescoping identity $\sum_t P_t = nP_0 + \sum_k \Delta_k(n-1-k)$ over main conversation calls, with Δ_k classified by the events of interval k; sub-calls of `compress_history` and `ocr` bill their own buckets. The buckets close exactly on the raw totals.	where the spend goes
Evidence recall	Fraction of the minimum-sufficient evidence set the trajectory covered.	anchor-level URL matching on identifier tokens, with a modality gate: image-bearing evidence counts only when read by a visual instrument; scene-level matching is the fallback upper bound.	missed evidence
Redundant-perception rate	Share of perception calls that repeat a previous (tool, target, URL) triple.	$\frac{\#\text{repeated}}{\#\text{perception calls}}$; stateful tasks that legitimately re-read can inflate it	no belief state

Table 8: Cost–success AUC by model and protocol; Δ is **ACP** minus **HSO**, on unrounded values.

Model	**HSO**	**ACP**	Δ
Non-thinking models			
GPT-4o-mini‡	0.000	0.068	+0.068
Qwen3-VL-Plus (w/o thinking)	0.286	0.324	+0.038
Gemini-2.5-Flash (w/o thinking)†	0.040	0.288	+0.248
GPT-5.4-mini (w/o thinking)	0.243	0.315	+0.072
GPT-4.1	0.159	0.334	+0.176
GLM-5V-Turbo (w/o thinking)	0.254	0.339	+0.085
Thinking models			
Gemini-2.5-Pro (w/ thinking)	0.336	0.337	+0.001
GPT-5.1 (w/ thinking)	0.371	0.384	+0.013
Gemini-3-Flash (w/ thinking)	0.285	0.346	+0.062
MiniMax-M3 (w/ thinking)	0.372	0.364	-0.008
Kimi-K2.6 (w/ thinking)	0.391	0.419	+0.028
Gemini-3.6-Flash (w/ thinking)	0.389	0.395	+0.006
Qwen3.5 (w/ thinking)	0.318	0.374	+0.056
Qwen3.5-Plus (w/ thinking)	0.312	0.360	+0.048
GPT-5.2 (w/ thinking)	0.455	0.479	+0.024
GPT-5.4 (w/ thinking)	0.467	0.498	+0.031
Claude-Opus-4.7 (w/ thinking)	0.321	0.420	**+0.100**
GPT-5.6-terra (w/ thinking)	**0.509**	**0.546**	+0.038
Field	0.306	0.366	+0.060
Field excl. degen.	0.342	0.390	+0.048

The grid is linear in tokens and the AUC integrates the trapezoid rule over it, so the top decade carries the largest weight; we keep the linear grid because the decision-relevant question, whether a task fits a practical per-episode budget, is linear in tokens, while Figure 18 plots the same curves on a log axis for readability. The frontier scores a task when at least one of its two runs succeeds while charging the mean cost of the two runs; a task that succeeds once and fails early once is therefore charged above its cheaper successful run. The headline direction is robust to this and to the grid. Scoring runs instead of tasks, requiring each successful run to fit the budget, using a pass@2 rule that scores a task when at least one successful run fits, extending the grid to 4M or 8M tokens, or making the grid logarithmic, each leaves 15 of the 18 models with the higher **ACP** area; the two models that flip, Kimi-K2.6 and Gemini-3.6-Flash, are those with the two smallest gains under the reported definition, and under a paired-task bootstrap with 1,000 resamples thirteen of the eighteen ΔAUC values have a 95% confidence interval entirely above zero and none entirely below. Two further robustness checks bound the same headline: the on-demand 3D render is reachable only under **ACP**, and excluding the 42 render-only tasks still leaves sixteen of the eighteen models with the higher area, with Gemini-3.6-Flash slipping just below parity at -0.006; and the `ocr` sub-call is a metered call at the model's own reported rate, so its cost is inside the ledger rather than an unbilled extra.

The token ledger. Every recorded call carries a tool field that separates the main conversation calls from three internal sub-calls: the history summarizer of `compress_history`, the one-question visual read of `ocr`, and the forced final answer. The ledger rests on a telescoping identity over the main calls' reported prompt sizes, which holds exactly: the total input equals the first prompt times the number of calls plus each interval's net change times the number of later calls that re-bill it. Each interval's net change is attributed to the class of the events it contains, with perception tools charging observation content, memory tools charging memory at a possibly negative net when compression removes older content, and action results charging the remaining share. The summarizer's own bill is the memory bucket's fixed part, and the visual-read sub-call bills observation content. Completion tokens that the API keeps out of the context are never re-billed, so the model's own bucket counts only its visible output. The buckets therefore sum to the raw totals by construction, and the shares reported in the main text are this accounting rather than an estimate. Two approximations enter only the split of the first prompt under the ablated design, where the fixed scaffold is estimated from the same model's agent-controlled first prompts. Because tokenizers and image-billing rates differ

across APIs, cross-model comparisons of absolute cost are in each API's own unit; the shares within one model are unit-free.

F DETAILED EXPERIMENTAL RESULTS

All rates are @2 over the raw 652 tasks each model ran, and every figure and table here plots the complete sweep: 18 models, 2 protocols, 652 tasks, two runs per task. The section opens with the full results under **HSO** behind the protocol comparison of Section 4.2, continues with the cost anatomy behind Section 5.2 and the perception-behavior diagnostics behind Sections 5.1 and 5.3, and closes with the success stratifications along the benchmark's taxonomies.

F.1 RESULTS UNDER **HSO**

Table 9 gives the full **HSO** results behind the Δ columns of Table 2; observation content takes 88% to 96% of every row outside the two degenerate cells, and † and ‡ mark the degenerate cells of Section 5.2. Figure 18 unfolds every AUC of the two protocols into its complete frontier curve: each curve is computed from all 652 tasks of the same sweep, and the typical panel shows **ACP** lying on or above **HSO** while reaching the same success at a budget several times smaller. Figure 19 shows the context-growth mechanism behind the cost structure. Its panels follow the behavioral categories of the analysis, namely evidence acquisition, perception regulation, and the stable strong models; each line is one model under one protocol, solid denoting **HSO** and dashed denoting **ACP**, and the median metered prompt size climbs over the call index by a factor of roughly nineteen to seventy over the plotted range. Together the three are the numerical reference for every **HSO** number quoted in the main text.

Table 9: Full **HSO** results behind Table 2; cost in thousand metered tokens per episode. †/‡: degenerate cells.

	Task success			*Token cost (k)*		
Model	Avg@2	Pass@2	Pass^2	Total	Perc	Other
Non-thinking models						
Gemini-2.5-Flash (w/o thinking)†	0.035	0.040	0.031	**9k**	**8k**	**1k**
GPT-4o-mini‡	0.051	0.060	0.043	11423k	11000k	423k
GPT-4.1	0.176	0.244	0.109	1063k	1012k	51k
Qwen3-VL-Plus (w/o thinking)	0.214	0.308	0.120	157k	138k	19k
GPT-5.4-mini (w/o thinking)	0.284	0.362	0.207	887k	835k	53k
GLM-5V-Turbo (w/o thinking)	0.327	0.434	0.219	1299k	1237k	62k
Thinking models						
Gemini-2.5-Pro (w/ thinking)	0.186	0.362	0.009	369k	347k	22k
GPT-5.1 (w/ thinking)	0.333	0.393	0.273	181k	169k	12k
Gemini-3-Flash (w/ thinking)	0.378	0.448	0.308	1215k	1122k	93k
Claude-Opus-4.7 (w/ thinking)	0.388	0.439	0.337	1177k	1108k	69k
Qwen3.5 (w/ thinking)	0.413	0.482	0.344	1315k	1243k	73k
Qwen3.5-Plus (w/ thinking)	0.454	0.501	0.406	1505k	1433k	72k
GPT-5.2 (w/ thinking)	0.460	0.506	0.414	399k	382k	17k
GPT-5.4 (w/ thinking)	0.470	0.517	0.423	279k	267k	12k
Gemini-3.6-Flash (w/ thinking)	0.477	0.523	0.431	784k	750k	34k
MiniMax-M3 (w/ thinking)	0.481	0.557	0.405	984k	917k	67k
Kimi-K2.6 (w/ thinking)	0.493	0.560	0.426	946k	903k	43k
GPT-5.6-terra (w/ thinking)	**0.551**	**0.592**	**0.511**	495k	438k	57k
Field	0.343	0.407	0.279	1360k	1295k	66k
Field excl. degen.	0.380	0.452	0.309	816k	769k	47k

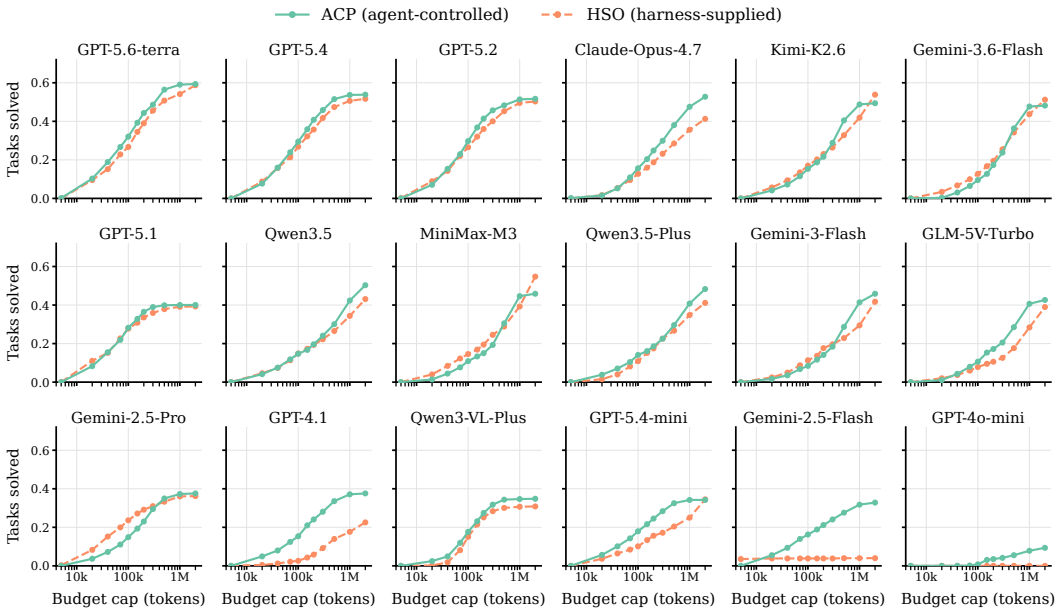


Figure 18: Cost–success frontier curves, one panel per model, ordered by **ACP** AUC: **ACP** (solid) and **HSO** (dashed) on a log token axis.

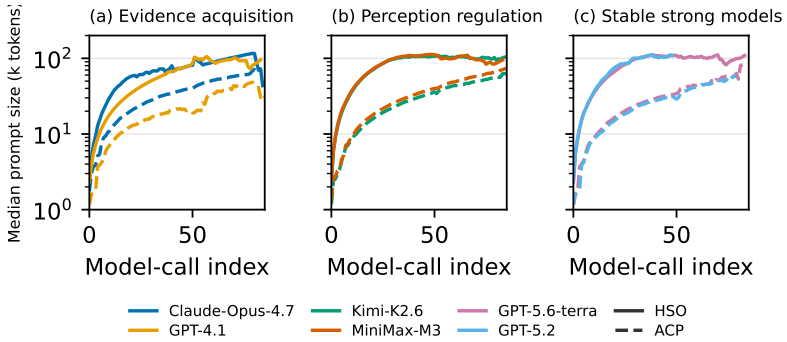


Figure 19: Median metered prompt size by model-call index; solid lines **HSO**, dashed **ACP**.

F.2 Cost anatomy

Table 10 gives the full token ledger behind Section 5.2, on the raw provider basis, for both protocols: Obs. is observation content before removals, Scaf. the fixed scaffold, Out the model's own visible output, Reas. its reasoning tokens, and Mem the memory tools' own bills plus net removals, so negative entries mean compression removed more than it added. Obs. can exceed the episode total for the same reason: each observation is re-billed at every later call, and the removals that offset it live in the negative Mem entries. This table decomposes raw tokens; Table 2 reports metered tokens, which subtract the cache discount, so the two are not column-by-column comparable. Figure 20 gives the stacked-bar view of the **ACP** ledger, one bar per model ordered by frontier AUC with the field mean on top, splitting raw token usage into observation content and everything else. The **HSO** panel of the table doubles as the per-model detail behind the cost structure of the supplied design, and observation content dominates every row under both protocols.

Table 10: The full token ledger per model, in raw tokens per episode; Reas. is reasoning tokens, billed once.

Model	Protocol	Total/ep	Obs.	Scaf.	Out	Reas.	Mem
GPT-5.6-terra (w/ thinking)	**ACP**	493k	396k	32k	111k	3	-48k
	HSO	852k	924k	21k	75k	2	-170k
Claude-Opus-4.7 (w/ thinking)	**ACP**	699k	587k	55k	83k	0	-25k
	HSO	1.32M	1.49M	47k	30k	0	-244k
GPT-5.4 (w/ thinking)	**ACP**	343k	305k	28k	10k	0	0
	HSO	368k	358k	14k	3k	0	-6k
GPT-5.2 (w/ thinking)	**ACP**	417k	370k	32k	14k	0	0
	HSO	601k	604k	18k	8k	0	-29k
Qwen3.5-Plus (w/ thinking)	**ACP**	899k	810k	46k	40k	6	-4k
	HSO	1.46M	1.69M	32k	32k	6	-300k
Qwen3.5 (w/ thinking)	**ACP**	840k	742k	45k	55k	0	-2k
	HSO	1.32M	1.5M	33k	40k	0	-258k
Gemini-3.6-Flash (w/ thinking)	**ACP**	675k	604k	52k	19k	1	-185
	HSO	1.1M	1.11M	27k	19k	1	-52k
Kimi-K2.6 (w/ thinking)	**ACP**	836k	741k	48k	39k	9	-1k
	HSO	1.67M	1.96M	32k	31k	12	-361k
MiniMax-M3 (w/ thinking)	**ACP**	1.06M	917k	60k	78k	3	-573
	HSO	1.79M	1.9M	38k	78k	5	-239k
Gemini-3-Flash (w/ thinking)	**ACP**	1.11M	868k	53k	279k	8	-86k
	HSO	1.99M	1.96M	38k	99k	16	-116k
GLM-5V-Turbo (w/o thinking)	**ACP**	735k	649k	49k	36k	4	-2k
	HSO	1.31M	1.46M	33k	26k	4	-209k
GPT-5.1 (w/ thinking)	**ACP**	149k	124k	19k	6k	0	-3
	HSO	245k	230k	12k	4k	0	-67
GPT-4.1	**ACP**	546k	495k	40k	20k	0	-9k
	HSO	1.22M	1.35M	35k	24k	0	-183k
Gemini-2.5-Pro (w/ thinking)	**ACP**	574k	498k	44k	33k	3	-2k
	HSO	539k	613k	16k	15k	2	-105k
GPT-5.4-mini (w/o thinking)	**ACP**	318k	281k	28k	10k	0	-678
	HSO	1.23M	1.18M	27k	46k	0	-23k
Gemini-2.5-Flash (w/o thinking)†	**ACP**	906k	815k	43k	88k	2	-40k
	HSO	14k	12k	2k	263	0	-556
Qwen3-VL-Plus (w/o thinking)	**ACP**	389k	329k	38k	28k	0	-6k
	HSO	242k	215k	16k	13k	0	-2k
GPT-4o-mini‡	**ACP**	881k	1.27M	61k	45k	0	-491k
	HSO	11.55M	32.81M	58k	371k	0	-21.68M

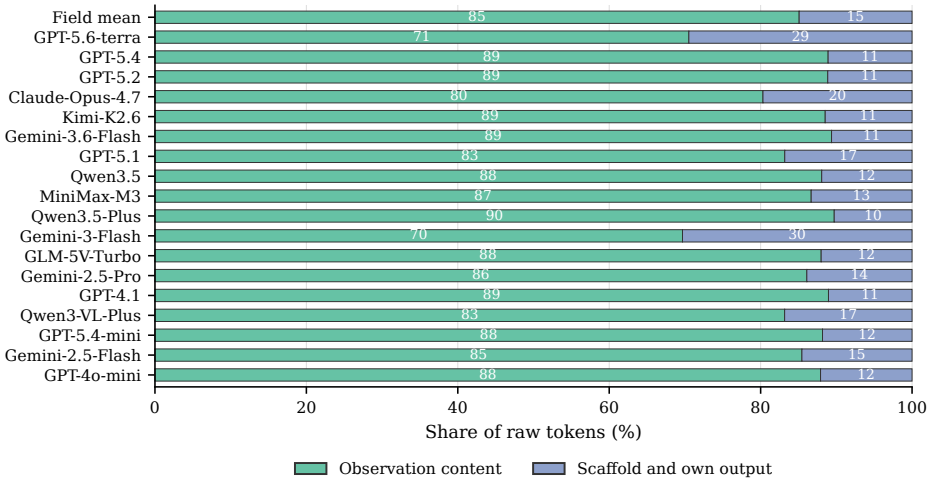


Figure 20: The **ACP** token ledger, ordered by frontier AUC with the field mean on top.

F.3 PERCEPTION BEHAVIOR

Instrument selection. Table 11 is the full instrument-selection matrix behind the bulk habits Section 5.1 names. The screenshot-heavy photographers and the read-heavy remainder are visible in it, as is the near-absence of the focused visual read and the 3D render across the field.

Table 11: Instrument selection under **ACP**: mean calls per episode.

Model	A11y read	Screenshot	Focused read	3D render	Memory
Screenshot-heavy models					
MiniMax-M3	6.91	14.14	0.33	0.08	0.00
GLM-5V-Turbo	6.56	9.55	0.47	0.02	0.01
Kimi-K2.6	8.02	9.41	0.20	0.02	0.01
Claude-Opus-4.7	6.54	7.39	0.41	0.09	0.03
Read-heavy models					
Gemini-3-Flash	10.61	1.27	0.55	0.09	0.13
GPT-4.1	10.43	2.68	1.49	0.03	0.05
Qwen3.5	10.30	3.45	0.42	0.05	0.01
Gemini-3.6-Flash	10.20	1.44	0.13	0.10	0.00
Qwen3.5-Plus	9.87	3.47	0.15	0.04	0.01
GPT-5.6-terra	9.69	1.04	0.20	0.04	0.00
GPT-5.4	8.81	0.71	0.36	0.01	0.00
GPT-5.2	8.47	0.94	0.30	0.03	0.00
GPT-5.4-mini	8.35	0.34	0.15	0.01	0.00
Qwen3-VL-Plus	8.22	1.72	0.76	0.02	0.47
Gemini-2.5-Flash	6.85	0.33	0.32	0.00	0.11
Gemini-2.5-Pro	6.47	0.66	0.55	0.01	0.01
GPT-5.1	4.96	0.87	0.24	0.03	0.00
GPT-4o-mini	2.93	1.01	0.10	0.00	1.44

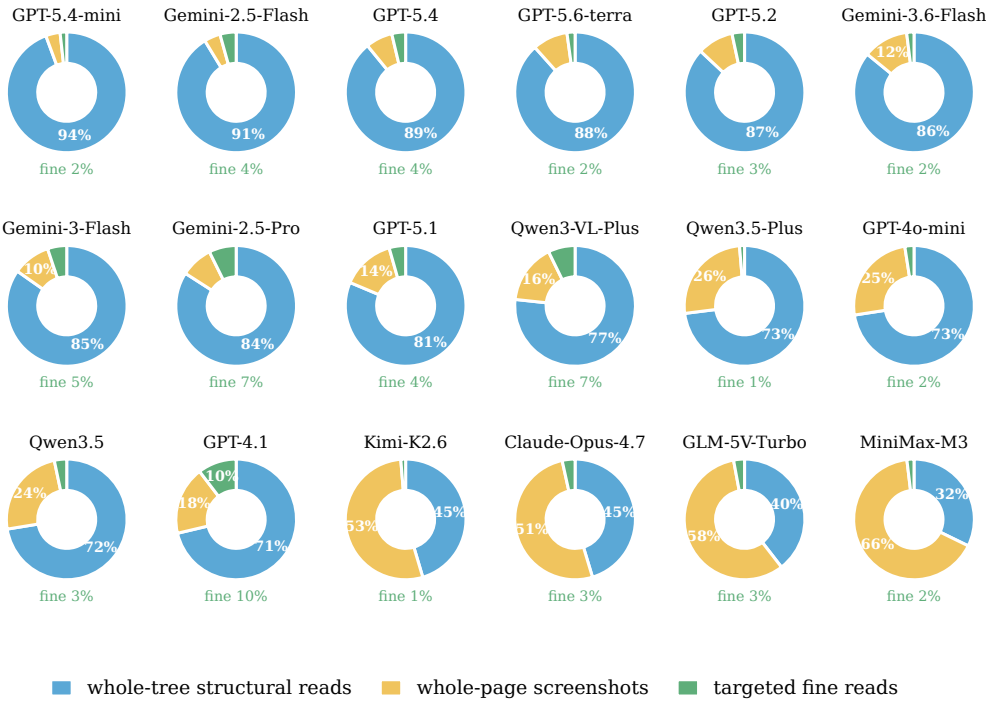


Figure 21: Instrument mix of all eighteen models under **ACP**, ordered by whole-tree read share.

Figure 21 shows the same matrix as the table above, one pie per model. The field splits into two habits that no model escapes: fourteen models read whole accessibility trees for at least seven of every ten calls, and the four heaviest photographers, MiniMax-M3, GLM-5V-Turbo, Kimi-K2.6, and Claude-Opus-4.7, invert the ratio and spend half to two thirds of their calls on whole-page

screenshots. The targeted fine read is the small slice everywhere: its largest share in the field is GPT-4.1's 10%, twelve of the eighteen models stay below 4%, and the on-demand 3D render is nearly invisible throughout.

Evidence acquisition and infeasibility. Table 12 reports, per model and protocol, the evidence recall, namely the share of a task's minimum evidence set the trajectory covered, together with the false decline on feasible tasks and the correct abstention on the 44 genuinely infeasible tasks. The lower-recall group under **HSO** gains recall across the board under **ACP**, and the judgment layer moves with it; the association between the recall change and the success change across the field is $r = 0.65$.

Table 12: Evidence recall and infeasibility judgments by model and protocol; lower is better for false decline.

	Evidence recall (%)			*False decline (%)*		*Correct abstention (%)*	
Model	**HSO**	**ACP**	Change	**HSO**	**ACP**	**HSO**	**ACP**
*Lower **HSO** evidence recall (8 models)*							
GLM-5V-Turbo	59.8	70.4	+10.6	33.0	23.0	73.9	68.2
Qwen3.5	59.6	71.9	+12.3	30.2	13.9	**89.8**	73.9
Claude-Opus-4.7	56.3	71.6	+15.3	33.7	16.3	78.4	81.8
Gemini-2.5-Flash	55.6	60.6	+4.9	**0.3**	7.2	1.1	39.8
Qwen3-VL-Plus	48.4	50.6	+2.2	57.6	42.7	88.6	78.4
Gemini-2.5-Pro	47.2	65.7	+18.5	9.0	15.6	33.0	51.1
GPT-4.1	32.2	59.0	**+26.8**	66.8	28.9	85.2	68.2
GPT-4o-mini	4.1	8.7	+4.6	66.2	85.7	76.1	86.4
*Higher **HSO** evidence recall (10 models)*							
Kimi-K2.6	**74.2**	71.8	-2.4	13.0	18.5	83.0	79.5
MiniMax-M3	74.1	**74.2**	+0.0	6.9	11.9	53.4	52.3
Gemini-3.6-Flash	74.1	72.9	-1.2	14.9	21.2	85.2	**90.9**
GPT-5.6-terra	73.5	72.3	-1.2	4.3	**2.6**	67.0	62.5
Gemini-3-Flash	72.8	73.0	+0.3	2.3	9.7	25.0	65.9
GPT-5.4	71.7	69.5	-2.2	15.3	17.2	68.2	71.6
Qwen3.5-Plus	71.6	71.4	-0.2	14.1	14.5	82.6	88.5
GPT-5.2	70.9	70.6	-0.3	24.5	22.6	68.2	68.2
GPT-5.1	68.6	67.3	-1.3	30.4	27.2	63.6	63.6
GPT-5.4-mini	65.4	63.3	-2.1	17.6	12.4	52.3	43.2

Process diagnostics. Table 13 gives the redundant-perception rate and the rubric score by model and protocol; the redundant-perception rate is the share of perception calls that repeat an already-seen tool, target, and URL triple, and the rubric score is the mean per-constraint pass rate when grader checks are available. Under **HSO** the perception instruments are disabled, so the rate there measures the harness's injection schedule rather than any model choice: the near-one values of the collapsed cells re-deliver the same page by design. Under **ACP** the rate becomes a model behavior, spans 0.063 to 0.300 across the field, and falls relative to **HSO** for seventeen of the eighteen models; the rubric score does not move systematically, which is the partial-credit view of the same episodes.

Table 13: Process diagnostics by model and protocol; lower is better for the redundant-perception rate.

	Redundant-perception rate		*Rubric score*	
Model	**HSO**	**ACP**	**HSO**	**ACP**
Thinking Models				
GPT-5.6-terra	0.452	0.091	0.558	**0.564**
Claude-Opus-4.7	0.686	0.157	0.347	0.456
GPT-5.4	0.418	0.093	0.444	0.455
GPT-5.2	0.478	0.101	0.409	0.415
Qwen3.5-Plus	0.685	0.213	0.448	0.407
Qwen3.5	0.602	0.224	0.384	0.405
Gemini-3.6-Flash	0.526	0.190	0.483	0.432
Kimi-K2.6	0.585	0.289	0.475	0.366
MiniMax-M3	0.565	0.300	0.479	0.342
Gemini-3-Flash	0.597	0.152	**0.580**	0.434
GPT-5.1	0.424	**0.063**	0.280	0.301
Gemini-2.5-Pro	0.489	0.109	0.200	0.335
Non-thinking Models				
GLM-5V-Turbo	0.688	0.254	0.296	0.324
GPT-4.1	0.853	0.172	0.135	0.260
GPT-5.4-mini	0.638	0.149	0.254	0.267
Gemini-2.5-Flash	**0.045**	0.160	0.013	0.304
Qwen3-VL-Plus	0.780	0.151	0.163	0.194
GPT-4o-mini	0.975	0.170	0.058	0.075

F.4 Success stratifications

By scenario. Table 14 reports the per-scenario change in average success between the protocols, with negative cells set in red; the absolute values behind the change appear in the released per-task CSV. Cross is the weakest column for every model, and the mean change is positive on six of the ten scenarios. Figure 22 gives the complete picture the table summarizes: every cell of the full model-by-scenario grid, under both protocols, from the same sweep, with the two panels sharing one color scale so a cell can be compared across protocols directly. The Cross column is dark throughout, the scanned-attachment Mail and pixel-bearing Stay and Shop columns separate the field into tiers, and the **HSO** panel carries the collapsed cells of the weaker models. Because the render-only family is reachable only through the on-demand 3D instrument, we also recompute the headline without it: excluding those 42 tasks, 16 of the 18 models still attain a higher frontier AUC under **ACP**, with Gemini-3.6-Flash slipping just below parity at -0.006 and MiniMax-M3 the remaining exception.

By task type. Table 15 stratifies success by task type; the infeasible column tests whether the agent declines rather than guesses, though a blanket-refusal policy also scores there, since declining everything earns $44/652 = 6.7\%$ Avg@2, the floor any abstention-heavy strategy must beat, so it is best read together with the false-decline column of Table 12. Consolidating the ten types into the five diagnostic levels of Appendix B.3, the field is worst on L2, the distributed tasks that require combining information across sources; the ladder is not monotone because L4 contains the execution tasks, whose success is high once the evidence is in hand, while L2 and L3 hold the composition and conditional-perception tasks that separate perception planning from reading.

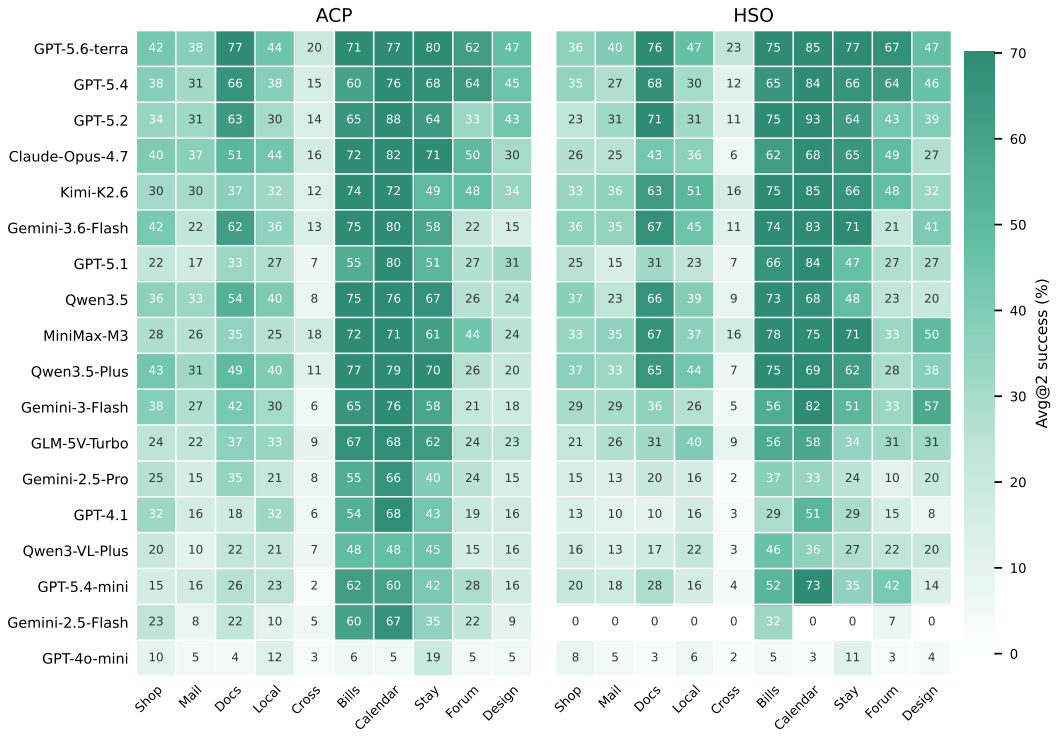


Figure 22: Success by scenario and model, **ACP** left and **HSO** right; rows ordered by **ACP** AUC.

Table 14: Per-scenario change in Avg@2, **ACP** minus **HSO**, in points; negative cells in red.

Model	Shop	Mail	Docs	Local	Cross	Bills	Cal	Stay	Forum	Des.
GPT-5.6-terra	+7	-2	+1	-3	-3	-3	-8	+3	-6	+0
Claude-Opus-4.7	+14	**+12**	+8	+7	**+9**	+10	+14	+6	+1	+3
Kimi-K2.6	-3	-6	-26	-19	-3	-1	-13	-17	+0	+1
GPT-5.4	+4	+3	-2	+8	+3	-5	-8	+2	+0	-1
GPT-5.2	+11	+0	-8	-1	+3	-10	-5	+0	-10	+4
MiniMax-M3	-6	-10	-32	-12	+2	-6	-4	-9	+12	-26
Qwen3.5-Plus	+6	-1	-15	-3	+4	+2	+10	+7	-2	-18
Qwen3.5	-1	+9	-12	+1	-1	+2	+9	+19	+2	+4
Gemini-3.6-Flash	+6	-13	-5	-9	+2	+2	-4	-12	+1	-26
Gemini-3-Flash	+10	-2	+6	+4	+1	+10	-6	+7	-12	-39
GLM-5V-Turbo	+3	-4	+6	-6	+1	+10	+11	+28	-7	-8
Gemini-2.5-Pro	+11	+2	+15	+5	+5	+17	+32	+16	+14	-5
GPT-5.1	-3	+3	+1	+5	+0	-11	-4	+4	+0	+4
GPT-5.4-mini	-5	-2	-3	+7	-2	+10	-13	+6	-14	+3
GPT-4.1	+19	+5	+8	**+16**	+3	+25	+18	+14	+3	+8
Qwen3-VL-Plus	+4	-3	+4	-1	+4	+2	+12	+18	-7	-4
GPT-4o-mini	+2	+0	+1	+6	+1	+1	+3	+7	+1	+1
Gemini-2.5-Flash	**+23**	+8	**+22**	+10	+5	**+29**	**+67**	**+35**	**+15**	**+9**
Mean	**+6**	**-1**	**-2**	**+1**	**+2**	**+5**	**+6**	**+7**	**-1**	**-5**

Table 15: **ACP** Avg@2 by task type; per column, the best value is in **bold** and the worst underlined.

Model	Single_retr.	Single_perc.	Multi_src	Multi_goal	Cond_perc.	Conflict_v.	Incompl.	Dyn_reveal	Exec.	Infeas.
GPT-5.6-terra	**0.52**	**0.44**	**0.45**	**0.15**	**0.54**	**0.61**	**0.67**	0.32	**0.75**	0.62
Claude-Opus-4.7	0.47	0.34	**0.45**	0.09	0.37	0.50	0.63	0.39	0.69	0.82
Kimi-K2.6	0.39	0.22	0.38	0.06	0.31	0.36	0.54	0.37	0.59	0.80
GPT-5.4	0.47	0.40	0.38	0.12	0.40	0.54	0.61	0.23	0.68	0.72
GPT-5.2	0.47	0.34	0.42	0.11	0.41	0.38	0.61	0.35	0.61	0.68
MiniMax-M3	0.40	0.20	0.44	0.11	0.32	0.35	0.48	0.35	0.59	0.52
Qwen3.5-Plus	0.48	0.34	0.38	0.09	0.33	0.36	0.57	0.44	0.60	0.88
Qwen3.5	0.44	0.31	**0.45**	0.07	0.42	0.40	0.54	0.29	0.59	0.74
Gemini-3.6-Flash	0.38	0.31	0.41	0.09	0.40	0.35	0.54	**0.45**	0.54	**0.91**
Gemini-3-Flash	0.40	0.25	0.41	0.07	0.25	0.34	0.50	0.42	0.56	0.66
GLM-5V-Turbo	0.34	0.22	0.36	0.07	0.30	0.27	0.46	0.29	0.53	0.68
Gemini-2.5-Pro	0.30	0.14	0.27	0.07	0.21	0.29	0.37	0.35	0.46	0.51
GPT-5.1	0.31	0.20	0.21	0.06	0.30	0.31	0.39	0.24	0.50	0.64
GPT-5.4-mini	0.30	0.09	0.24	0.01	0.21	0.32	0.33	0.40	0.42	0.43
GPT-4.1	0.34	0.15	0.18	0.05	0.15	0.24	0.35	0.31	0.50	0.68
Qwen3-VL-Plus	0.23	0.12	0.20	0.05	0.13	0.23	0.33	0.23	0.31	0.78
GPT-4o-mini	0.02	0.00	0.02	0.00	0.01	0.02	0.00	0.02	0.03	0.86
Gemini-2.5-Flash	0.26	0.11	0.23	0.05	0.20	0.25	0.43	0.31	0.36	0.40

G EVALUATION PROMPTS

The agent is driven by a scenario-agnostic system prompt and a tool contract; the full texts of both protocols are reproduced verbatim below, with the template variable `{goal}` shown literally. The prompt contains no task-specific routing or instrument-selection heuristic, because handing the agent an efficient perception strategy would turn the benchmark into instruction-following; what it states are the rules of the game, that every perception call costs tokens under **ACP**, that observations are supplied automatically under **HSO**, that context capacity is limited and managed by the model's own memory tools, and that a verified impossibility is declared with the `INFEASIBLE` sentinel. The two scaffolds differ only in the perception paragraph, and the two tool contracts differ only in the perception block; everything else is shared verbatim (Appendix C).

System prompt under ACP — template variable {goal} shown literally

```
You are an autonomous web agent operating across several independent
    web applications.

You receive only a task description --- no screenshot, no DOM tree, no
     hint about
which app to use. You start at a desktop portal hosting several
    independent web apps
(each app card shows its name and purpose). You must decide for
    yourself which
app(s) to open and in what order.

You accumulate observation history, but context capacity is limited.
    You will be
warned when nearing the limit; when full, you must call memory tools
    to compress
or forget old content before receiving new observations. What to keep
    and when
to compress is your decision.

Perception is self-initiated: each perception tool call costs tokens.
    What to
read, in what order, at what fidelity --- all your choice. When a
    belief you hold
may be stale, it is your judgment whether to re-verify.

Task: {goal}

Complete the task by performing any actions needed on the site(s).
    When done,
call submit_answer to end the episode.

Most tasks are solvable. A few may be genuinely impossible. Do not
    declare a
task infeasible lightly --- fully explore the relevant pages and
    actions first.
Only after you have verified that no action sequence on any available
    site can
satisfy it, declare it infeasible. To declare a task infeasible, call
submit_answer("INFEASIBLE"). Declaring a solvable task as infeasible
    scores
zero; correctly identifying a genuinely infeasible task scores full
    marks.
```

Tool contract under ACP

```
You have these tools. Each turn, output one JSON object {"tool":
    "...", "args": {...}}.
```

```
Memory (manage your observation history; when context is full,
    compress or forget
before continuing):
- `compress_history(upto_turn=N, keep_ids=[])` --- replace old
    messages [0..N] with a
  summary you write; messages in keep_ids are preserved verbatim (
    pinned, e.g. a
  key screenshot). Compression itself costs tokens.
- `forget_image(turn_id)` --- drop the screenshot of the given turn (
    text part may stay).
- `downgrade_image(turn_id)` --- downgrade the given screenshot to a
    low-detail
  thumbnail (85 tokens).

Perception (each call costs tokens; choose tool and region):
- `read_a11y(bid=None, depth=None)` --- return the accessibility tree
    as text.
  - `read_a11y()` --- full-page tree (with [bid] ids)
  - `read_a11y(depth=N)` --- full tree pruned to N levels
  - `read_a11y(bid="42")` --- that element's subtree (depth is ignored
    )
- `screenshot(locator=None)` --- return a PIL image crop.
  - `screenshot()` --- full viewport
  - `screenshot(locator="bid:42")` --- crop of an element
  - `screenshot(locator="bbox:0,0,200,200")` --- pixel-coordinate crop
     (for canvas /
    chart / map areas with no [bid])
- `ocr(locator, question)` --- ask one VLM question about a cropped
    image; `question`
  is required.
  - `ocr(locator="bid:42", question="...")` --- element crop
  - `ocr(locator="bbox:0,0,200,200", question="...")` --- coordinate
    crop
- `render_3d_view("asin", view?, azimuth?, elevation?)` --- server-
    side .glb render
  from a chosen camera angle. view presets (front/side/back/top/detail
    /bottom) or
  raw azimuth/elevation (degrees, override view). Read facts only
    visible in 3D
  (frame back, folding mechanism, base footprint). ABO shop only.

Action (drive the browser; [bid] ids come from the accessibility tree)
    :
- `browser_action("action_string")` --- any BrowserGym verb:
  - `click("bid")` | `fill("bid","text")` (fill also works on date
    inputs, format
    YYYY-MM-DD, and range sliders)
  - `press("bid","Enter")` | `select_option("bid","opt")` | `goto("url
    ")` | `go_back()`
  - `scroll(0,300)` | `hover("bid")` | `drag_and_drop("bid1","bid2")`
  - `mouse_click(x,y)` | `mouse_drag_and_drop(x1,y1,x2,y2)` (
    coordinate actions, for
    canvas / 3D viewer / slider thumbs / map dragging)
- `set_slider("bid", value)` --- set a range slider and fire its
    events (360 degree
  turntable scrub)
- `wait()` --- wait one step (advance the step counter, trigger
    possible
  controlled_reveal; costs 0 tokens but returns no new info)

Answer:
- `submit_answer("answer")` --- finalize the episode (irreversible
    when the task is a
```

```
  checkout / reservation / adjudication). The task description
    specifies what to
  submit (an id, a value, a yes/no, a decision, or multiple values).
    When the task
  requires multiple values, submit them as a single '|'-separated
    string in the
  order given by the task description. Only submit after you have the
    exact answer
  from the page --- never guess.
```

System prompt under HSO — only the perception paragraph differs

```
You are an autonomous web agent operating across several independent
    web applications.

You receive only a task description --- no screenshot, no DOM tree, no
     hint about
which app to use. You start at a desktop portal hosting several
    independent web apps
(each app card shows its name and purpose). You must decide for
    yourself which
app(s) to open and in what order.

You accumulate observation history, but context capacity is limited.
    You will be
warned when nearing the limit; when full, you must call memory tools
    to compress
or forget old content before receiving new observations. What to keep
    and when
to compress is your decision.

Perception is provided automatically: each step you receive the
    current page's full
screenshot and accessibility tree (text with [bid] ids), without
    calling any tool.

Task: {goal}

Complete the task by performing any actions needed on the site(s).
    When done,
call submit_answer to end the episode.

Most tasks are solvable. A few may be genuinely impossible. Do not
    declare a
task infeasible lightly --- fully explore the relevant pages and
    actions first.
Only after you have verified that no action sequence on any available
    site can
satisfy it, declare it infeasible. To declare a task infeasible, call
submit_answer("INFEASIBLE"). Declaring a solvable task as infeasible
    scores
zero; correctly identifying a genuinely infeasible task scores full
    marks.
```

Tool contract under HSO — identical except that the perception block is absent

```
You have these tools. Each turn, output one JSON object {"tool":
    "...", "args": {...}}.

Memory (manage your observation history; when context is full,
    compress or forget
before continuing):
```

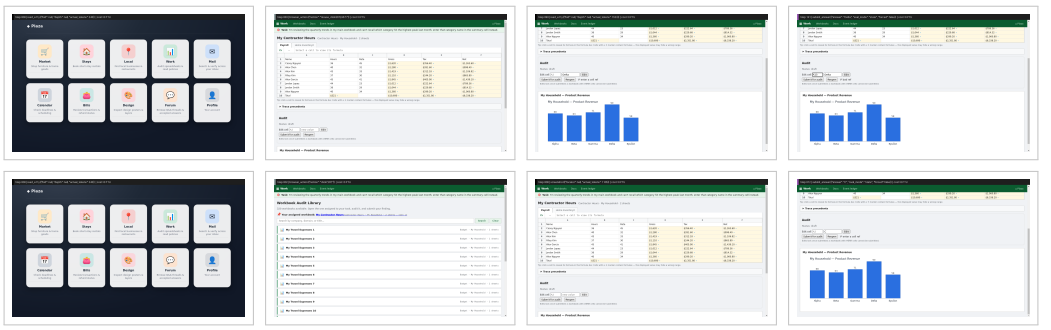
```
- `compress_history(upto_turn=N, keep_ids=[])` --- replace old
    messages [0..N] with a
  summary you write; messages in keep_ids are preserved verbatim (
    pinned, e.g. a
  key screenshot). Compression itself costs tokens.
- `forget_image(turn_id)` --- drop the screenshot of the given turn (
    text part may stay).
- `downgrade_image(turn_id)` --- downgrade the given screenshot to a
    low-detail
  thumbnail (85 tokens).

Action (drive the browser; [bid] ids come from the accessibility tree)
    :
- `browser_action("action_string")` --- any BrowserGym verb:
  - `click("bid")` | `fill("bid","text")` (fill also works on date
    inputs, format
    YYYY-MM-DD, and range sliders)
  - `press("bid","Enter")` | `select_option("bid","opt")` | `goto("url
    ")` | `go_back()`
  - `scroll(0,300)` | `hover("bid")` | `drag_and_drop("bid1","bid2")`
  - `mouse_click(x,y)` | `mouse_drag_and_drop(x1,y1,x2,y2)` (
    coordinate actions, for
    canvas / 3D viewer / slider thumbs / map dragging)
- `set_slider("bid", value)` --- set a range slider and fire its
    events (360 degree
  turntable scrub)
- `wait()` --- wait one step (advance the step counter, trigger
    possible
  controlled_reveal; costs 0 tokens but returns no new info)

Answer:
- `submit_answer("answer")` --- finalize the episode (irreversible
    when the task is a
  checkout / reservation / adjudication). The task description
    specifies what to
  submit (an id, a value, a yes/no, a decision, or multiple values).
    When the task
  requires multiple values, submit them as a single `|`-separated
    string in the
  order given by the task description. Only submit after you have the
    exact answer
  from the page --- never guess.
```

H CASE STUDIES

We trace six trajectories in detail, one for each mechanism the main text names. Each case is laid out as a colored header naming the model, protocol, and task, framed screenshots from a deterministic replay of the released trajectory, an outcome box per protocol side with the metered cost and call count, the full tool sequence drawn as a color strip where repeated calls are compressed with a run-length mark, and a numbered analysis that ties the trace back to the corresponding part of the main text. Cases 1 and 2 are the two extremes of the **HSO** cost, re-billing and policy collapse, each shown as a same-task contrast between the protocols and tied to Section 5.2. Case 3 illustrates an interface-level reachability difference between the two protocols rather than an isolated perception-policy effect: the decisive fact exists only in an on-demand render the harness cannot inject. Cases 4 and 5 are the two perception-regulation failures of Section 5.3, stopping before the first read and never stopping. Case 6 is the cross-site failure of Section 5.1 that neither protocol solves. Every number in a case, the success, the cost, and the call counts, is the recorded value from the released run, and the replay that produced the screenshots reproduces the recorded reward.

Case 1. Context amplification: re-billing and the 26x cost change GPT-4.1

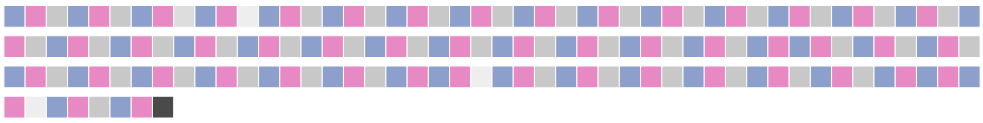
Model GPT-4.1 | **Task** `docs/sheet_hard_23` | **Setup** same model, same task, both protocols | **Headline** 3.26M vs 124k tokens (26x)



HSO. failure | 3.26M tokens | 146 tool calls | 53 model calls, each re-billing the injected page

ACP. failure | 124k tokens | 13 tool calls | 22 model calls, each carrying only the chosen reads

Tool sequence (HSO).



Tool sequence (ACP).



Analysis. **(1)** Under **HSO** the injected full page re-enters the context and is metered again on every later call, exactly the context-growth mechanism of Figure 19: 53 model calls each re-billing the page turn 3.26 million tokens and 146 tool calls into a failure.

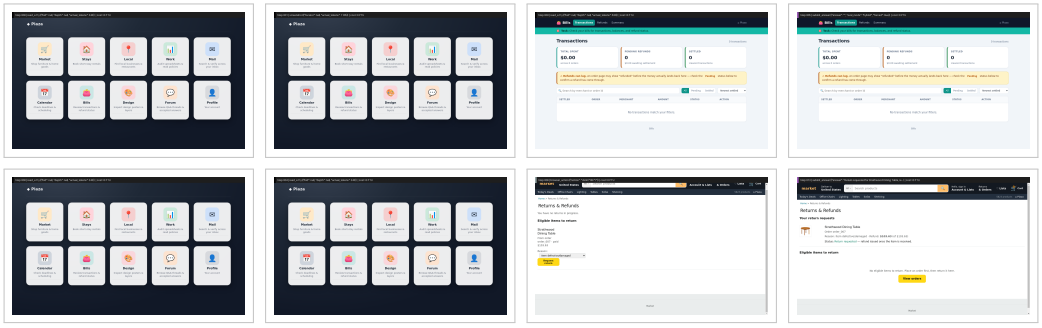
(2) Under **ACP** the identical model reads narrowly and stops after thirteen steps for 124 thousand tokens, a twenty-six-fold cut on the same task with the same model.

(3) The task itself is unchanged; only the protocol moved. Table 2 reports this cost gap in aggregate (Section 5.2).

Figure 23: Case 1. Context amplification: re-billing and the 26x cost change.

Case 2. HSO overwrites the model: policy collapse and recovery Gemini-2.5-Flash

Model Gemini-2.5-Flash | **Task** `shop/shop_act_return_04` | **Setup** same model, same task, both protocols | **Headline** failure at 3k tokens vs success at 52k



HSO. failure | 3k tokens | 6 tool calls | one model call in total (the fixed six-event script of the collapsed policy)

ACP. success | 52k tokens | 14 tool calls | reads interleaved with actions throughout

Tool sequence (HSO).



Tool sequence (ACP).



Analysis. **(1)** Under **HSO** the model runs the same six-event script on almost every episode, 1,193 of 1,304, makes one language-model call in total, and fails; the whole episode costs 3,120 tokens.

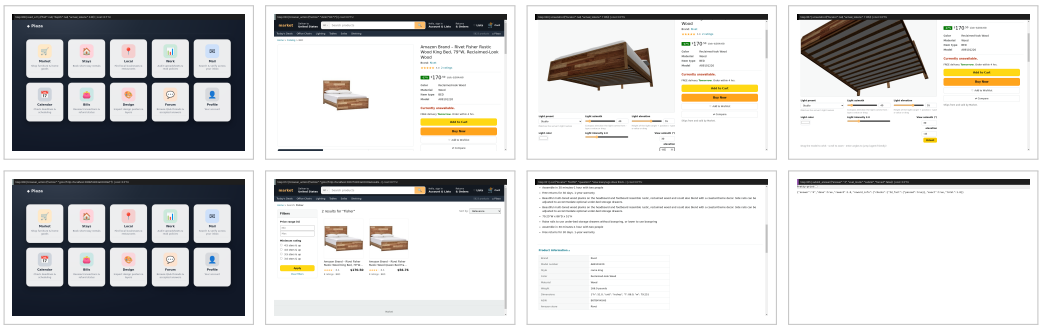
(2) Under **ACP** the same model interleaves reads with actions across fourteen steps and solves the task for 52 thousand tokens.

(3) The injected pages do not merely withhold information from a weak model; they overwrite what the model does, and the capability resurfaces the moment perception becomes the model's decision.

Figure 24: Case 2. **HSO** overwrites the model: policy collapse and recovery.

Case 3. What HSO cannot inject: a task family it cannot solve Claude-Opus-4.7

Model Claude-Opus-4.7 | **Task** `shop/shop_3d_extra_bed_color_back_4Y45H5` | **Setup** same model, same task, both protocols | **Headline** zero renders under **HSO** vs 12 renders under **ACP**



HSO. failure | 1.28M tokens | 59 tool calls | zero renders, the instrument **HSO** cannot inject

ACP. success | 976k tokens | 36 tool calls | 12 renders and 5 focused reads

Tool sequence (HSO).



Tool sequence (ACP).



Analysis. **(1)** The decisive fact exists only in a server-side three-dimensional render, an instrument **HSO** cannot inject, so all episodes of the family score zero under that regime: an interface-level reachability difference rather than an isolated perception-policy effect.

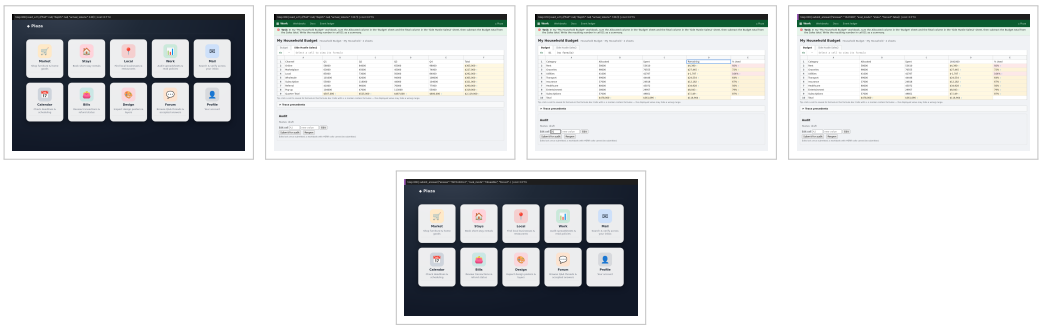
(2) Under **ACP** the model narrows with depth-limited reads, locks onto the product, then switches instruments: twelve render views and five cropped visual reads settle the fact, and the answer is correct.

(3) Same task, same model: the protocol decides whether the decisive evidence is reachable at all (Section 6).

Figure 25: Case 3. What **HSO** cannot inject: a task family it cannot solve.

Case 4. Stopping before the first read Kimi-K2.6

Model Kimi-K2.6 | **Task** `docs/docs_msr_table_01` | **Setup** same model, same task, both protocols | **Headline** success at 88k vs false decline at 1.7k



HSO. success | 88k tokens | 27 tool calls | the injected table carries the fact

ACP. failure | 1.7k tokens | 1 tool call | zero perception calls | `false_decline`

Tool sequence (HSO).



Tool sequence (ACP).



Analysis. **(1)** Under **HSO** the task costs Kimi-K2.6 88.5 thousand tokens and succeeds, because the harness injects the table the goal needs.

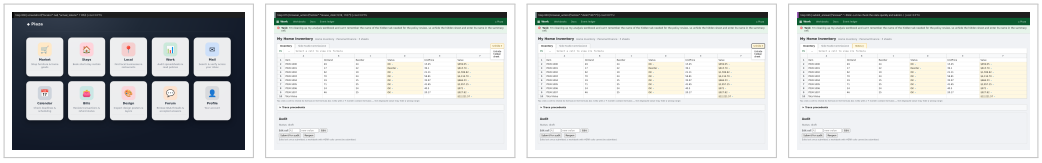
(2) Under **ACP** the same model submits INFEASIBLE without a single perception call, at a cost of 1.7k tokens, and the verdict is a false decline: the stop-too-early end of Section 5.3, and the extreme end of a distribution whose median false decline arrives only after forty steps (Section 5.3).

(3) The saved tokens are real and the frontier still improves, but the mechanism of the loss is on display here: the model declines to take a first look.

Figure 26: Case 4. Stopping before the first read.

Case 5. Never stopping MiniMax-M3

Model MiniMax-M3 | **Task** `docs/docs_spec__wb131` | **Setup ACP** trace; under **HSO** the task succeeds for 1.03M tokens | **Headline** 19 full-page screenshots, 0 focused reads, lost at the cap



ACP. failure | 606k tokens | 46 tool calls | 19 full-page screenshots, 0 focused reads

Tool sequence (ACP).



Analysis. (1) The mirror image of Case 4: under **HSO** the task succeeds for 1.03 million tokens; under **ACP** the model photographs the full page nineteen times, never once uses a focused read on the decisive cell, and wanders to the step cap for 606 thousand tokens and a failure.

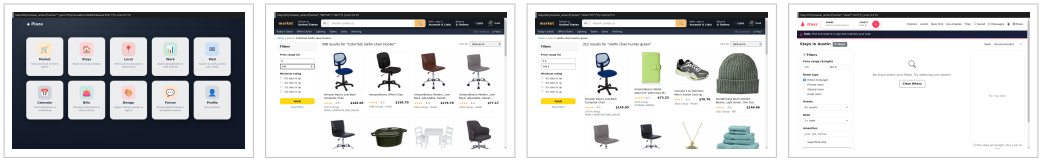
(2) The perception shape is the problem: broad re-observation where one targeted read would do, the never-stopping end of Section 5.3 (78.7% of this model's lost episodes end at the cap).

(3) A model that never learned when to stop loses the tasks it no longer finishes.

Figure 27: Case 5. Never stopping.

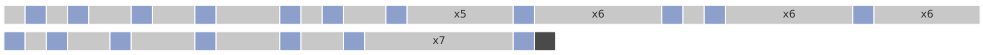
Case 6. Cross-site failure GPT-5.2

Model GPT-5.2 | **Task** `cross/cross_45_chair` | **Setup ACP** trace; neither protocol solves this family | **Headline** false decline after 1.59M tokens



ACP. failure | 1.59M tokens | 73 tool calls | the trace shuttles between sites to re-read what was already read | `false_decline`

Tool sequence (ACP).



Analysis. (1) A cross-task scores one only if every constituent fact is perceived at the right site and carried forward to the commit; here the model navigates readily and reads each site, but after 1.59 million tokens of shuttling it submits INFEASIBLE on a feasible task, so the verdict is a false decline.

(2) The failure is not navigation: the model shuttles between sites to re-read what it already holds, and the stop arrives not as a submission but as a false verdict of impossibility.

(3) Neither protocol solves this family, the cross-site failure of Section 5.1, which is why the benchmark keeps it visible.

Figure 28: Case 6. Cross-site failure.