SENSITIVITY IS NOT VALIDITY: LLM PLAN JUDGES PREDICT EXECUTION WORSE THAN OVERLAP

Anonymous authors
Paper under double-blind review

ABSTRACT

We ask whether plan evaluators predict what happens when the plan is executed – in AI2-THOR, under an oracle low-level controller that removes perception and actuation as causes of failure. Plan-level evaluation promises to separate an embodied agent's reasoning from its perception and control, but only if its scores predict something real. We meta-evaluate deterministic, symbolic, embedding-based and rubric-guided LLM-judge evaluators by how well each predicts execution of the *same* plans. Across $24{,}615$ rollouts of $1{,}641$ ALFRED-derived episodes, a plain overlap metric predicts execution better than EMBER – our own rubric judge, a contestant here rather than a proposal (primary pool, n=$6{,}564$: AUROC 0.890 vs. 0.786; preregistered difference -0.104, 95 % CI $[-0.125, -0.082]$). The deficit holds for all fifteen planners we executed, eleven of them in four families scoring the same episodes, with a mean of -0.114 over those eleven. One family is architecturally disjoint from the one we developed on, and one planner is multimodal, its image moving plans but not execution. A frontier judge given the same inputs still trails (0.857). The judge's usual defence – that it credits valid alternatives overlap penalizes – fails on natural output and holds only weakly under control: on execution-verified rewrites of gold plans it beats LCS by a preregistered $+0.102$, but by $+0.079$ once no rewrite names a new object, it is near chance on precondition violations a symbolic checker catches, and an order-free overlap metric beats it throughout. A second preregistered test finds no localization gain over a single overlap scalar. The ordering holds for decisions: choosing one of eleven planners' plans per instruction, overlap picks an executable plan $50.2\,\%$ of the time, the judge $36.0\,\%$. What the judge adds with a reference is small (exploratory): $+0.016$ AUROC over overlap, reordering plans below the top rather than changing which is chosen, and $+0.005$ over a fitted composite of cheap text features, the strongest scorer we have (0.947). Without a reference, a frontier judge is the best option we tested. An evaluator's validity has to be established for the decision it serves. We contribute the protocol, a validated executor, and these execution-anchored results.

1 INTRODUCTION

LLM judges have become the default measuring instrument wherever outputs are open-ended, and the benchmarks that certify them mostly stay inside language: a judge is scored against human preference (Zheng et al., 2023) or another model's verdict, which is model-based evidence for a model-based metric. A verifiable key opens that loop (Tan et al., 2025); open-ended outputs rarely have one. That leaves the question a metric actually has to answer untouched. Does the score predict anything that happens outside the evaluation itself? In this paper the reference is neither a judge nor a preference. It is execution: we run the plan and see.

Embodied planning is one of the few places where that question has a clean answer: a predicted plan can be *executed*, and execution succeeds or fails without consulting any model. That behavioural criterion lets us ask of a judge what measurement theory asks of any instrument (Jacobs & Wallach, 2021) – not whether it is sensitive, but whether it is *valid* for the construct it claims to measure. We use embodied planning as the testbed, not the topic.

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

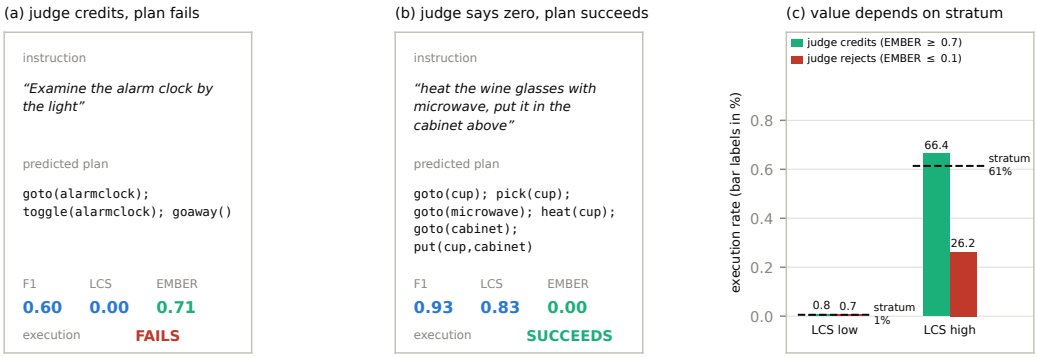

Figure 1: When the rubric judge and a plain overlap metric disagree, execution arbitrates, and sides with overlap both ways. **(a)** The judge awards 0.71 to a plan that toggles the clock instead of the lamp and ends in an undefined verb; its objects resolve in the scene, so no synonym layer would save it; **(b)** 0.00 to a plan that completes the task. **(c)** *Within* an overlap stratum (dashed = that stratum's own rate) the judge adds no lift where the reference predicts failure; exemplars were selected with their outcome known (Sec. 5.2).

Today's plan evaluators fall short in complementary ways: overlap metrics penalize valid alternatives and miss semantic errors that preserve surface tokens, the complaint that drove machine translation past BLEU (Callison-Burch et al., 2006), while rubric-based LLM judges handle alternatives but import self-preference (Panickssery et al., 2024) and order effects strong enough to flip a verdict (Wang et al., 2024). Diagnostic benchmarks (Sun et al., 2026; Li et al., 2024) broaden *coverage* yet rank *models*, not *metrics*.

Fig. 1 previews what ranking models instead of metrics costs. Our starting point is the distinction those benchmarks leave implicit: **a metric that separates more model pairs is not thereby a more correct metric**. Sensitivity is not validity.

What this paper is not. It proposes no metric. EMBER (Episode-level Metric for Behavioral Evaluation of Reasoning) is *our own* rubric-guided LLM judge over a shared observation–instruction–action (O–I–A) interface, entered as a stand-in for the form of rubric judging (Liu et al., 2023c; Kim et al., 2024a) rather than for any published system: one contestant in an execution-grounded *meta-evaluation*, not a proposal. What we offer is the validation protocol, and its negative result is the thesis rather than a way-station to a better score.

Our central finding is negative. Across $1{,}641$ ALFRED-derived episodes executed in AI2-THOR under an *oracle* low-level controller, execution success is dominated by a plan's action-sequence *structure*, which cheap overlap metrics capture directly: a one-line LCS score predicts execution at AUROC 0.890 where the rubric judge reaches 0.786, a preregistered paired deficit of -0.104 (95 % CI $[-0.125, -0.082]$). Every deterministic metric that beats it needs a gold plan, which a deployed system lacks; without one a capable judge given the task's objects is the best scorer we tested, and a small one selects worse than a symbolic check.

The oracle controller is the design, not a shortcut. A plan score claims to measure plan quality, so the criterion must isolate plan quality from everything else that can make an episode fail. Handing perception and actuation to an oracle does that, and it isolates the axis that motivates a semantic judge: what is left to explain a failure is the plan's content and the executor's fixed rules, so a rubric that understands intent has its clearest chance there. We test the instrument rather than assume it. Physical refusals show no detectable bias against non-gold receptacles (0.60, CI $[0.07, 1.59]$), but noun resolution is rigid, and dropping every referent failure a synonym layer would have caught still leaves -0.095 (Appendices E.1 and E.2). The criterion is also structural by construction and so favours structural metrics; Sec. 5.1 tests whether that alone explains the result. It does not, nor when the planner is given the scene image (Sec. 5.6). We attempted the non-oracle version and abandoned it at a preregistered gate (Sec. 6).

Our contributions are:

1. **Why execution is the criterion, and the protocol that makes it usable.** A shared O–I–A interface with converters from ALFRED and LIBERO, and a meta-evaluation anchoring plan evaluators to symbolic and simulator criteria on the same trajectory clusters, including a validated oracle AI2-THOR executor ($93/100$ gold-plan success, Appendix E).
2. **Does the judge lose? One preregistered reversal and one preregistered failure to win.** Under an oracle controller, deterministic overlap metrics predict task execution clearly better than the rubric LLM judge; and the judge's components do not reach the registered localization gain over the best single scalar (-0.002 against a required $+0.08$), while trailing the five scalars combined on every class we label. Both are cluster-bootstrap paired tests against thresholds fixed in advance, and the first holds for all eleven planners in four families ($4{+}4{+}1{+}2$, Sec. 4), one of them architecturally disjoint from the family we developed on, where the deficit is *larger* (-0.146).
3. **When does it help?** Two preregistered perturbation studies: the judge credits constructed alternatives LCS penalizes ($+0.102$), but mostly by noticing unfamiliar objects ($+0.079$ without them, near chance on precondition violations), and set overlap beats it. Exploratory: its $+0.016$ AUROC over overlap reorders plans below the top and changes almost no choice, and over a cheap fitted composite (0.947) it adds $+0.005$. Without a reference, a frontier judge with the task's objects is the best scorer we tested (0.815).
4. **What to use** (Sec. 5.9): with a reference, overlap as the default and a fitted cheap composite where execution labels exist; without one, a capable judge given the task's objects, disjoint in family from the planners.
5. **Audits, and two cautions we pay for.** We audit same-family self-preference, a compositional gap that fails to replicate across families (retracting an emergence claim), and a small-judge hallucination on empty plans. Two cautions cost us: reconstructing reference-free scores from gold-conditioned verdicts overstates the advantage $2.4\times$, and our selective hybrid missed its call-reduction threshold, so we report it failed.

2 Related Work

Table 4 in Appendix B places this paper among the benchmarks nearest to it, on the two axes that separate them: whether the thing judged is a model or a metric, and where the reference comes from.

Diagnostic planning benchmarks. The Agent Planning Benchmark (Sun et al., 2026) scores plans with a rubric LLM judge checked for consistency, not against execution, and benchmarks *models* on non-embodied tasks; the Embodied Agent Interface (Li et al., 2024), our closest neighbour, unifies four planning abilities with simulator and symbolic anchors yet runs no metric-vs-metric contest, and EmbodiedBench (Yang et al., 2025) and LoTa-Bench (Choi et al., 2024) broaden coverage, which is orthogonal to metric validity.

Symbolic plan validation. VAL (Howey et al., 2004) verifies plans against formal semantics, assuming a full formal domain, and PlanBench (Valmeekam et al., 2023) uses that machinery to benchmark planners. ViPlan (Merler et al., 2026) weighs grounding those predicates with a vision-language model against letting it plan directly, and its household domain favours the latter; natural-language instructions admit many valid plans (two distinct successful action sequences per trajectory at the median, Appendix I) and only a rule-checkable fragment: our checker reads the plan, not the room (Appendix H). We treat symbolic validity as one anchor and one contestant.

Meta-evaluating metrics. Asking which metric is trustworthy rather than which model is best is routine in machine translation (Mathur et al., 2020; Rei et al., 2020) and reward modelling (Lambert et al., 2025); our reference-based/reference-free split is that field's quality-estimation problem (Specia et al., 2018), transposed to plans.

LLM-as-a-judge, and execution as its criterion. Zheng et al. (2023) validated chat judges against human preference and exposed position, verbosity and self-enhancement biases; order alone can flip a verdict (Wang et al., 2024), evaluators favour their own generations (Panickssery et al., 2024), and rubric-guided judging (Liu et al., 2023c; Kim et al., 2024a) is the form EMBER shares; we run the second, in its Prometheus 2 form (Kim et al., 2024b) (Appendix S.1). Nearest to our thesis, Norman et al. (2026) separate a judge's *reliability* from its validity over $541{,}000$ judgments and Chen et al. (2026) find judges unstable across scenarios; both anchor validity in human-aligned

quality, where we anchor it in execution. Reward-model evaluation asks our question of a different proxy: PPE (Frick et al., 2025) and RewardBench 2 (Malik et al., 2026) test whether benchmark accuracy predicts downstream use, and RM-Bench (Liu et al., 2025) separates content from style with controlled pairs, as CheckList (Ribeiro et al., 2020) does for NLP behaviour; we ask it of plan evaluators, against execution, for two decisions that rank differently. The closest prior line evaluates code against execution: EvalPlus (Liu et al., 2023b) strengthens the tests a generator is scored against, while ICE-Score (Zhuo, 2024) and CodeJudge (Tong & Zhang, 2024) are themselves reference-free LLM judges of code, validated against functional correctness rather than a reference. A test suite is also met by many programs, so what embodied plans add is not that; it is an executor whose failures we can partition (grounding, precondition, goal), an oracle that removes perception and control, and the same evaluators scored against symbolic validity and execution on the same plans. We contribute that protocol, the finding it yields, and the human anchor as a preregistered design. ALFRED (Shridhar et al., 2020) and LIBERO (Liu et al., 2023a) supply the tasks, zero-shot LLM planners (Huang et al., 2022) the generation method; we hold planners (Qwen, 2024) fixed and meta-evaluate the *evaluators*.

3 Protocol and Evaluators

O–I–A interface. Each episode is an observation–instruction–action tuple: an initial observation, a natural-language instruction, and a predicted *plan* – typed actions in a ten-verb DSL (`goto`, `pick`, `put`, `open`, `close`, `toggle`, `slice`, `heat`, `cool`, `clean`). Converters map ALFRED `high_pddl` and LIBERO goal predicates into this common form, so every evaluator scores the same plan objects.

Evaluators under test. We compare plan exact match (EM), action token-set F1, an order-sensitive longest-common-subsequence (LCS) plan score, a subgoal-prefix score, plan length error, a rule-based symbolic validity check, a sentence-embedding cosine (all-MiniLM-L6-v2) between predicted and gold plans, and EMBER, a rubric LLM judge scoring coverage c, logical validity v, semantic equivalence q and redundancy $r \in \{0, 1, 2\}$ into $s = c \cdot v \cdot (0.7\,q + 0.3\,(1 - r/2))$, with c and v as hard multiplicative preconditions. We report r as *efficiency*, $e = 1 - r/2$, so all four read alike. The judge is given the instruction, the task's declared object and receptacle arguments (two or three items from ALFRED's `pddl_params`, not a scene inventory), the action DSL and its constraint list, the predicted plan and, in the reference-conditioned setting, the gold plan; Appendix D prints it verbatim and Appendix C gives every formula, domain and decoding setting. A selective *hybrid* skips the judge when the symbolic checker already proves a plan invalid.

Two anchors. Each evaluator is scored against criteria it did not produce. (i) *Symbolic*: a rule checker of DSL preconditions. (ii) *Execution*: each plan runs in AI2-THOR under an oracle low-level controller that resolves DSL nouns to ground-truth object identities and performs oracle navigation, so execution isolates plan structure from perception and motor control; the outcome is the ALFRED goal-condition checker. A third, preferential anchor – blinded human judgment – is designed and released as a preregistered study but is not run here (Appendix W). The executor is our measuring instrument, so it is validated first: replaying gold DSL plans succeeds on $93/100$ stratified trajectories (mean goal-condition fraction 0.952), clearing the preregistered 0.85 threshold, and on $88.2\,\%$ of all $1{,}641$ episodes when replayed as the perturbation study's control (Appendix L.4): the oracle removes perception and actuation as causes, not every non-plan failure.

4 Experimental Setup

Data. $1{,}641$ ALFRED-derived episodes in 506 trajectory clusters; the ~ 3 paraphrases of one trajectory share a gold plan, so episodes are not independent and every test is at the trajectory level. LIBERO contributes 650 episodes as a protocol generality check without an execution anchor (Appendix Y). **Planners.** Fifteen executed: eleven in four families – four Qwen2.5 (0.5B–7B), four Qwen3 (0.6B–8B), Mistral-7B-v0.3, and multimodal Qwen2.5-VL-7B with and without its image – plus the VLM's two no-object arms and two scripted skeletons (oracle-grounded and language-conditioned). Every deficit claim is over the eleven, the four others being ablations; all fifteen are negative. **Judges.** Six spanning 0.429–0.857 standalone execution AUROC: Qwen2.5 at 1.5B/3B/7B/32B, cross-family Mistral-7B-v0.3, and hosted gpt-5.1 (with and without the task's

Table 1: Execution prediction under an oracle controller (the four Qwen2.5 planners, n=6,564, execution rate 0.204), on two axes: *ranking* (AUROC/AUPRC) and *calibration* (Brier, lower better; raw, and after out-of-fold Platt recalibration). Overlap metrics beat the rubric judge on ranking. Bold: best per column of these six; Table 42 adds the two Sec. 3 names that are not here. Fig. 8 plots the two axes against each other.

	Ranking		Brier ↓	
Evaluator	AUROC	AUPRC	raw	recal.
Plan order (LCS)	**0.890**	**0.639**	**0.171**	**0.103**
Action F1 (token-set)	0.866	0.514	0.467	0.137
Selective hybrid	0.831	0.528	0.194	0.120
Embedding cosine	0.825	0.530	0.620	0.156
EMBER (7B judge)	0.786	0.475	0.245	0.131
Exact match (EM)	0.543	0.271	0.187	0.152
base-rate predictor	—	—	*0.162*	*0.162*

objects); local judges run at temperature 0 and empty plans are scored deterministically. **Statistics.** Intervals use a trajectory-cluster bootstrap (10,000 resamples for the preregistered tests, 2,000–5,000 for the later exploratory ones), paired over the same clusters; Holm–Bonferroni is applied to the ten discrimination pairs – not to the four preregistered claims as a family, which matters only for claim 3's noninferiority half – and every other interval is unadjusted – over 150 in the appendices alone, so no single exploratory interval is a controlled test. A post-hoc split-half replication choosing every threshold and weight on one half carries the deficit's sign onto all four held-out halves, the scene-disjoint split included (-0.093 to -0.120; Appendix A). Thresholds and decision rules were frozen in a preregistration before outcomes were observed; both amendments, the failed gates and the full setup are in Appendix A.

5 Results

5.1 Execution meta-evaluation: overlap beats the judge

We executed fifteen planners' predicted plans on the full episode set ($15 \times 1{,}641 = 24{,}615$ oracle rollouts, each allowed two attempts), spanning the four families of Sec. 5.6; success rises monotonically with scale within Qwen2.5 (7.7 % to 32.2 %), neither at floor nor ceiling. Table 1 reports, for the four Qwen2.5 planners (n=6,564), each evaluator's AUROC as a predictor of binary task execution. The order-sensitive overlap metric wins (LCS 0.890), then action F1 (0.866); the rubric judge trails at 0.786, below even the embedding baseline (0.825) and the hybrid (0.831). The preregistered paired test – EMBER minus the best non-LLM baseline, with a trajectory-cluster bootstrap – is -0.104 (95 % CI $[-0.125, -0.082]$), failing the preregistered criterion ($\geq +0.05$, CI lower bound > 0) in the *opposite* direction. The preregistration named AUROC *or* average precision; on the second the deficit is larger, -0.164 (CI $[-0.205, -0.122]$), which at a 0.204 base rate is the quantity a threshold-setter would read. Pooling in the oracle-grounded skeleton enlarges it (-0.125), and on the skeleton's own plans LCS predicts execution perfectly – a property of a planner that emits one structure, which is why the skeletons are excluded from every deficit claim.

Why does a richer semantic score lose to token overlap? The environment supplies perception, grounding and motor execution, so a plan succeeds when its action sequence is structurally close to a workable one – exactly what LCS and F1 measure. Close, though, is not identical: only 117 of 6,564 predicted plans reproduce the gold plan exactly and 115 of those succeed, 8.6 % of the 1,338 successes; the median successful plan scores 0.667 on LCS with a quarter at or below 0.5, and trajectories that succeed at all do so by two distinct action sequences at the median (Appendix I). Execution is not the gold plan rescored. Nor is the deficit an artifact of mechanical failures the plan text cannot speak to: on the episodes where every action executed and the goal check alone decided the outcome (n=3,116; a slice selected on the outcome, which removes exactly the failures overlap is expected to win) overlap still leads, 0.835 vs. 0.770 (-0.065, CI $[-0.099, -0.030]$), as on the 73 % of plans that are well-formed and symbolically valid (-0.094, CI $[-0.121, -0.068]$; Appendix H). Nor is it the rubric: asked directly for the probability the plan executes, the judge

Table 2: The deficit persists under judge choice, planner family, the score's shape and the rubric's wording. Each row recomputes EMBER−LCS execution AUROC under one decomposition (family rows pool the planners in parentheses, 1,641 each); every interval excludes zero. Family rows use 4,000 resamples against the headline's 10,000. Per-planner deltas: Fig. 2a.

	Variant	EMBER	Δ vs. LCS (95 % CI)
Judge	7B (primary)	0.786	-0.104 $[-0.125, -0.082]$
	32B (strongest)	0.810	-0.079 $[-0.097, -0.062]$
	Mistral (cross)	0.694	-0.195 $[-0.213, -0.177]$
	gpt-5.1 (frontier)	0.857	-0.033 $[-0.047, -0.019]$
Family	Qwen2.5 (4)	0.786	-0.104 $[-0.125, -0.082]$
	Qwen3 (4)	0.782	-0.104 $[-0.129, -0.080]$
	Mistral (1)	0.720	-0.146 $[-0.179, -0.114]$
	Qwen2.5-VL (2)	0.723	-0.125 $[-0.162, -0.087]$
Score form	multiplicative	0.786	-0.104 $[-0.125, -0.082]$
	additive	0.812	-0.077 $[-0.097, -0.058]$
Rubric	wording: checklist	0.607	-0.283 $[-0.309, -0.254]$
	wording: free-form	0.673	-0.217 $[-0.240, -0.194]$

reaches 0.619 reference-free (against its rubric's re-elicited 0.710) and 0.662 with gold (against 0.786; Appendix Q). The oracle controller bounds it (Sec. 6).

A pooled number invites four objections – one weak judge, pooled planners, the score's shape, one rubric – so we decompose (Table 2); on the last, a *published* judge, Prometheus 2 (Kim et al., 2024b), trails LCS by -0.133 (Appendix S.1). The deficit persists under every tabulated judge, within every planner, and under the non-gating additive variant (0.812; -0.077, CI $[-0.097, -0.058]$), so the 25.1 % of scores pinned at zero by the multiplicative gate explain part of it at most. The judge rows shrink monotonically with judge capability without reaching zero, but that sequence restates rather than tests: LCS is 0.890 in every row, so each deficit is that judge's own standalone AUROC minus 0.890. No judge we could reach closes the deficit, and Sec. 5.5 tests capability on quantities that are not restatements. A fourth objection, the rubric's *wording*, that decomposition cannot reach: re-eliciting the same judge under a checklist and under a free-form critique leaves both *further* behind overlap (-0.283, -0.217, intervals excluding zero), so the rubric we report is the strongest of the three (Appendix V). A fifth, score *resolution* – AUROC halves ties and the judge takes 13 values to LCS's 61 – does not reach it either: LCS coarsened to four quantile levels, below the judge's own resolution, still leads by -0.082 (CI $[-0.104, -0.060]$; Appendix I).

Added to overlap the judge is a small *complement* (exploratory): an out-of-fold combination reaches AUROC 0.906 against LCS's 0.890 ($+0.016$, CI $[+0.010, +0.022]$), but a cheap text-feature composite already carries most of it (Appendix L.3). Its gain lives where overlap is already confident (Sec. 5.2): a nested gate chosen on training folds alone keeps it at 53.2 % of the calls ($+0.017$, CI $[+0.011, +0.023]$), and the same budget spent on the low-overlap side recovers nothing. It is not detected in Mistral (0.000, CI $[-0.004, +0.004]$), where the deficit is *largest*, and a frontier judge does not detectably enlarge it (Appendix L).

5.2 When they disagree, execution sides with overlap

The remaining analyses are exploratory. An aggregate AUROC gap says one evaluator ranks better, not who is right about a particular plan, so we isolated the disagreements – always *within* an overlap stratum, since a subgroup selected for low LCS already executes far less often (Fig. 1c).

The judge's value is concentrated rather than absent. Among plans overlap condemns (LCS ≤ 0.3, n=1,756, stratum execution 0.6 %) it adds nothing we can detect: the 259 plans it credits (score ≥ 0.7) yield 2 executions, the 893 it rejects (≤ 0.1) 6, too few to establish absence. Properly sized, on the 3,089 plans a symbolic checker accepts but overlap rates at or below its median, the judge is not detectably better (-0.024, CI $[-0.060, +0.010]$; Appendix L). That set is not a set of valid alternatives – the checker never sees the scene, so it passes plans naming absent objects too – which is why Sec. 5.3 tests alternatives directly. Among plans overlap rates highly (LCS ≥ 0.6, n=1,629, stratum execution 61.4 %) it discriminates sharply: the 960 it accepts execute 66.4 % of the time

against $26.2\,\%$ for the 84 it rejects, a gap of $+0.402$ (95 % CI $[+0.274, +0.512]$) resting on that arm of 84. These four cut-offs were chosen, not preregistered; sweeping them over 20 distinct settings keeps the gap positive in every one, with a median of $+0.438$ (Appendix L.5). The judge earns its keep by catching plans that look structurally right and are not – the failure mode overlap cannot see, and the reason its aggregate AUROC still trails: the stratum where it helps is the one overlap has already sorted.

We also read the disagreements the other way, classifying *why* the judge went wrong on each: half of the plans it credits name an object the task never declared (Appendix M). Giving the judge the room does not help: with every object type in the scene added to its prompt, the same judge trails overlap by *more* (-0.156, CI $[-0.183, -0.129]$), credits twice as many plans overlap condemns, and half of those still name an object outside the list it was handed (Appendix N). An object-type inventory does not close the loss.

5.3 The judge credits constructed alternatives, weakly

Natural output cannot separate structure from meaning, so we rewrote every gold plan under fixed rules – two keeping its meaning while lowering overlap, four breaking it while keeping overlap – executed all $9{,}579$ rewrites and $1{,}641$ gold controls, and registered the test first (Appendix L.4). Over $8{,}166$ pairs of an executing rewrite and a failing near-miss of one episode, the judge ranks the rewrite higher $74.6\,\%$ of the time and LCS $64.4\,\%$ ($+0.102$, CI $[+0.089, +0.116]$): by the registered rule the defence holds. A second registration removed a confound: the detours and distractors named objects outside the declared list, which the judge's prompt tells it to distrust. Without new objects the lead falls to $57.9\,\%$ against $50.0\,\%$ ($+0.079$, CI $[+0.060, +0.097]$), near chance ($54.9\,\%$) on a genuine precondition violation that a symbolic check catches every time. Set overlap beats the judge in both studies (action F1 $87.7\,\%$, $68.4\,\%$).

5.4 The comparison a deployed system faces

Table 1 hides an asymmetry: every evaluator that beats the judge needs a *gold plan*, which a robot receiving a new instruction has not got, while only the rubric's equivalence term refers to a gold plan. Restricting the comparison to evaluators computable without one asks the question a deployment faces, inheriting the hazard that such evaluators favour systems they resemble (Deutsch et al., 2022).

We re-judged all $6{,}564$ episodes with the gold plan withheld, scoring with the reference-free composite $c{\cdot}v{\cdot}e$ – the rubric minus equivalence, which has nothing to compare against absent a reference. So scored the judge predicts execution at AUROC 0.710, against 0.655 for a symbolic validity check, the best single alternative (Appendix O): an advantage of $+0.055$ (95 % CI $[+0.037, +0.072]$), modest and well below what a gold plan buys (0.890). Cutting the judge to the check's two values, so ties penalise both, leaves $+0.037$ (Appendix O.1). No single free feature reading the judge's inputs does better than symbolic validity (a declared-object grounding check 0.640, instruction–plan similarity 0.538). Pooling all nine is another matter. Out of fold they reach 0.789, above the 7B judge (-0.079, CI $[-0.099, -0.058]$) but below the frontier judge given the same inputs (-0.027, CI $[-0.047, -0.005]$). It is fitted on execution labels; frozen, it transfers to three unseen planner families (0.741–0.818, Appendix P).

What matters is where the components came from. Reconstructing reference-free scores from gold-conditioned verdicts inflates the advantage $2.4\times$ ($+0.132$ against a re-elicited $+0.055$), so reference-free judging must be re-elicited, not reconstructed (Appendix S).

5.5 Capability fits better than family

This section is exploratory. We expected self-preference, but Mistral is also the *weaker* judge (0.694 against 0.786 and 0.810), so we added weak *same*-family judges (Qwen2.5-1.5B, 3B) in its capability band. They behave like Mistral (Fig. 2b; Appendix S): across six judges the gated gain tracks standalone capability ($r{=}0.83$, permutation $p{=}0.028$; descriptive, since both axes share the judgments), and a family term is not detectable with capability held fixed – three residual degrees of freedom, so this cannot show family is irrelevant, and it places no threshold.

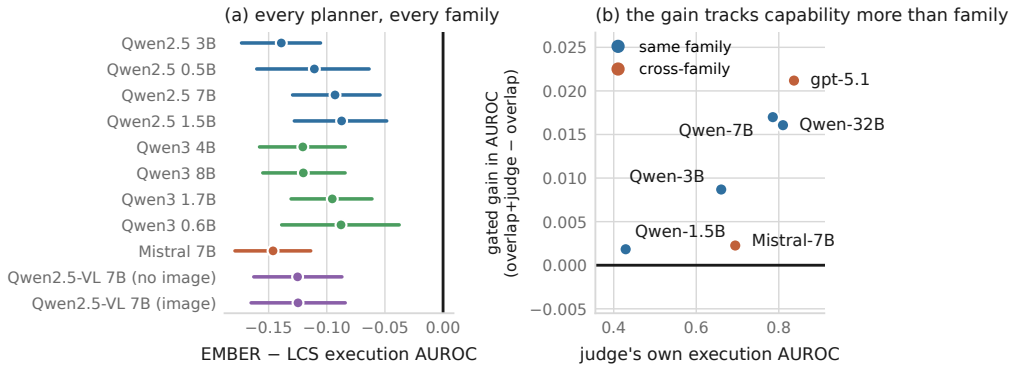

Figure 2: **(a)** The judge trails overlap in all eleven planners, across all four families; every interval lies left of zero (95 % cluster CIs). **(b)** Across six judges the gated gain rises with the judge's own execution AUROC regardless of family (gpt-5.1 here without the object list).

We then ran a hosted frontier judge (gpt-5.1), cross-family with respect to every planner. Its first arms lacked the declared-object list the local judges receive, so we reran it with the list under a registration fixed beforehand (Appendix C). The deficit narrows but survives: 0.857 against LCS's 0.890, -0.033 (CI $[-0.047, -0.019]$; 0.837 without the list), and it still selects worse than overlap (-0.011). Reference-free the ordering reverses: it reaches 0.815 (0.774 without the list) against symbolic's 0.655, $+0.160$ (CI $[+0.141, +0.179]$), above the fitted nine-signal composite ($+0.027$, CI $[+0.005, +0.047]$), and it selects better than the check ($+0.081$). A contamination probe reproduced 0 of 150 gold plans (Appendix R).

5.6 The deficit survives three further planner families

Our fourth preregistered claim asked for generality across two planner families *and* two domains; the planner half now holds across four. Under the identical protocol we executed a second generation (Qwen3 at 0.6B/1.7B/4B/8B), an architecturally disjoint family (Mistral-7B-v0.3, whose primary judge is therefore cross-family), and a *multimodal* planner (Qwen2.5-VL-7B, with and without its image, which moves a third of its plans but not its execution rate, Appendix H). *Every planner gives a negative delta whose interval excludes zero* (Fig. 2a). Pooled by family (Table 2), Qwen3 reproduces Qwen2.5's deficit to three decimals, the deficit is *larger* in the disjoint family, the eleven deltas pool to -0.114 (CI $[-0.137, -0.089]$), and re-scoring Mistral with the 32B judge holds the sign (Appendix J). The domain half does not hold: execution is ALFRED-only.

5.7 Discrimination is not validity

A second defence of the judge is that it separates planner pairs an exact-match success rate cannot. It does: among five planners whose exact-match rates span a 2.7-point band (their execution rates span 7.7–32.2 %), EMBER separates $9/10$ pairs under clustering and Holm correction. But so do action F1 and plan-length error ($10/10$) and LCS ($9/10$): plan-length error separates *every* pair yet ranks fifth of seven on execution, below every overlap metric and the judge (Appendix V.1).

5.8 Nor do the components localize failure

A third defence of a rubric judge is diagnostic rather than predictive: even if the scalar score ranks poorly, its *components* should reveal *why* a plan failed. We preregistered this separately and tested it against the execution logs. A rule-based parser of the executor's error, fixed before any model was fitted, labels $4{,}049$ of $5{,}226$ failed rollouts into five classes, the largest being `goal_not_met` (n=1,778, every action succeeding and the plan doing the wrong thing), and excludes the rest rather than forcing them into a class (Appendix X); a multinomial classifier with cluster-grouped cross-validation then predicts the class from each feature set.

The components do not localize better: macro-F1 0.286 for {coverage, validity, equivalence, efficiency} versus 0.508 for the five deterministic scalars together (majority baseline 0.122; -0.221,

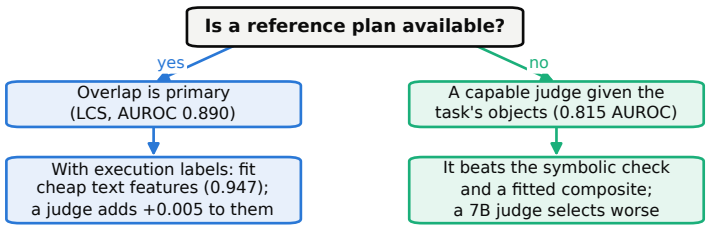

Figure 3: The recommendations as one decision. Fitted scorers need execution labels; the frontier judge's numbers are with the task's objects.

95 % CI $[-0.248, -0.194]$). The registered endpoint, components against the best *single* scalar (LCS alone, 0.288), is -0.002 (CI $[-0.024, +0.020]$): four rubric numbers localize no better than one overlap number, and the claim fails. Two length features carry most of the combined baseline (0.336), but two overlap features without lengths (0.314) also beat the components, and the five scalars win on every class, from `goal_not_met` (0.691 vs. 0.606) to `physical` (0.399 vs. 0.000). Caveats: four deterministic features and the equivalence term need the gold plan, two classes are nearly definitional in surface form (35.8 % of labels), and this asks what survives in four scalars, not whether a judge prompted to diagnose could do better.

5.9 What to use, and when

Whether a reference exists decides most of what follows (Fig. 3), whether execution labels exist decides the rest, and only the first clause of (i) rests on a preregistered test. (i) *With a reference*, make overlap the primary signal: selecting one of eleven planners' plans per instruction, overlap picks an executable plan 50.2 % of the time and the judge 36.0 % (-0.141, CI $[-0.161, -0.123]$), and at no budget does a judge alone detectably beat it. Where execution labels exist, fit a cheap composite of text features on them (0.947, top-1 $+0.023$ over LCS); adding the judge to it gains $+0.005$ AUROC and nothing detectable on either decision (Appendices L.2 and L.3). (ii) Recalibrate on execution before any threshold (Brier $0.171 \rightarrow 0.103$). (iii) Keep any judge disjoint in family from the planners and fixed for the benchmark's lifetime. (iv) *Without a reference*, use a capable judge given the task's objects, re-elicited rather than reconstructed: it beats the symbolic check and a fitted nine-signal composite; a 7B judge selects worse than the check. (v) A judge added to overlap can be gated to 53 % of the calls ($+0.017$).

6 Limitations

A *non-oracle* controller, the test that would remove our main scope bound, failed a preregistered instrument gate: three builds reached 0.68 gold-plan success against a required 0.85, and we stopped rather than tune to a pass (Appendix F). Our result is therefore specific to the *oracle* controller: giving the *planner* perception leaves the deficit intact; it is the *controller*'s perception that matters.

Three bounds matter most. The reference-free $+0.055$ and $+0.160$ are orderings under oracle control, not deployment benefits (Appendix O). We test one rubric across six base models, two alternative wordings (Appendix V), a direct prompt and one published judge (Appendix S.1), not every way to build a judge. And execution is ALFRED-only, whose tasks are near-linear, so whether overlap keeps its lead where subgoals commute is open. Further: the human anchor is released as a design, not collected, so it is a resource rather than evidence (Appendix W), and other languages, DSLs and prompts are untested.

7 Conclusion

On ALFRED under an oracle controller, a rubric LLM judge predicts execution worse than a one-line overlap metric for all fifteen planners, as does a published judge on the primary pool, and localizes failure no better. Where a reference exists, make overlap the default, and add a judge only for a decision where it is shown to pay.

REPRODUCIBILITY STATEMENT

Code, the preregistration with its amendments and the failed gates, all 24,615 planner rollouts and 14,235 perturbation rollouts, all 148,663 stored judgments, and the decision log are available at `https://anonymous.4open.science/r/ember-meta-evaluation/`. Every number in the paper traces through a single facts sheet to a stored artifact, and the 54 GPU-free analysis scripts the verifier covers regenerate their 67 artifact files byte-identically; the seven exceptions are listed in `REPRODUCE.en.md` and in the verifier script. The headline was additionally re-derived with an independent implementation using none of our code (Appendix Z). The human-evaluation answer keys are withheld because releasing them would un-blind the study we release as runnable.

ETHICS STATEMENT

This work evaluates plan evaluators in simulation; no human-subject data were collected for any result reported here. The human-agreement study is released as a preregistered, runnable design whose answer keys are withheld so that running it does not un-blind it. All local planners and judges are publicly available models under permissive open licences (Qwen2.5, Qwen3 and Mistral-7B-v0.3 under Apache-2.0), as are ALFRED and LIBERO (MIT); one hosted frontier judge is used under its provider's terms; no personal data enters any prompt. The central finding is a caution about instruments: we see no direct path from it to harm beyond the general risk, which we argue against, of certifying an evaluator by its sensitivity alone.

AI USE STATEMENT

We used a generative AI tool (Anthropic's Claude) for tasks with required disclosure: proposing exploratory hypotheses after the preregistered tests were adjudicated, designing analysis methodology, implementing methods, reformatting datasets, and drafting interpretations. We did not use it to generate synthetic data, to formulate or prove mathematical claims, or for translation; the remaining required-disclosure tasks do not apply. The original preregistered claims and thresholds were set by the authors before any outcome was observed; each later amendment records its own date and what had been observed by then (Appendix A), and analyses outside the registration are labelled exploratory. We also used it for recommended-disclosure tasks: figures, code, drafting and editing text, and finding and formatting references. We have reviewed all AI-assisted work: every number traces to a stored artifact that its script regenerates byte-identically, the headline was re-derived with an independent implementation, and all thirty references were checked against their primary sources. That check matters: it caught two fabricated author names and a number of venue and metadata errors, and a separate pass over how each work is *described* caught seven places where the sentence did not match what the cited paper does, including a novelty claim two of our own citations falsify. Every one is corrected here. We take responsibility for the final content of this work.

REFERENCES

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 249–256, Trento, Italy, 2006. Association for Computational Linguistics.

Junjie Chen, Yuxi Dong, Haitao Li, Weihang Su, Yujia Zhou, Min Zhang, Yiqun Liu, and Qingyao Ai. Benchmarking LLM-as-a-judge for long-form output evaluation. *arXiv preprint arXiv:2606.01629v4*, 2026.

Jae-Woo Choi, Youngwoo Yoon, Hyobin Ong, Jaehong Kim, and Minsu Jang. LoTa-Bench: Benchmarking language-oriented task planners for embodied agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Daniel Deutsch, Rotem Dror, and Dan Roth. On the limitations of reference-free evaluations of generated text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Evan Frick, Tianle Li, Connor Chen, Wei-Lin Chiang, Anastasios N. Angelopoulos, Jiantao Jiao, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. How to evaluate reward models for RLHF. In *International Conference on Learning Representations*, 2025.

Richard Howey, Derek Long, and Maria Fox. VAL: Automatic plan validation, continuous effects and mixed initiative planning using PDDL. In *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 294–301, 2004.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9118–9147, 2022.

Abigail Z Jacobs and Hanna Wallach. Measurement and fairness. In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2021.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations*, 2024a.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4334–4353, 2024b.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, 2025.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li Erran Li, Ruohan Zhang, Weiyu Liu, Percy Liang, Li Fei-Fei, Jiayuan Mao, and Jiajun Wu. Embodied agent interface: Benchmarking LLMs for embodied decision making. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2024.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023a.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, 2023b.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, 2023c.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. RM-Bench: Benchmarking reward models of language models with subtlety and style. In *International Conference on Learning Representations*, 2025.

Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher J. Pal, and Siva Reddy. AgentRewardBench: Evaluating automatic evaluations of web agent trajectories. In *Conference on Language Modeling (COLM)*, 2025.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. RewardBench 2: Advancing reward model evaluation. In *International Conference on Learning Representations*, 2026.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4984–4997, 2020.

Matteo Merler, Nicola Dainese, Minttu Alakuijala, Giovanni Bonetta, Pietro Ferrazzi, Yu Tian, Bernardo Magnini, and Pekka Marttinen. ViPlan: A benchmark for visual planning with symbolic predicates and vision-language models. In *Findings of the Association for Computational Linguistics: EMNLP 2026*, 2026. To appear; arXiv:2505.13180v3.

Justin D. Norman, Michael U. Rivera, and D. Alex Hughes. Reliability without validity: A systematic, large-scale evaluation of LLM-as-a-judge models across agreement, consistency, and bias. *arXiv preprint arXiv:2606.19544*, 2026.

Arjun Panickssery, Samuel R Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Qwen. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. COMET: A neural framework for MT evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2685–2702, 2020.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4902–4912, 2020.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Lucia Specia, Carolina Scarton, and Gustavo Henrique Paetzold. *Quality Estimation for Machine Translation*, volume 39 of *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool, 2018.

Haoyu Sun, Wenxuan Wang, Mingyang Song, Jujie He, Weinan Zhang, Yang Liu, Yang Yang, and Yu Cheng. Agent planning benchmark: A diagnostic framework for planning capabilities in LLM agents. *arXiv preprint arXiv:2606.04874v2*, 2026.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *International Conference on Learning Representations*, 2025.

Weixi Tong and Tianyi Zhang. CodeJudge: Evaluating code generation with large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20032–20051, 2024.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450, 2024.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. EmbodiedBench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70576–70631, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Terry Yue Zhuo. ICE-score: Instructing large language models to evaluate code. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 2232–2242, 2024.

A PREREGISTERED SETUP AND AMENDMENTS

Data. 1,641 ALFRED-derived episodes in 506 trajectory clusters; the $\sim$3 paraphrases of one trajectory share a gold plan, so episodes are not independent and we test at the trajectory level. The seen/unseen split (1,237/404, Appendix R) is a compositional hold-out we construct by object–receptacle pairing, *not* ALFRED's official split, which the release also retains and Appendix K uses for the scene-shift test. LIBERO contributes 650 episodes as a protocol generality check (no execution anchor).

Planners. A scripted skeleton that is *oracle-grounded* – it reads ground-truth object and receptacle identities and should be read as a structure-only upper bound, not a language-conditioned baseline – and Qwen2.5 at 0.5B/1.5B/3B/7B (Qwen, 2024), prompted few-shot; this is the preregistered set, and Secs. 5.5 and 5.6 extend it to fifteen planners and six judges under the same protocol. **Judges (preregistered).** Qwen2.5-7B (primary), Qwen2.5-32B-AWQ (stricter, same family), and Mistral-7B-v0.3 (cross-family), served locally at temperature 0; empty predicted plans are scored deterministically (coverage 0) without a judge call. Across 148,663 judgments, zero judge failures occurred.

Statistics. Confidence intervals use a trajectory-cluster bootstrap (10,000 resamples for the preregistered tests, 2,000–5,000 for the later exploratory ones); comparisons are paired over the same clusters; Holm–Bonferroni is applied to the ten discrimination pairs and every other interval is unadjusted. Thresholds and decision rules were frozen in a preregistration (`analysis/preregistered_eval.yaml`) before outcomes were observed. Two amendments were made and are logged with timestamps in the release. First, the primary execution test covers the four *language* planners, excluding the oracle-grounded skeleton; we report every grouping, and including it makes the judge's deficit *larger* (-0.125), so the exclusion works against our own headline. This amendment was written after a 100-episode pilot had been run, which we record rather than claim it preceded all outcome data. Second, the perceptual-grounding instrument was given one round of permitted repairs after failing its gate, as its preregistered fallback allowed, and stopped there (Sec. 6). Analyses added in response to review are labelled exploratory where they appear.

The four preregistered claims, cited by number throughout, are: (1) EMBER predicts blinded human plan quality and oracle-controller execution better than the best non-LLM metric; (2) its components separate failure sources better than the best deterministic scalar; (3) a selective hybrid matches the full judge at $\geq 50\,\%$ fewer judge calls; (4) the directions hold across at least two planner families and two domains, with a consistent pooled summary. Claim 1's human half was never run (Appendix W); its execution half, and claims 2 and 3, fail; claim 4 holds for planners and not for domains.

Split-half replication. A preregistered interval says how far the estimate would move under resampling; it does not say whether the same conclusion is reached twice on disjoint halves of the data. We therefore split the 506 trajectory clusters at random into two halves, recomputed the headline deficit within each, and report both halves alongside their agreement:

The parity arms split on the parity of the trajectory's `FloorPlan` number, so the two sides hold *disjoint scenes* (48 and 44 of them): a held-out half whose rooms the tuning half never contained

Table 3: Split-half replication (exploratory, post-hoc). Every quantity is tuned on the tuning half – the gate's τ, the nine composite weights, the Platt fit – and reported on the held-out half, whose n is given. The deficit keeps its sign on all four arms and the complement keeps its sign on all four; the composite and the recalibrated Brier cover the full-pool value everywhere. Each half holds about 253 clusters, so every interval here is roughly $\sqrt{2}$ wider than the full-pool one.

Held-out half	n	EMBER−LCS	gated − LCS	composite	Brier ↓
random, tune A	3,268	-0.093 $[-0.121, -0.062]$	$+0.018$ $[+0.011, +0.026]$	0.783	0.127
random, tune B	3,296	-0.114 $[-0.144, -0.082]$	$+0.020$ $[+0.012, +0.029]$	0.796	0.135
odd floorplans, tune even	2,288	-0.120 $[-0.159, -0.082]$	$+0.018$ $[+0.006, +0.029]$	0.805	0.126
even floorplans, tune odd	4,276	-0.096 $[-0.122, -0.071]$	$+0.019$ $[+0.013, +0.025]$	0.770	0.135
full pool	6,564	-0.104	$+0.017$	*0.789*	*0.131*

Table 4: What each benchmark judges, against what. AgentRewardBench (Lù et al., 2025), JudgeBench (Tan et al., 2025), EvalPlus (Liu et al., 2023b), CodeJudge (Tong & Zhang, 2024) with ICE-Score (Zhuo, 2024), EmbodiedBench (Yang et al., 2025), the Embodied Agent Interface (Li et al., 2024). Prior work mostly ranks *models* rather than metrics, or scores metrics against a reference a model or a person produced; the code row is the exception and the nearest precedent. We score plan evaluators against what the plan does when an oracle-controlled executor runs it.

	What is judged	Reference used	Criterion
AgentRewardBench	judges of trajectories	expert human labels	human verdict
JudgeBench	LLM judges	verifiable keys	answer correctness
EvalPlus	code generators	none (execution only)	test execution
CodeJudge/ICE-Score	code metrics	none (metric input)	test execution
EmbodiedBench	planner models	task goals	simulator success
EAI	planner models	LTL goals, gold traj.	execution + symbolic
This paper	**plan evaluators**	**gold plan, and none**	**oracle execution**

still reproduces the deficit. Two cautions. The halves come from data already analysed, so this shows that no half carries the result on its own; it does not convert an exploratory finding into a confirmatory one. And ALFRED numbers rooms by type, so parity is close to arbitrary within a room type but the two sides are not exchangeable by construction – which is also why the parity arms are unequal in size.

B What the nearest benchmarks judge, and against what

Table 4 places the benchmarks nearest to ours on the two axes Related Work uses: what is judged, and what the reference and the criterion are. The benchmarks that execute judge planners, and most of the benchmarks that judge evaluators score them against labels a model or a person supplied. One row is not like that, and it is the nearest precedent rather than a contrast: ICE-Score (Zhuo, 2024) and CodeJudge (Tong & Zhang, 2024) already rank code metrics by how well each correlates with test-execution correctness, which is a metric-versus-metric contest decided by running the artifact. A test suite is also met by many programs, so a many-solution criterion is not what separates us. What embodied plans add is an executor whose failures partition into grounding, precondition and goal failures, an oracle controller that holds perception and actuation fixed while the plan varies, and the same evaluators scored against symbolic validity and execution on the same plans. That protocol, and what it finds, rather than the use of execution, is what this row contributes.

One finding in that table deserves a direct answer. AgentRewardBench (Lù et al., 2025) reports that rule-based evaluation *underreports* success, because a rule that checks for one accepted end state misses the other ways a web agent legitimately reaches it. That is a warning about our criterion, not about our judge, and the oracle controller is what limits it here: navigation and actuation are oracle, so a rollout the ALFRED goal-condition checker marks unsuccessful failed on the plan or on the executor's fixed rules – chiefly its rigid noun resolution (Appendix E.2) – rather than on the execution noise that drives the underreporting they measure. It does not remove the concern entirely, since the goal checker is still one predicate over end states; Appendix I measures how much room it leaves, and the answer is a median of two distinct successful action sequences per trajectory.

C SPECIFICATIONS: FORMULAS, DOMAINS, AND PROTOCOL

Plan order (LCS). Both plans are split on `;` and each step is stripped and lower-cased; the unit of comparison is the *whole action string*, not its characters. Writing LCS for the length of the longest common subsequence of the two step lists g and p,

$$\text{plan_order}(g, p) \;=\; \frac{\text{LCS}(g, p)}{\max(|g|, |p|)} \in [0, 1],$$

with both-empty scoring 1 and exactly-one-empty scoring 0. The max denominator penalises a plan for being too long as well as too short, which a gold-length or ROUGE-L harmonic denominator would not.

Action F1. Order-invariant set-F1 over *tokens*, not over actions. Each step is lower-cased and split on parentheses and commas, so `put(mug,sink)` yields the three tokens `put`, `mug`, `sink`. With G and P the token sets,

$$F_1 \;=\; \frac{2\,|G \cap P|}{|G| + |P|}.$$

Tokenising below the action is why a plan can score high on F1 and low on plan order: naming the right objects in the wrong structure keeps most tokens. In Fig. 1(a) that is exactly what happens.

Exact match (EM). 1 when the two normalised step lists are identical, else 0. It is a property of the *plan*; the planners' execution rate, an aggregate over rollouts, is a different quantity.

EMBER component domains. The rubric asks for, and the parser enforces, coverage $c \in \{0, 1\}$, validity $v \in \{0, 1\}$, equivalence $q \in [0, 1]$ and redundancy $r \in \{0, 1, 2\}$, combined as $s = c \cdot v \cdot (0.7\,q + 0.3\,(1 - r/2))$. A verdict that is not JSON is retried and then flagged. A verdict that parses but omits a field is not: the parser fills a missing coverage or validity with 0, a missing redundancy with 2, and clamps equivalence into $[0, 1]$, so the "zero judge failures" count in Appendix T is a count of unparseable answers, not of complete ones. The hosted-judge wrapper, moreover, recorded the flag as false whenever the call returned rather than passing on the judge's own flag; it now passes it on. That did not mask a failure here: an exhausted retry returns the rationale `judge_failed_to_return_json`, and none of the 13,128 stored gpt-5.1 verdicts carries it, while every one carries a non-empty rationale. Raw responses were not stored, so a verdict that parsed but omitted a field cannot be recovered after the fact. Because c and v are binary gates and q is in practice reported at a few round values, s takes 13 distinct values on the primary pool; that coarseness is measured against LCS in Appendix I. The prompt that elicits them is printed verbatim in Appendix D.

Judge decoding. Local judges run at temperature 0 with `max_tokens`= 512 and up to 2 retries on a malformed verdict. The hosted frontier judge rejects both `temperature`= 0 and `max_tokens`; the client sends the same arguments, then drops exactly the parameter the server names and retries, so that model runs at its own default temperature with `max_completion_tokens`= 512. Every hosted result in this paper is a single sample under those defaults, not an average over repeats, which is why Appendix R cites only its exact-match figure and not the run-to-run verb-sequence rate. One input difference must be stated. The local judges receive the task's declared object and receptacle arguments; the hosted frontier judge's client did not pass them, so gpt-5.1 judged with an empty object list. Every gpt-5.1 number in this paper therefore comes from a judge given less than the local ones, and the capability comparison of Sec. efsec:family mixes that input difference with a capability difference. We do not claim a direction for it: a fixed judge need not improve when given more input (the scene-inventory judge of Appendix efapp:sceneinv got worse), so we did not assume one: the information-matched arm was registered before it ran (`amendments_frontier_matched_inputs`) and rerun with the objects (`analysis/frontier_matched_inputs.py`). With them gpt-5.1 reaches 0.857 with the gold plan (against 0.837 without, a paired +0.020) and still trails LCS by −0.033 [−0.047, −0.019]; reference-free it reaches 0.815 (against 0.774), above the symbolic check by +0.160 and above the nine-signal composite by +0.027 [+0.005, +0.047]. The main text reports the object-given arm; the object-free numbers elsewhere in this appendix are labelled as such. The DSL handed to every judge is the executor's 10 verbs and its 4 constraints, from `data/alfred/ember/dsl.json`.

Two attempts. The preregistered nondeterminism policy credits success within $K{=}2$ attempts, each from a fresh scene reset, breaking on the first success, and applies identically to gold and predicted plans so that it cancels in every paired comparison. We later measured the executor to be deterministic (910 duplicate rollouts, zero outcome disagreements, Appendix E.2), which makes the second attempt a no-op: it runs only after a failure and reproduces it. The policy is kept as preregistered rather than removed after the fact, and no result depends on it.

Instance binding. An unindexed noun resolves to the *nearest visible* instance of its type, not to the instance the task's goal condition names, so the controller does not disambiguate on the planner's behalf. The two aliases it carries are in Appendix E.2.

The gated score. The gate composes two probabilities from the same folds, so it is one ranking and not two glued together. With p_{LCS} the out-of-fold probability from the overlap-only model and p_{both} that from the two-feature model,

$$s_i \;=\; \begin{cases} p_{\mathrm{both},i} & \text{if } \mathrm{LCS}_i \geq \tau, \\ p_{\mathrm{LCS},i} & \text{otherwise,} \end{cases}$$

and AUROC is taken over s across all episodes. In the nested variant both arms are refitted inside each outer fold, and τ is chosen on that fold's training data with an inner cross-validation, so nothing that enters a held-out score has seen its labels; the baseline is raw LCS, which no fitting touches. An earlier version fitted the overlap-only arm on all labels; correcting it moved the gated gain by less than 0.001.

D THE JUDGE'S PROMPT, VERBATIM

The rubric is the instrument under test, so it is printed here rather than described. Both blocks below are emitted by the same `build_system_prompt` and `build_user_prompt` that produced every judgment in this paper, called by `analysis/make_appendix_tables.py` when this appendix is generated, so the text here cannot drift from the text that was sent. Angle brackets mark the per-episode fields.

Three things a reader should notice. The judge is told the score identity $c \cdot v \cdot (0.7\,q + 0.3\,(1 - r/2))$ and asked to return the components and the product, so the arithmetic is not left to it. It is instructed to set equivalence to 0.5 when no gold plan is supplied, which is what makes the reference-free arm a different elicitation rather than the same one with a blank field. And its object field carries the task's two or three declared PDDL arguments, not a scene inventory; Sec. 5.2 reports what happens when the full room is supplied instead.

System message.

```
You are EMBER-JUDGE, an evaluator for instruction-to-action reasoning.
You DO NOT execute any environment. You only judge the logical
    ↪ consistency and instruction satisfaction of an action plan.

Rules:
1) Output MUST be valid JSON, and nothing else.
2) Be conservative: if uncertain, assign lower scores.
3) Do NOT invent actions or objects not present in the input.
4) Do NOT reward fluent natural language; only judge the plan.
5) Use the provided allowed actions and constraints strictly.
```

User message.

```
[CONTEXT]
Allowed actions (DSL):
- goto(x)
- pick(x)
- put(x,y)
- open(x)
- close(x)
- toggle(x)
- slice(x)
```

```
- heat(x)
- cool(x)
- clean(x)

Constraints (logical validity rules):
- put(x,y) requires pick(x) earlier in the plan.
- slice(x) requires pick(knife) earlier in the plan.
- heat(x)/cool(x)/clean(x) require pick of the object being processed
    ↪ earlier in the plan.
- Do not use undefined verbs or extra natural language.

Objects (optional):
["<the task's declared PDDL object and receptacle arguments>"]

[EPISODE]
Instruction:
<the episode's natural-language instruction>

Predicted plan (semicolon-separated):
<the planner's plan, semicolon-separated>

Gold plan (optional; may be empty):
<the gold plan, or empty in the reference-free setting>

[TASK]
Return a JSON object with the following schema:
{
  "coverage": 0 or 1,
  "validity": 0 or 1,
  "equivalence": a number from 0.0 to 1.0,
  "redundancy": 0, 1, or 2,
  "final_score": a number from 0.0 to 1.0,
  "failure_tags": [ ... ],
  "rationale": "short explanation"
}

Scoring guidelines:
- coverage: 1 if the plan clearly addresses all required objects/targets
    ↪ /relations implied by the instruction.
- validity: 1 if no constraint violations occur (e.g., placing without
    ↪ picking).
- equivalence: compare pred_plan vs gold_plan by meaning. If no gold
    ↪ plan is provided, set equivalence = 0.5.
- redundancy: 0 if minimal/efficient, 1 if slightly redundant, 2 if
    ↪ clearly redundant/unnecessary.
- final_score must be computed as:
  final_score = coverage * validity * (0.7*equivalence + 0.3*(1 -
    ↪ redundancy/2))

failure_tags must be chosen from:
['missing_object', 'wrong_target', 'missing_step', 'order_error', '
    ↪ constraint_violation', 'extra_steps', 'unknown_action', '
    ↪ ambiguous_instruction']
```

Stringency variants. The default calibration appends nothing. The two stringency arms of Table 32 append one of the following to the system message, changing nothing else.

lenient.

```
Calibration: be GENEROUS. When a plan plausibly satisfies the
    ↪ instruction, give coverage=1 and validity=1; reserve 0 for clear
    ↪ failures. Prefer higher equivalence when uncertain.
```

strict.

```
Calibration: be STRICT. Give coverage=1 and validity=1 only when the
    ↪ plan unambiguously and completely satisfies every requirement; any
    ↪  doubt means 0. Prefer lower equivalence when uncertain.
```

E THE ORACLE EXECUTOR, VALIDATED

Before any predicted plan was executed, the executor had to reproduce the gold plans: the preregistered gate was 0.85 on 100 stratified trials. It reached $93/100$ with a mean goal-condition fraction of 0.952.

Table 5: Gold-plan execution by ALFRED task type. Of the seven failures, four are `put` refusals on awkward receptacle pairings, listed below; the other three completed every action but left a goal condition unmet.

Task type	trials	success	rate
`look_at_obj_in_light`	15	15	1.00
`pick_and_place_simple`	15	15	1.00
`pick_and_place_with_movable_recep`	14	10	0.71
`pick_clean_then_place_in_recep`	14	14	1.00
`pick_cool_then_place_in_recep`	14	14	1.00
`pick_heat_then_place_in_recep`	14	13	0.93
`pick_two_obj_and_place`	14	12	0.86

Refused actions: `put(dishsponge,pot)`, `put(bread,microwave)`, `put(egg,bowl)`, `put(glassbottle,plate)`. A replay of the recorded gold trajectories reproduced $99/100$ (replaying ALFRED's recorded low-level actions rather than the DSL plan, which places the seven failures in the DSL-to-action layer rather than the simulator); rerunning this validation after the perceptual-executor edits reproduced the same $93/100$ (`VERIFY_oracle_gold_after_perceptual_edits.json`).

Execution rate by planner and task type. Fig. 4 shows where each of the fifteen executed planners sits between floor and ceiling per task type, and why `goal_not_met` dominates the failure classes.

E.1 IS THE EXECUTOR'S ERROR NEUTRAL? A TEST, AND A TRAP

Sec. 1 argues that under an oracle controller nothing but the plan's own content can explain a failure. That is an assumption about the instrument, so we tested it. AI2-THOR will refuse a placement that reads as legal (`No valid positions to place object found`) when the chosen receptacle has no free surface in that scene. If such refusals fell preferentially on receptacles the gold plan does *not* use, the instrument would penalise divergence for reasons that are not about planning, and in the direction that hurts the judge, since a plan reaching the goal by another receptacle is what a semantic judge should credit.

They do not. Among receptacle actions the simulator actually attempted, a non-gold receptacle draws a physical refusal at $1.48\,\%$ against $2.47\,\%$ for a gold one ($15/1013$ and $106/4291$): a relative risk of 0.60, with a trajectory-cluster interval of $[0.07, 1.59]$. The point estimate is below one and the interval spans it, so the data show no physical penalty for choosing a different receptacle – though the interval is wide enough that a moderate one would not have been detected, and we say that rather than claim the test is decisive.

The trap. Non-gold receptacles *do* fail more often overall, and it is easy to read that as instrument bias. The excess is somewhere else. Classified by the simulator's own message, a non-gold receptacle fails to *resolve* at $13.1\,\%$ against $1.1\,\%$ for a gold one, a relative risk of 11.7. A referent that does not exist in the scene is a planning error, and it is the error the criterion is meant to punish; Appendix M reaches the same place from the other direction, half of the plans the judge credits naming an object the task never declared. Our own first pass filed `no valid receptacle found` and `this target object is not a receptacle` under physics rather than resolution and

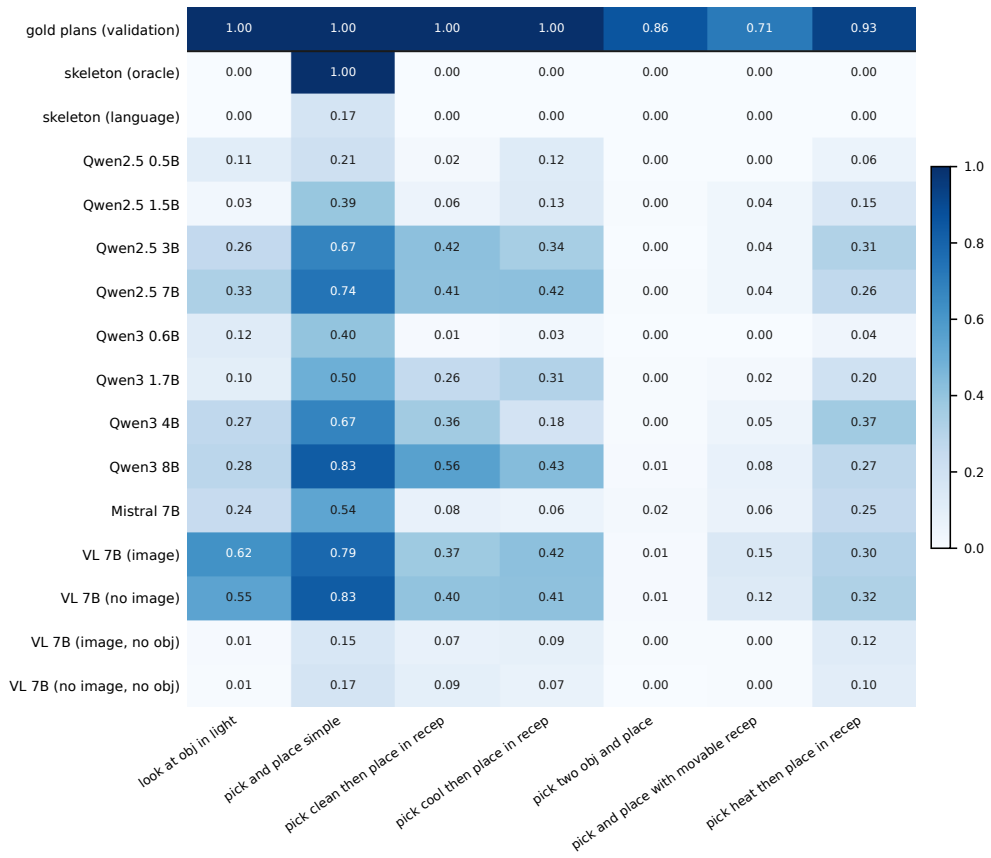

Figure 4: Execution success rate by planner (rows) and ALFRED task type (columns), with the gold-plan validation as the top row. Cells are rates; column widths are not proportional to n.

reported a $2.78\times$ bias that does not exist. The two classes have to be kept apart, or the planner's error is charged to the simulator.

For completeness, the bound the mistaken reading would have needed: removing every rollout whose recorded failure is a non-gold physical refusal (15 of 6564) leaves the headline at -0.1037, CI $[-0.1251, -0.0821]$, against the published -0.104 $[-0.125, -0.082]$.

What secures the headline is not the interval but the exposure. The physical class is measured only over failures the simulator described, and `unspecified` plus `other` messages are a sixth of classified failures, so the physical count is a lower bound. Even so, the rollouts a non-gold physical refusal could explain are few enough that no plausible bias among them moves the result, which is what the removal above shows.

E.2 Lexical rigidity: how much of the lead does it buy?

The oracle controller resolves a DSL noun by looking it up in the scene's object-type map, and it carries exactly two aliases, `sink`→`sinkbasin` and `bathtub`→`bathtubbasin`. A plan that writes `refrigerator` where AI2-THOR writes `fridge` therefore fails outright. The objection this invites is serious: an overlap metric compares against the gold plan's canonical vocabulary, so it is aligned with that rigidity by construction, while a judge that credits the synonym is marked wrong for being right. We take it seriously enough to measure it.

Referent failures split three ways. A name is an *alias* miss when the scene holds a type reachable from it by a rule fixed in advance – substring containment either way, a normalised edit distance at or below 0.25, or one of a short list of ordinary English synonyms, all of them printed in the artifact. It is *absent* when the name is a real AI2-THOR type that this room does not contain, and *unknown*

when it appears in no scene in the corpus. Only the first class is what the objection is about; the other two are plans naming things that are not there.

Of 996 referent failures, 621 (62.3 %) are alias misses, 47 (4.7 %) name an absent type and 328 (32.9 %) name nothing in the corpus. The alias class is reached by substring in 375 cases, by synonym in 144 and by spelling in 102; `counter`, `table`, `stove`, `refrigerator` are typical. So the objection is right about the mechanism, and it covers more than half of these failures.

It does not carry the result. Dropping every rollout whose recorded failure is an alias miss – 621 of 6,564, removed rather than recoded as successes, since a resolved noun does not make the rest of a plan run – leaves EMBER at 0.7904 against LCS at 0.8856: a deficit of −0.0952, CI [−0.1176, −0.0728], against the published −0.104 [−0.125, −0.082]. Removing those plans moves the deficit by under a tenth. That is a statement about the plans as scored, not a causal bound: we did not re-execute them under a synonym layer, so what they would have done next – succeed, or fail at a later step – is not observed, and the deletion neither establishes nor excludes a larger effect of lexical rigidity on execution.

Two further notes on the executor's resolution, both of which were reviewer questions. It binds an unindexed noun to the *nearest visible* instance of that type, not to the instance the task's goal condition names, so it does not quietly disambiguate on the planner's behalf. And the two aliases it does carry were fixed before any predicted plan was scored, for receptacles whose ALFRED name and AI2-THOR name differ; we did not add more after seeing the failures, which is why the alias class above is measured rather than repaired.

Executor determinism. An interrupted AI2-THOR run was resumed, which re-executed episodes the first pass had already completed and so gives an unplanned repeat-execution test: 910 episodes ran twice, in separate processes, and **none disagreed** on execution success. The oracle executor is therefore deterministic across restarts on this machine, at least on this sample; we make no claim across `ai2thor` versions or machines. The operational consequence is that a resumed run legitimately writes duplicate episode ids, so every analysis deduplicates last-wins before use.

F THE NON-ORACLE INSTRUMENT

We attempted the non-oracle test and report its failure. Our *perceptual* executor resolves DSL nouns only among observed objects, leaving navigation and motor execution oracle. Preregistration required gold plans to reproduce at ≥ 0.85; it reached 0.51, and 0.61 after the permitted fixes, 87 % of failures being "the object was never observed" – an instrument on which *correct* plans fail two times in five measures object search, not plan quality, so we stopped at the gate.

Rebuilding that instrument outside the preregistration, with an active search over nearby reachable stances (ordered by distance from the agent, never by proximity to the target), reached 0.68 – still short, and we stopped rather than tune to a pass. Why it stops is the informative part: 26 of the 32 residual failures are "unobserved"; 20 of them have the same shape, `goto(`*`receptacle`*`)` then `pick(`*`object`*`)`, and a steeper downward sweep changed none of them. The DSL's `goto` targets the receptacle while perceptual grounding needs the object *seen*, and the plan never tells the agent to look for it, so no search policy inside the plan can close this. Dropping oracle grounding therefore changes the plan interface rather than a flag. Until that is faced, execution-grounded meta-evaluation is confined to the oracle regime, which bounds our negative result and equally any positive one. We release the executor and all three gate artifacts.

Our reference-free result inherits this sharply: we argue from deployment – a robot has no gold plan – yet score against oracle-controller execution, which a deployed robot equally lacks and which removes the grounding failures a semantic judge should catch. The reference-free advantages of Sec. 5.4 – +0.055 for the 7B judge and +0.160 for the frontier one given the task's objects – are orderings under oracle control, not deployment benefits.

G THE EXECUTION DEFICIT, PLANNER BY PLANNER

Table 6 gives the deficit of every family planner and of the two skeletons; Fig. 2a plots the eleven family members.

Table 6: EMBER−LCS execution AUROC for the eleven family planners and the two scripted skeletons (the VLM's two no-object arms are in Table 7). All thirteen are negative; the eleven family members' intervals all exclude zero and pool to a joint trajectory-bootstrap mean of -0.114 $[-0.137, -0.089]$. The two scripted skeletons are reported but excluded from every summary.

Family	Planner	exec. rate	EMBER	Δ vs. LCS (95 % CI)
Qwen2.5	0.5B	0.077	0.742	-0.111 $[-0.160, -0.064]$
	1.5B	0.116	0.795	-0.087 $[-0.128, -0.048]$
	3B	0.299	0.751	-0.139 $[-0.173, -0.105]$
	7B	0.322	0.745	-0.093 $[-0.129, -0.054]$
Qwen3	0.6B	0.090	0.802	-0.088 $[-0.139, -0.038]$
	1.7B	0.202	0.768	-0.095 $[-0.131, -0.061]$
	4B	0.280	0.739	-0.121 $[-0.158, -0.084]$
	8B	0.359	0.756	-0.120 $[-0.155, -0.084]$
Mistral	7B	0.186	0.720	-0.146 $[-0.179, -0.114]$
Qwen2.5-VL	7B (image)	0.394	0.712	-0.125 $[-0.165, -0.084]$
	7B (text)	0.392	0.733	-0.125 $[-0.163, -0.087]$
scripted	oracle-grounded	0.147	0.784	-0.216 $[-0.260, -0.175]$
	language-conditioned	0.026	0.896	-0.097 $[-0.154, -0.050]$

H THREE ROBUSTNESS CHECKS: MULTIMODAL PLANNERS, A POOLED REFERENCE-FREE COMPARATOR, THE WELL-FORMED SUBSET

The full 2×2. Table 7 gives all four arms of the multimodal ablation. All are negative with intervals excluding zero, and the image-on and image-off deltas are identical to three decimals, so the deficit does not depend on the planner having visual input. Removing the object list collapses execution to $6.5\,\%$ and drives LCS's AUROC to 0.98: when plans are that bad, structure alone predicts failure, which is why those arms are an ablation rather than a fair family comparison.

What the image is worth. A planner that sees the scene is only evidence about seeing if the picture does something, so we measured its effect on the planner rather than assuming it (`analysis/vlm_image_effect.py`). Over the $1,641$ episodes the two arms emit the same normalised plan $66.3\,\%$ of the time. The image is not inert: it rewrites 553 plans and flips 141 execution outcomes, a $25.5\,\%$ flip rate among the plans it changed. What it does not do is help. The two arms agree on $91.3\,\%$ of outcomes and their execution rates differ by $+0.002$ (0.394 with the frame, 0.392 without), so the changes cancel. The honest reading of this arm is therefore narrow, and we state it that way in the main text: the deficit survives a planner given visual input whose visual input we verified does not change how often its plans run. It is not a test of perception-grounded *evaluation*, and could not be: no evaluator in this paper ever sees an image.

Table 7: Qwen2.5-VL-7B, all four conditions (n=1,641 each).

Condition	Exec. rate	EMBER	LCS	Δ (95 % CI)
image + objects	0.394	0.712	0.837	-0.125 $[-0.165, -0.084]$
no image + objects	0.392	0.733	0.859	-0.125 $[-0.163, -0.087]$
image, no objects	0.065	0.821	0.981	-0.160 $[-0.199, -0.125]$
no image, no objects	0.065	0.832	0.987	-0.156 $[-0.198, -0.119]$

How fair is the reference-free comparator? The main text compares the reference-free judge against the strongest single reference-free baseline, symbolic validity. A reviewer will reasonably ask what happens against the best *combination* of reference-free signals, so we pool symbolic validity, DSL well-formedness and plan length into one out-of-fold, cluster-grouped logistic model. That comparator reaches 0.675 against symbolic validity's 0.655, and the judge's advantage falls from $+0.119$ $[+0.097, +0.139]$ to $+0.099$ $[+0.070, +0.128]$. It is smaller, and it still excludes zero. We report the single-baseline figure in the main text because it is the preregistered comparator, and this one because it is the harder test.

Information-matched baselines. The comparator above reads the plan text alone, while the judge also reads the instruction and the task's declared object and receptacle arguments. A reviewer will ask what a free baseline with the *same* inputs achieves, so we built three (exploratory): the fraction of plan arguments found in the task's declared objects, the cosine between instruction and plan text under `all-MiniLM-L6-v2`, and the fraction of plan arguments occurring in the instruction. Table 8 scores them on the primary pool (`analysis/reference_free_matched.py`).

Table 8: Reference-free execution prediction with the judge's own inputs (6,564 episodes, 506 clusters, 2,000-resample trajectory-cluster intervals; exploratory). The pooled row is fitted on execution labels out of fold; every other row is unfitted.

Reference-free signal	AUROC (95 % CI)
Referent grounding (task arguments)	0.640 [0.624, 0.655]
Instruction–plan cosine	0.538 [0.510, 0.566]
Instruction–argument overlap	0.529 [0.501, 0.557]
Symbolic validity (main text)	0.655 [0.645, 0.665]
Pooled, all nine free signals	0.789 [0.770, 0.808]
EMBER 7B, gold-free	0.710 [0.692, 0.728]
EMBER gpt-5.1, gold-free	0.774 [0.752, 0.795]

No single matched feature displaces the judge. Referent grounding is the best of the three at 0.640, still short of symbolic validity's 0.655, so the preregistered argmax over single baselines is unchanged and the $+0.055$ and $+0.119$ of the main text stand; against grounding itself the judge leads by $+0.069$ $[+0.048, +0.090]$ (7B) and $+0.134$ $[+0.107, +0.160]$ (gpt-5.1). Instruction–plan similarity is close to chance (0.538), which is itself informative: what makes overlap against a gold plan strong is the reference, not textual similarity to the instruction.

A *pooled* free comparator is a different story. An out-of-fold logistic over all nine non-judge signals reaches 0.789 – the three plan-text features alone reach 0.674 here, the 0.675 above being the same pool from an artifact that rounds well-formedness to an integer, so reading the judge's inputs is worth 0.115 of it – and it beats the gold-free 7B judge (-0.079 $[-0.099, -0.058]$) while drawing level with the frontier one (-0.015 $[-0.038, +0.008]$). Two things temper that. The pooled comparators are fitted on execution labels while the judge is not fitted at all, the same asymmetry the comparator above already carries, so this row is what a fitted pool of free signals already reaches, not a scorer one could drop in. And it needs execution labels to exist: the reference-free setting removes the gold plan, not past execution labels, so this row is the competitor where such labels exist and unavailable where they do not.

The hard subset. Two failure classes – unknown vocabulary and a referent absent from the scene – are cheap for a checker to catch and are also exactly what overlap against the gold plan penalizes. Restricting to plans that are DSL-well-formed and symbolically valid removes the first class outright. It does not remove the second: our symbolic checker reads the plan and not the scene, so of the 967 rollouts that fail on an unresolvable referent, 716 (74 %) survive into the subset. The subset is therefore a test against malformed and ordering-violating plans, not against grounding failures, and a scene-aware filter is untested.

That leaves 4,796 of 6,564 plans (73.1 %), executing at 0.273 against 0.204 overall. Execution AUROC there: EMBER 0.774, action F1 0.847, LCS 0.867, success rate 0.544; EMBER minus LCS is -0.094 $[-0.121, -0.068]$. The deficit is not carried by the plans a checker could have rejected.

I Two objections to the criterion: score resolution and structural slack

Score resolution. AUROC scores a tie at half credit. On the primary pool EMBER takes 13 distinct values, with 31 % of scores at 0.5 and 25 % at 0.0, while LCS takes 61; a coarse score can therefore trail a fine one at equal validity. We coarsen LCS to a few levels and recompute the preregistered paired delta at each resolution, by quantile binning (primary) and by equal-width binning (Table 9).

Table 9: The deficit against LCS coarsened to a handful of levels (primary pool, n=6,564, trajectory-cluster CIs; the first column is the number of levels the binning actually realizes). The raw score has 61 levels; the judge has 13. Under quantile binning the deficit holds down to four levels and dissolves only at three; under equal-width binning it holds at every rung.

	quantile bins		equal-width bins	
levels	LCS AUROC	EMBER−LCS (95 % CI)	LCS AUROC	EMBER−LCS (95 % CI)
3	0.790	-0.004 $[-0.025, +0.017]$	0.858	-0.072 $[-0.093, -0.051]$
4	0.867	-0.082 $[-0.104, -0.060]$	0.856	-0.071 $[-0.095, -0.048]$
5	0.888	-0.102 $[-0.122, -0.082]$	0.880	-0.095 $[-0.115, -0.074]$
6	0.858	-0.072 $[-0.096, -0.049]$	0.878	-0.092 $[-0.114, -0.070]$
8	0.890	-0.104 $[-0.125, -0.083]$	0.889	-0.103 $[-0.124, -0.082]$
10	0.889	-0.104 $[-0.125, -0.082]$	0.886	-0.100 $[-0.121, -0.079]$
raw (61)	0.890	-0.104 $[-0.125, -0.083]$	0.890	-0.104 $[-0.125, -0.083]$

Structural slack. Under an oracle controller, does "executes" collapse to "is the gold plan"? If it did, an overlap metric would be optimal by construction. It does not. On the primary pool 117 of 6,564 predicted plans reproduce the gold plan exactly (1.8 %), and those are 8.6 % of the 1,338 successes. Among successful plans the LCS score has median 0.667 (quartiles 0.571 and 0.750), 24.6 % score at or below 0.5, and 91.4 % score below 1. Of the 311 trajectories with at least one success, the median has *two* distinct successful action sequences; among the 277 with at least two successes, 83.4 % succeed by at least two distinct sequences. Over all eleven family planners the picture is the same: 2.1 % of plans match gold exactly, 8.5 % of successes do, and the median successful LCS is 0.750. Execution tolerates real structural deviation, and this is a stronger answer to the "structural by construction" objection than the outcome-selected slice of Sec. 5.1, because nothing here is selected on the outcome.

J THE DISJOINT-FAMILY RESULT UNDER A SECOND JUDGE AND A SECOND COMPARATOR

The cross-family generality result rests on one planner, so we vary the two choices that could have produced it. Neither does (Table 10): the deficit keeps its sign and interval under a different judge and under a different deterministic comparator, and the stronger judge narrows it exactly as the capability trend of Sec. 5.5 predicts.

Table 10: Mistral-7B planner (n=1,641): the deficit is not specific to the primary judge or to LCS.

Judge	Comparator	EMBER AUROC	Comparator AUROC	Δ (95 % CI)
Qwen2.5-7B (primary)	LCS	0.720	0.867	-0.146 $[-0.179, -0.114]$
Qwen2.5-7B (primary)	Action F1	0.720	0.828	-0.108 $[-0.143, -0.074]$
Qwen2.5-32B	LCS	0.757	0.867	-0.109 $[-0.137, -0.081]$
Qwen2.5-32B	Action F1	0.757	0.828	-0.071 $[-0.103, -0.040]$

An earlier version of this appendix said the same-family decomposition was unavailable. It was not: the judgments had been collected for the self-preference 2 × 2 and simply never carried into this table. They are here now (`analysis/mistral_same_family.py`). Scored by the Mistral-7B judge, its own family, the planner gives EMBER 0.737 against LCS 0.867, a deficit of -0.130 $[-0.154, -0.105]$; the cross-family Qwen judge gives -0.146 $[-0.179, -0.114]$. Both exclude zero, so the deficit here is not an artifact of judging a planner from another family, and the same-family pairing is the *smaller* of the two, which is the direction self-preference predicts.

K THE DEFICIT UNDER ALFRED'S FLOORPLAN SPLIT

ALFRED's official validation split partitions exactly the episodes we already executed – 820 `valid_seen` and 821 `valid_unseen`, summing to our 1,641 – by floorplan, and we had retained it without using it. It costs no new rollouts to ask whether the deficit depends on which rooms the episodes take place in.

What it does not test is worth stating first. Our planners are zero-shot and never trained on ALFRED, so seen versus unseen is not a train/test distinction for them; it is a different distribution of rooms and object inventories. It is *not* a second domain, and the domain half of preregistered claim 4 remains unmet.

Table 11: The execution deficit under ALFRED's floorplan split. The two strata are disjoint in episodes and in trajectory clusters, so the contrast is unpaired and each stratum is resampled independently.

Stratum	exec. rate	pooled Δ (95 % CI)	joint trajectory bootstrap ($k = 11$)
`valid_seen`	0.241	-0.110 $[-0.146, -0.076]$	-0.112 $[-0.150, -0.075]$
`valid_unseen`	0.253	-0.117 $[-0.145, -0.090]$	-0.113 $[-0.143, -0.083]$

No difference is detectable across the split: the pooled contrast is -0.007 $[-0.052, +0.037]$, which does not exclude zero, on an interval wide enough to admit a substantial shift in either direction. Every one of the eleven planners is negative in both strata; the interval excludes zero for 11 of 11 in `valid_seen` and 10 of 11 in `valid_unseen`, the exception being Qwen3-0.6B (-0.063 $[-0.142, +0.013]$). No single planner's seen-versus-unseen difference excludes zero either. Execution rates are similar or slightly higher in the shifted scenes, so they are not simply harder.

The judge's complement (Appendix L) also transfers: $+0.018$ $[+0.008, +0.027]$ in `valid_seen` and $+0.014$ $[+0.007, +0.022]$ in `valid_unseen`, both excluding zero. Set against its absence in one planner family, the positive result survives a change of scene more comfortably than a change of planner.

Per planner. The deficit is negative for all eleven planners in both floorplan strata, and the complement is positive for every planner but Mistral-7B (Fig. 5, Table 12).

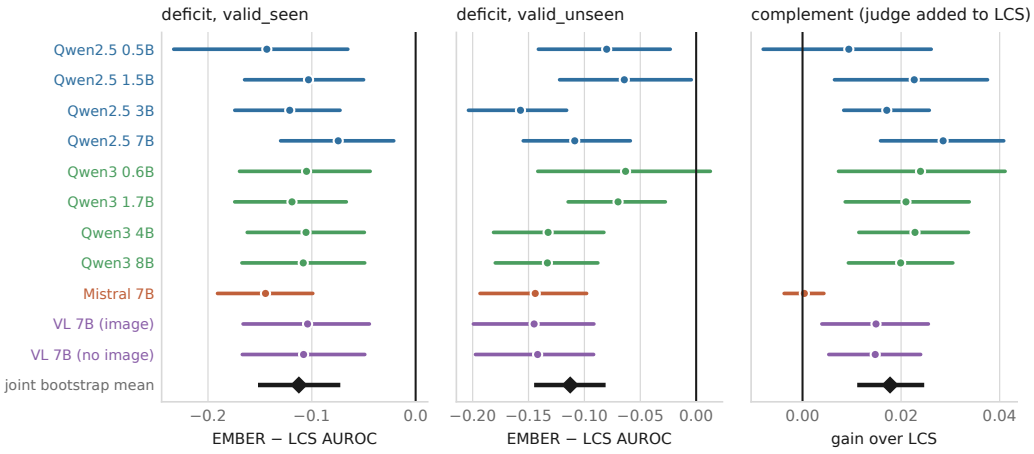

Figure 5: Eleven planners, three quantities. Left and centre: the execution deficit in ALFRED's `valid_seen` and `valid_unseen` floorplans; right: the judge's out-of-fold complement over LCS. Bars are 95 % trajectory-cluster intervals; the diamond is the joint trajectory-bootstrap mean. The deficit is negative for every planner in both strata; the complement is absent for Mistral.

K.1 Does the deficit generalize over planners?

Every interval elsewhere in this paper resamples trajectory clusters, which answers what would happen with different *episodes*. When Sec. 5.6 says the deficit holds across planners it is asking a different question, and a trajectory bootstrap cannot answer it. Here the planner is the unit.

Treating the eleven per-planner deltas as the observations, the planner-level t-interval is -0.1136, 95 % CI $[-0.1274, -0.0999]$, with a between-planner standard deviation of 0.0205. It is *narrower* than the joint trajectory bootstrap's $[-0.137, -0.089]$, not wider, because the eleven deltas sit close together: the spread across planners is small next to the sampling noise within one. Dropping any

Table 12: Per-planner deficit by floorplan stratum, and per-planner complement.

Planner	seen Δ (95 % CI)	unseen Δ (95 % CI)	complement (95 % CI)
Qwen2.5-0.5B	-0.143 $[-0.233, -0.065]$	-0.080 $[-0.141, -0.023]$	$+0.009$ $[-0.008, +0.026]$
Qwen2.5-1.5B	-0.103 $[-0.165, -0.050]$	-0.064 $[-0.122, -0.004]$	$+0.023$ $[+0.006, +0.038]$
Qwen2.5-3B	-0.121 $[-0.174, -0.073]$	-0.157 $[-0.204, -0.116]$	$+0.017$ $[+0.008, +0.026]$
Qwen2.5-7B	-0.075 $[-0.130, -0.021]$	-0.109 $[-0.155, -0.059]$	$+0.029$ $[+0.016, +0.041]$
Qwen3-0.6B	-0.105 $[-0.170, -0.044]$	-0.063 $[-0.142, +0.013]$	$+0.024$ $[+0.007, +0.041]$
Qwen3-1.7B	-0.119 $[-0.174, -0.067]$	-0.070 $[-0.115, -0.028]$	$+0.021$ $[+0.009, +0.034]$
Qwen3-4B	-0.106 $[-0.162, -0.049]$	-0.132 $[-0.181, -0.083]$	$+0.023$ $[+0.011, +0.034]$
Qwen3-8B	-0.108 $[-0.167, -0.049]$	-0.133 $[-0.180, -0.088]$	$+0.020$ $[+0.009, +0.030]$
Mistral-7B	-0.145 $[-0.191, -0.099]$	-0.144 $[-0.193, -0.098]$	0.000 $[-0.004, +0.004]$
Qwen2.5-VL-7B (image)	-0.104 $[-0.166, -0.045]$	-0.145 $[-0.199, -0.092]$	$+0.015$ $[+0.004, +0.026]$
Qwen2.5-VL-7B (text)	-0.108 $[-0.167, -0.049]$	-0.142 $[-0.197, -0.092]$	$+0.015$ $[+0.005, +0.024]$

single planner moves the mean only within $[-0.1163, -0.1104]$, and every one of the eleven refits still excludes zero.

Dropping a whole family is the stronger test, since planners inside a family are not independent draws either:

Table 13: Leave-one-family-out. Each row refits the planner-level interval with one family removed. Regenerated by `analysis/planner_generalization.py`.

Family removed	Planners left	Mean Δ	95 % CI
Mistral	10	-0.1104	$[-0.1235, -0.0972]$
Qwen2.5	7	-0.1171	$[-0.1353, -0.0989]$
Qwen2.5-VL	9	-0.1111	$[-0.1280, -0.0942]$
Qwen3	7	-0.1180	$[-0.1386, -0.0974]$

Three things this does not establish. Eleven planners in four families are not a random sample of the planners anyone might use, so the interval describes this set and not a population. Two of the eleven are one VLM with and without its image, which is close to a duplicate draw; averaging them into a single planner gives -0.1125, $[-0.1277, -0.0973]$ over 10 planners, which still excludes zero. And all eleven score the same $1{,}641$ episodes, the same scenes and the same gold plans, so the eleven deltas are correlated through the shared task distribution while the t-interval treats them as independent. Both push in the direction of an interval that is too narrow, which is the direction that matters, so read it as a consistency check rather than as a generalization guarantee.

L THE JUDGE AS A COMPLEMENT, AND WHAT IT COSTS

The preregistered tests asked whether the judge *beats* overlap. These exploratory analyses ask whether it is worth *adding* to overlap, which is the question a practitioner faces once the reference is in hand. The judge arm is always fitted out of fold (trajectory-cluster-grouped five-fold, standardization refitted inside each fold). The overlap-only arm is fitted globally on purpose: AUROC is invariant under monotone transformation, so a one-feature logistic cannot gain from seeing the labels, whereas refitting it per fold would blend five different monotone maps into one ranking and depress the baseline below raw LCS – inflating every gain measured against it.

Gating the judge to episodes with LCS $\geq \tau$ traces gain against cost; the $\tau{=}0$ row is the full gain, calling the judge on every episode.

A threshold read off that sweep is selected on the data it is scored on, so the deployable figure comes from a nested procedure: each outer fold chooses τ on its training folds using an inner cross-validation, then applies it to the held-out fold. That procedure spends $53.2\,\%$ of the judge calls for $+0.017$ $[+0.011, +0.023]$, against $+0.016$ for calling the judge everywhere. The per-fold thresholds were $0.5, 0.5, 0.5, 0.375, 0.5$.

Ten of the eleven planner estimates are positive and one is zero to three decimals; two intervals cover zero (Qwen2.5-0.5B and Mistral-7B), and Mistral-7B is centred on it, so we report the complement as replicating in three families rather than generalizing across planner families.

Table 14: The gated judge. Gating to roughly half the episodes does not cost the gain; below that it collapses. A reverse gate spending the same budget on the low-overlap side recovers nothing, locating the gain in the stratum where overlap is already confident.

τ	judge calls	Δ vs. LCS	share of full gain
0.000	100.0 %	+0.016	100.0 %
0.167	94.4 %	+0.016	99.9 %
0.250	87.5 %	+0.017	102.6 %
0.333	73.0 %	+0.016	97.8 %
0.375	61.6 %	+0.017	107.4 %
0.500	51.5 %	+0.018	109.8 %
0.667	22.1 %	+0.006	34.3 %
0.750	14.5 %	+0.002	13.1 %

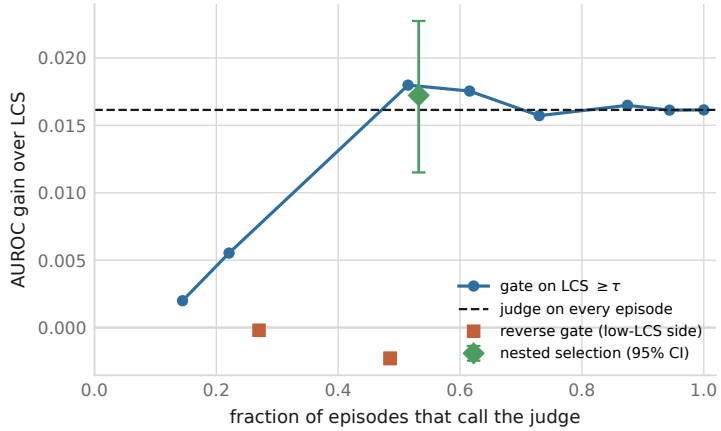

Figure 6: Gain over LCS against the fraction of episodes that call the judge. The forward gate keeps the full gain down to about half the calls and collapses below; the reverse gate spends the same budget on the low-overlap side and recovers nothing; the diamond is the nested selection with its interval.

Finally, judge capability and the complement come apart. On the same 6,564 episodes the frontier judge reaches AUROC 0.911 against overlap's 0.890, a complement of +0.021 [+0.016, +0.028] versus Qwen2.5-7B's +0.016. The paired difference is +0.005 [−0.001, +0.011] and does not exclude zero: a stronger judge is a markedly better substitute for the reference – its standalone deficit against LCS halves, −0.104 to −0.053 (Sec. 5.5) – without being a detectably better complement to it.

One further null belongs here. On the valid-alternative subset – the 3,089 plans a symbolic checker accepts but LCS scores at or below its median, where a semantic judge should most help – EMBER still does not beat LCS at predicting execution (AUROC 0.767 versus 0.792; paired −0.024, 95 % CI [−0.060, +0.010]).

L.1 Decision-level utility, token counts, and partition stability

AUROC ranks every plan against every other. Two uses of a score rank differently, so we test both on the stored rollouts (`analysis/decision_utility.py`; exploratory, added after the main results were known). *Selection*: for each instruction several planners propose a plan and the highest-scoring one is executed; ties at the maximum are resolved in expectation, so no selector gains from a tie-break order. *Acceptance*: the top q of plans in the primary pool are accepted, ties at the cut fractionally. The fitted scorers are the ones the paper recommends, not proxies: the out-of-fold combination and the nested gate, fitted on the pooled candidates with trajectory-cluster folds, so every candidate for an instruction is held out together. Intervals are trajectory-cluster bootstraps (B=10,000), paired; †gpt-5.1 was run without the declared-object list (Appendix C).

Table 15: The complement by planner family. It is absent in the family where the deficit is largest; the between-planner SD is 0.008 against the deficit's 0.020.

Planner family	n	complement (95 % CI)
Qwen2.5	6,564	$+0.016$ $[+0.010, +0.022]$
Qwen3	6,564	$+0.018$ $[+0.011, +0.024]$
Mistral	1,641	0.000 $[-0.004, +0.004]$
Qwen2.5-VL	3,282	$+0.015$ $[+0.005, +0.024]$
Joint trajectory bootstrap ($k = 11$)		$+0.018$ $[+0.012, +0.024]$

Table 16: Execution rate of the selected plan. Left: the eleven planners that score the same 1,641 episodes; right: the four Qwen2.5 planners, where the frontier and re-elicited reference-free judgments exist. Δ is against LCS.

Selector	11 planners	Δ	4 planners	Δ
LCS	0.502	–	0.388	–
action F1	0.471	-0.030	0.394	$+0.006$
EMBER (7B)	0.360	-0.141	0.310	-0.078
OOF combination (LCS + EMBER)	0.497	-0.005	0.388	$+0.000$
nested gate	0.496	-0.005	0.390	$+0.002$
LCS, ties broken by EMBER	0.499	-0.003	0.390	$+0.001$
EMBER (gpt-5.1)†	–	–	0.366	-0.022
symbolic validity	–	–	0.265	-0.123
EMBER (7B), reference-free	–	–	0.238	-0.151
EMBER (gpt-5.1)†, reference-free	–	–	0.320	-0.068
symbolic, ties broken by EMBER (7B, ref.-free)	–	–	0.273	-0.116
random candidate	0.247	-0.254	0.204	-0.184
best single planner, chosen in hindsight	0.394	-0.107	0.322	-0.066
oracle (any candidate executes)	0.567	$+0.065$	0.429	$+0.041$

Overlap's lead carries over to selection. Choosing among eleven planners' plans, overlap selects one that executes 0.502 of the time and the 7B judge 0.360 (-0.141, $[-0.161, -0.123]$); the judge's selections are worse than always running the best single planner chosen in hindsight (0.394). The complement does *not* carry over. Selecting by the out-of-fold combination or the nested gate gives $+0.000$ $[-0.004, +0.004]$ and $+0.002$ $[-0.002, +0.006]$ on the four-planner pool and -0.005 $[-0.010, -0.001]$ and -0.005 $[-0.010, -0.001]$ on the eleven: no gain, and on the larger pool a small loss. The AUROC gain is a gain in ranking plans *across* instructions, which is what acceptance uses, and not in choosing among the plans for one instruction. Without a reference, the 7B judge re-elicited without the gold plan selects *worse* than the symbolic check (-0.027, $[-0.037, -0.019]$) although it leads it on AUROC, and the frontier judge selects better ($+0.055$, $[+0.042, +0.068]$). Two judges cannot place a threshold; they show that an AUROC lead need not survive the decision it is used for.

Table 17: Precision (execution rate among accepted plans) when the top q of the primary pool (base rate 0.204) is accepted, and its paired difference from LCS; the text gives fixed-cut and policy-level intervals.

Scorer	q=0.1		q=0.2		q=0.3	
	prec.	$-$LCS	prec.	$-$LCS	prec.	$-$LCS
LCS	0.709	–	0.662	–	0.526	–
OOF combination (LCS + EMBER)	0.755	$+0.046$	0.693	$+0.031$	0.562	$+0.036$
nested gate	0.755	$+0.046$	0.693	$+0.031$	0.562	$+0.036$
EMBER (7B)	0.612	-0.097	0.495	-0.167	0.398	-0.128
EMBER (gpt-5.1)†	0.596	-0.113	0.579	-0.084	0.551	$+0.025$
action F1	0.520	-0.189	0.513	-0.149	0.513	-0.014
symbolic validity	0.273	-0.436	0.273	-0.389	0.273	-0.253
EMBER (7B), reference-free	0.340	-0.369	0.340	-0.322	0.340	-0.186
EMBER (gpt-5.1)†, reference-free	0.513	-0.196	0.513	-0.149	0.495	-0.031

Acceptance is where the complement can pay, and how much depends on which uncertainty is asked about. The combination's accepted plans execute more often than overlap's by $+0.046$ at q=0.1, $+0.031$ at 0.2 and $+0.036$ at 0.3. Holding the cut fixed, the intervals are $[+0.013, +0.082]$, $[+0.018, +0.044]$ and $[+0.021, +0.050]$. Re-choosing the top-q cut inside every resample, as a new pool's top q would be (the policy, 2,000 draws), they are $[-0.000, +0.090]$, $[-0.007, +0.044]$ and $[+0.022, +0.046]$: only the largest budget excludes zero. The nested gate accepts the same plans, since the top of the pool lies above its threshold. No judge alone beats overlap at a small budget; the frontier judge's point estimate passes it only at q=0.3 ($+0.025$, policy $[-0.002, +0.042]$). The reference-free scorers take too few distinct values to resolve a small budget, so their precision is flat or nearly flat across q.

Reconstructed token counts. The nested gate calls the judge on 53.2 % of episodes. Token counts *reconstructed* with the judge's tokenizer – on the prompt rebuilt from the stored inputs, and on the verdict re-serialized from the stored fields – put 53.3 % of input and 53.5 % of output tokens on those calls (610 against 617 input tokens per call, called versus not), so the gate does not concentrate calls on longer plans. These are not cost measurements: raw generations, retries, latency and GPU time were not recorded, and we report calls, input tokens and output tokens separately rather than a price.

Three kinds of uncertainty, kept apart. (i) The bootstrap intervals on the complement and the gate resample trajectories with the out-of-fold predictions held fixed, so they are conditional on one partition and its fits. (ii) Rerunning the whole procedure – partition, fits, inner cross-validation and threshold choice – under 20 partitions fixed in advance (`analysis/cv_stability.py`) gives a complement of $+0.0161$ (SD 0.0004, range $[+0.0153, +0.0168]$) and a gated gain of $+0.0173$ (SD 0.0005) at 51.5–53.7 % of calls, positive in all 20: sensitivity to the partition of these 506 clusters, not to a new sample of trajectories. (iii) Judge sampling is in neither: every judge was decoded once, the hosted one at its default sampling.

L.2 Why a better ranking does not make a better choice

Four metrics on the same scores separate what AUROC averages together (`analysis/decision_decomposition.py`; exploratory). With ties counted half, pooled AUROC is $w \cdot \mathrm{AUC}_{\text{within}} + (1 - w) \cdot \mathrm{AUC}_{\text{between}}$, where w is the share of executing/failing pairs that come from the same instruction. Adding a constant per instruction changes the pooled AUROC and cannot change a choice, so pooled AUROC is not sufficient for selection; that is arithmetic. What the data add is how small w is, and where the choices move.

Table 18: The four metrics on the four-planner pool (w=0.00033) and the eleven-planner pool (w=0.00035).

	4 planners				11 planners			
Scorer	pooled	within	top-1	prec.@10	pooled	within	top-1	prec.@10
LCS	0.890	0.907	0.388	0.709	0.884	0.904	0.502	0.768
action F1	0.866	0.940	0.394	0.520	0.858	0.919	0.471	0.588
EMBER (7B)	0.786	0.784	0.310	0.612	0.769	0.760	0.360	0.669
OOF combination	0.906	0.922	0.388	0.755	0.899	0.913	0.497	0.792
nested gate	0.907	0.925	0.390	0.755	0.899	0.914	0.496	0.792

The judge does improve ordering *within* an instruction: combined with overlap it raises within-instruction AUROC by $+0.015$ $[+0.006, +0.023]$ on four planners and $+0.009$ $[+0.005, +0.014]$ on eleven. Yet top-1 moves by $+0.0002$ $[-0.004, +0.004]$ and -0.0050 $[-0.010, -0.001]$. The audit says why. Only 672 of 1,641 instructions have both an executing and a failing candidate, and on those overlap's top choice already executes 90.0 % of the time. Replacing overlap by the combination changes the chosen plan in 305 instructions and changes its outcome in 49 (28 better, 21 worse); the rest of the within-instruction gain reorders candidates below the top. The judge corrects overlap where overlap's ranking does not decide anything. Overlap's top score is tied in 783 instructions, and ties are credited in expectation, so this is not a tie-breaking artefact.

L.3 THE JUDGE AGAINST A CHEAP COMPOSITE

The published complement adds the judge to LCS alone. A judge that adds information LCS lacks may add nothing to a stronger cheap composite, so we fixed the feature sets before fitting (`analysis/cheap_composite_increment.py`; exploratory, reference-conditioned): B0 is raw LCS; B1 adds action F1, length error, both plan lengths, symbolic validity, well-formedness, four declared-object grounding features and instruction-argument overlap, all cheap and text-only; B2 adds an instruction–plan embedding cosine; J1, J2 and J3 add the judge to LCS, B1 and B2. Every fitted arm is an out-of-fold logistic on the same trajectory-cluster folds and labels.

Table 19: AUROC, precision of the top $10\,\%$, and top-1 selection, both pools.

	4 planners			11 planners		
Arm	AUROC	prec.@10	top-1	AUROC	prec.@10	top-1
B0: LCS	0.890	0.709	0.388	0.884	0.768	0.502
B1: cheap text features	0.947	0.933	0.412	0.946	0.961	0.520
B2: B1 + embedding	0.950	0.918	0.414	0.950	0.957	0.520
J1: LCS + judge	0.906	0.755	0.388	0.899	0.792	0.497
J2: B1 + judge	0.952	0.929	0.414	0.952	0.956	0.521
J3: B2 + judge	0.954	0.921	0.415	0.954	0.947	0.522

The cheap composite is the strongest scorer we have: it lifts AUROC over LCS by $+0.057$ $[+0.048, +0.067]$, the precision of the top tenth by $+0.224$, and top-1 selection by $+0.023$ $[+0.017, +0.030]$, the only selection gain in this paper. Against it the judge adds $+0.0052$ $[+0.0029, +0.0075]$ AUROC on four planners and $+0.0064$ $[+0.0043, +0.0086]$ on eleven, and nothing detectable on either decision (top-1 $+0.0024$ $[+0.0000, +0.0055]$; precision -0.004 $[-0.026, +0.008]$). Most of what the judge adds to LCS, cheap signals already carry. Two cautions: every fitted arm needs execution labels, which raw LCS does not; and B1's grounding features read the task's declared objects, the same input the judge receives.

L.4 CONTROLLED, EXECUTION-VERIFIED ALTERNATIVES

Natural planner output cannot separate structure from meaning, because overlap and validity move together in it. So we rewrote every gold plan under fixed rules (`analysis/perturbation_plans.py`) and executed every rewrite: two rules meant to preserve meaning while lowering overlap, and four meant to break execution while keeping it. The design, the endpoint and its reading rule were registered before any rewrite was executed or judged (`amendments_perturbation_study`). A rule's intent never substitutes for its outcome: every pair is formed from measured execution. The judge is the published 7B rubric with its published inputs.

Table 20: The rewrites: how often each executes, and the mean score each evaluator gives it.

Rewrite	n	executes	LCS	action F1	EMBER
gold plan	1,641	0.882	1.000	1.000	0.873
valid: one redundant `goto`	1,641	0.881	0.854	1.000	0.847
valid: detour, every `goto` doubled	1,641	0.883	0.605	0.933	0.578
near: wrong target object	1,641	0.000	0.587	0.856	0.054
near: wrong final receptacle	1,374	0.020	0.684	0.892	0.259
near: final step dropped	1,641	0.036	0.826	0.948	0.719
near: first pick moved later	1,641	0.839	0.826	1.000	0.809

The registered endpoint is within-episode pairwise accuracy: over every pair of an executing meaning-preserving rewrite and a failing near-miss of the same episode ($8{,}166$ pairs, $1{,}456$ episodes, 455 trajectory clusters), how often a scorer ranks the rewrite higher, ties counting half. The judge does so 0.746 of the time and LCS 0.644: $+0.102$ $[+0.089, +0.116]$, which under the registered rule *supports the judge's defence*. On the $2{,}884$ pairs where overlap is wrong by construction – the rewrite has the lower LCS – the judge ranks them right 0.503 of the time, which is chance: it does not recover what overlap gets backwards, it only avoids LCS's systematic error there. Reference-free, the re-elicited judge beats the symbolic check on the same pairs ($+0.135$, $[+0.121, +0.149]$).

Three things bound that. First, an order-free overlap metric does better than the judge on the same pairs: action F1 ranks them right 0.877 of the time ($+0.130$ over the judge, $[+0.115, +0.146]$), and over all rewrites it predicts execution at AUROC 0.827 against the judge's 0.743 and LCS's 0.600. The meaning-preserving rules add redundant and detour steps, which LCS's length normalisation penalises and a set overlap ignores; the judge's advantage is over one overlap metric on one kind of rewrite, not over overlap. Second, one near-miss rule failed to break execution: moving the first pick after the next step still executes $83.9\,\%$ of the time under the oracle controller, and the executing ones are left out of the pairs, as the registration requires. Third, these are constructed contrasts; they show the judge *can* credit a valid alternative that LCS penalises, not that planners produce such alternatives often enough to matter. On natural output the judge still trails every overlap metric (Sec. 5.1), and on the $3{,}089$ natural plans a symbolic checker accepts at low overlap it is not detectably better (Sec. 5.2).

A second registration, without the new-object cue. Three of the first rules name an object outside the task's declared list – the detour, the wrong object, the wrong receptacle – and the judge's prompt tells it to distrust such objects, so its lead could be a cue rather than an understanding. A second registration (`amendments_perturbation_v2`, fixed before execution) added two rules that name no new object: every `goto` repeated (executes $88.1\,\%$), and the first pick moved to just after the step that needs it held (executes $0.0\,\%$, so a genuine precondition violation), and restricted the pairs to rewrites without new objects ($5{,}164$ pairs). The judge ranks the executing rewrite higher 0.579 of the time against LCS's 0.500 ($+0.079$, $[+0.060, +0.097]$): by the registered rule the defence still holds, at under half the margin of the first study once the cue is gone. On the precondition violation alone the judge is at 0.549, near chance, and the symbolic check, which never sees the gold plan, flags every such rewrite. Across these pairs action F1 reaches 0.684 ($+0.106$ over the judge, $[+0.083, +0.129]$), the symbolic check 0.732, and the reference-free judge 0.395, below chance (-0.337 against the check, $[-0.356, -0.319]$).

L.5 The arbitration thresholds, swept

Figure 1(c) and the $+0.402$ within-stratum gap rest on four cut-offs that were chosen, not preregistered: the judge is confident above 0.7 and doubtful at or below 0.1, the reference condemns at or below 0.3 and rates highly at or above 0.6. The rejected arm at the published setting holds 84 plans. A reader is entitled to ask what the neighbouring cut-offs say, so we swept them.

Over 20 cells bracketing the published values in both directions on all three axes, the gap is positive in *every* one, ranging $+0.255$ to $+0.675$ with a median of $+0.438$. The published cell sits at $+0.402$, close to that median rather than at an extreme. 16 of the 20 have an interval excluding zero; the rest are the cells whose rejected arm shrinks towards the grid's corner, where it falls as low as 6 plans. The finding is a property of the data and not of the corner we reported, but the arm size is the real limit on it, and it does not improve anywhere in the grid: the largest rejected arm here is 358. Two disclosures about the method. Cells whose thresholds select identical arms are the same experiment, so they are collapsed. The published cell's interval here comes from this sweep's own per-cell seed and so differs in the third decimal from Sec. efsec:disagree's $[+0.274, +0.512]$, which is the published number: 44 of the grid's triples were duplicates, which an earlier version of this table counted separately. And a bootstrap replicate that empties an arm leaves the difference undefined and is dropped, which conditions the interval on replicates that kept both arms populated and so narrows it; the worst-affected cell, the one with the six-plan arm, loses 342 of $10{,}000$.

M Why the judge goes wrong: a failure taxonomy

Sec. 5.2 asks *where* the judge's verdicts diverge from execution. This asks *why*. We classify every disagreement into a taxonomy of judge-failure modes, fixed before the labelling and applied to the same within-stratum disagreements, so that the classes partition the disagreements rather than being read off the ones that were easiest to name.

The reading is the same in both disagreement strata: the plans the judge credits fail for the reasons the plans everyone condemns fail, only less often. Half of them (131 of 259) name an object or receptacle the task never declared – a grounding failure at $50.6\,\%$ against $33.3\,\%$ across all rollouts – which is exactly the check the grounding-aware free composite of Appendix P performs, and exactly

Table 21: Threshold sweep, the five widest and five narrowest gaps of 20. Gap is within-stratum execution for plans the judge credits minus plans it rejects, with a trajectory-cluster interval; n_{rej} is the smaller arm. The published setting is marked. Regenerated by `analysis/threshold_sweep.py`.

Thresholds	Gap	95 % CI	n_{rej}	
`lcs≥0.5,judge≥0.9,judge≤0.35`	+0.675	[+0.591, +0.753]	358	
`lcs≥0.5,judge≥0.9,judge≤0.1`	+0.675	[+0.591, +0.751]	357	
`lcs≥0.5,judge≥0.8,judge≤0.35`	+0.562	[+0.504, +0.615]	358	
`lcs≥0.5,judge≥0.8,judge≤0.1`	+0.562	[+0.505, +0.617]	357	
`lcs≥0.6,judge≥0.8,judge≤0.1`	+0.554	[+0.427, +0.666]	84	
`lcs≥0.6,judge≥0.7,judge≤0.1`	+0.402	[+0.277, +0.514]	84	**published**
`lcs≥0.5,judge≥0.7,judge≤0.1`	+0.390	[+0.338, +0.439]	357	
`lcs≥0.8,judge≥0.9,judge≤0.1`	+0.282	[−0.069, +0.927]	6	
`lcs≥0.8,judge≥0.8,judge≤0.1`	+0.270	[−0.081, +0.908]	6	
`lcs≥0.8,judge≥0.7,judge≤0.1`	+0.256	[−0.096, +0.889]	6	
`lcs≥0.8,judge≥0.6,judge≤0.1`	+0.255	[−0.097, +0.893]	6	

Table 22: Why the credited plans fail, by rule from stored data. Columns are the stratum the judge credits and overlap condemns (EMBER $\geq$ 0.7, LCS $\leq$ 0.3), the stratum both condemn, and all rollouts. Classes overlap by construction, so the first block does not sum to 100 %; the second block assigns each plan one class under a fixed precedence (grounding > symbolic > executor > malformed). The classes are rules over the plan text, not verdicts on the rollout: 2 of the 259 credited plans do execute, one counted under grounding (it names an undeclared object and runs anyway) and one unclassified. The malformed class covers empty plans, unparseable steps and verbs outside the DSL; in the credited column it is only the last two, since empty plans score 0 deterministically and cannot be credited.

	judge credits, overlap condemns (n=259)	both condemn (n=893)	all rollouts (n=6,564)
overlapping classes			
grounding (argument outside the declared objects)	50.6 %	62.6 %	33.3 %
symbolic precondition violation	16.6 %	13.5 %	10.1 %
executor error recorded	71.4 %	66.2 %	69.3 %
empty or malformed plan	44.0 %	55.8 %	18.0 %
primary class			
grounding	131	559	2,189
symbolic	21	29	370
executor	61	149	2,478
malformed	45	156	393
unclassified	1	0	1,134

what a rubric asked for "semantic equivalence" does not perform. The judge's credited stratum is, if anything, *more* grounding-broken than the population it is drawn from. Two exemplars, both credited at EMBER $\geq$ 0.7 by the 7B judge: a plan for "move a rag from cabinet to the tub" that routes through `tubbasin` where the task declares `bathtubbasin`, which the executor rejects with "no instance of `tubbasin` in scene"; and a plan for "put a cut apple in a pot" that slices before picking up the knife, which the symbolic checker flags and the executor refuses. Four cautions. The declared objects are the task's two or three PDDL arguments, not a scene inventory, so the grounding class fires on correct plans too – which is why the all-rollout column is there, and why the class must be read as a contrast between strata rather than as an error count. The classes overlap. The precedence in the second block is a reporting convenience, not a causal claim. And 2 of the 259 credited plans do execute; they are classed with the rest, so both counts and percentages are taken over the full stratum of 259.

One methodological note on the exemplars. Fig. 1's first version chose its two cases by judge score, before the rollouts existed, and both later contradicted the reading they had been picked to illustrate. We replaced them with exemplars selected once their outcome was known, and say so in the caption:

anecdotes chosen by the signal under test confirm it. Exemplar (a) was then re-selected a second time, under a rule added after Appendix E.2 showed that alias misses are the executor's rigidity rather than the judge's error: candidates whose recorded failure is an alias miss are excluded, and the existing ordering is applied to the rest. The previous exemplar, a plan that failed only because `bathroomcabinet` is not how the scene names its cabinet, is exactly the case a judge is entitled to credit.

N GIVING THE JUDGE THE ROOM

The judge of Sec. 4 sees the task's two or three declared arguments, never the scene, and Appendix M finds that half of the plans it credits against the reference name an object the task never declared. The natural objection is that the judge is under-informed rather than incapable. We tested it under a preregistered reading rule (`amendments_judge_scene_inventory`): the same 7B judge, the same rubric and parsing, with one block added to the prompt listing every object type in the episode's AI2-THOR scene – the fixed objects of the floorplan read from the simulator plus the movables the episode places, 36 types per scene on average across all 1,641 episodes. Both conditions were re-elicited on the primary pool (n=6,564 each, 0 parse failures).

Table 23: The judge with the scene inventory in its prompt, primary pool. Deltas are paired trajectory-cluster bootstraps (2,000 resamples). The main rubric's own deficit recomputed on the same rows is -0.104 $[-0.126, -0.083]$.

condition	inventory judge	main rubric	$\text{EMBER}_{\text{inv}}-\text{LCS}$ (95 % CI)	$\text{EMBER}_{\text{inv}}-\text{main}$ (95 % CI)	ρ vs main
gold plan shown	0.733	0.786	-0.156 $[-0.183, -0.129]$	-0.052 $[-0.071, -0.034]$	0.69
reference-free	0.705	0.710	-0.185 $[-0.211, -0.159]$	-0.005 $[-0.022, +0.012]$	0.77

The room does not help. With the gold plan shown, the deficit against overlap widens from -0.104 to -0.156 $[-0.183, -0.129]$, and the inventory judge trails the main rubric itself by -0.052 $[-0.071, -0.034]$; reference-free the two judges are indistinguishable (-0.005 $[-0.022, +0.012]$). Under the rule fixed before the run – an interval still entirely below zero, and a deficit not halved – this inventory does not close the loss. That is a result about one representation of the room (object types, no instances, states, containment or spatial relations) in one prompt; it does not show that no information could close it.

Nor does it stop the judge crediting objects the room lacks. On the stratum the judge credits while overlap condemns, the inventory judge credits 524 plans against the main rubric's 259 on the same rows, of which 3 execute; 258 of the 524 (49.2 %) name an object outside the inventory it was just shown, against 46.3 % for the main rubric measured the same way (its 50.6 % in Appendix M is against the declared arguments). The misses are near-synonyms a checker flags and a judge waves through: a plan sent to `tub` in a room whose list says `bathtub`, to `refrigerator` where the list says `fridge`.

The complement shrinks with it. Substituting the inventory judge into the nested gate of Appendix L gives $+0.008$ $[+0.004, +0.012]$ at 53 % of the calls, against the main rubric's $+0.017$ $[+0.011, +0.023]$ on the same rows. Reference-free, the inventory judge (0.705) still beats symbolic validity by $+0.050$ $[+0.026, +0.074]$ and still trails the pooled free composite by -0.084 $[-0.106, -0.060]$, so nothing in Sec. 5.9 moves.

O EVERY REFERENCE-FREE COMPARATOR

The reference-free comparison in Sec. 5.4 names the best baseline; here is the whole list, on the primary pool, with the gold-requiring evaluators for context. The reconstructed row uses components elicited *with* the gold plan and is therefore an upper bound on a post-hoc reference-free evaluator (the re-elicited figure is 0.710).

O.1 THE REFERENCE-FREE LEAD, WITH RESOLUTION TAKEN AWAY

The preregistered reference-free advantage is measured against a symbolic validity check, and that check is binary. AUROC scores ties at half credit, so a two-valued predictor cannot order plans

Table 24: Execution AUROC of every evaluator computable without a gold plan, and of the three that need one.

Evaluator	AUROC
EMBER reconstructed ($c \cdot v \cdot e$)	0.787
symbolic validity	0.655
DSL well-formedness	0.591
plan length (negated)	0.523
coverage alone	0.593
validity alone	0.566
EMBER full rubric (needs gold)	0.786
LCS (needs gold)	0.890
action F1 (needs gold)	0.866

within either of its two values; a perfect binary predictor still reaches 1, but an imperfect one loses the within-class ordering a many-valued score may carry, and the comparison then mixes validity with resolution. This paper makes that argument elsewhere, to explain why a checklist rubric variant loses, and it owes the same courtesy to the baseline it is beating.

The gap is narrower than the objection assumes. In the reference-free setting the judge's score takes 3 distinct values against the check's 2, so there is little resolution to strip. We strip it anyway, cutting the judge at the threshold that maximises Youden's J (1.00 on average, chosen out-of-fold) so that both predictors are two-valued:

	Judge	Symbolic check	Δ (95 % CI)
As published (continuous judge)	0.7096	0.6551	$+0.0545$ $[+0.0359, +0.0722]$
Resolution matched (binarised)	0.6917	0.6551	$+0.0366$ $[+0.0161, +0.0559]$
Balanced accuracy, own cut	0.6900	0.6551	$+0.0350$

Roughly a third of the published $+0.055$ is resolution and the rest is judgment. The matched comparison keeps the sign and excludes zero, so the preregistered claim is not a measurement artifact, but it is smaller than the headline number implies and we report both.

P THE GROUNDING-AWARE FREE COMPOSITE, WITH ITS WEIGHTS

Sec. 5.4 reports, where no reference exists, a fixed composite of signals computable from what the judge itself is given: the instruction, the task's declared objects and the plan. Fixing it means publishing it. The nine features are symbolic validity, DSL well-formedness, negative plan length, the grounded fraction of plan arguments and its lenient variant, an all-grounded indicator, the negative count of ungrounded arguments, instruction–plan cosine and instruction–argument overlap. Each is standardized by the mean and standard deviation in Table 25, combined as $\text{score} = b + \sum_k w_k(x_k - \mu_k)/\sigma_k$ and passed through a logistic. The coefficients were fitted once on the primary pool, rounded to two decimals, and then never refitted.

A three-feature version, refitted on the same pool under the same protocol, can be computed by hand:

$$\begin{aligned}\text{score} = -2.04 &+ 1.27\, z(\text{symbolic validity}) \\ &+ 1.00\, z(\text{grounded fraction}) \\ &- 0.03\, z(\text{well-formedness}).\end{aligned}$$

It reaches AUROC 0.713 $[0.690, 0.735]$ out of fold against the nine-feature 0.789 $[0.770, 0.808]$, and its direction is the readable one: a plan predicts execution when the symbolic checker accepts it and its arguments name things the task declared. Its weights are not a subset of the nine-feature weights: dropping the other six moves well-formedness from $+0.09$ to -0.03, so it carries almost nothing once symbolic validity and grounding are in.

Four cautions, and they bound the recommendation. The formula is reference-free but not label-free: it is fitted on execution labels, so the transfer result shows that the *weights* carry to new planners, not that the composite could have been written without supervision. The declared objects are the

Table 25: The frozen nine-feature composite (intercept -2.50), and what the frozen formula scores on planner families it was never fitted on. Transfer families are scored on the same $1{,}641$ ALFRED episodes: the planner changes, the tasks, scenes and controller do not.

Feature	w	μ	σ
symbolic validity	1.35	0.7307	0.4436
well-formedness	0.09	0.9406	0.1663
negative plan length	0.27	−5.0888	2.2810
grounded fraction	2.82	0.8822	0.2025
grounded fraction (lenient)	−0.73	0.9057	0.1831
all arguments grounded	−1.04	0.6587	0.4741
negative ungrounded count	0.72	−0.7633	1.4596
instruction–plan cosine	0.48	0.4997	0.1014
instruction–argument overlap	−0.03	0.5741	0.3024
AUROC, nine-feature / three-feature			
primary pool, out of fold (n=6,564)		0.789 / 0.713	
Qwen3, frozen (n=6,564)		0.818 / 0.721	
Mistral, frozen (n=1,641)		0.806 / 0.714	
Qwen2.5-VL, frozen (n=3,282)		0.741 / 0.637	

task's two or three PDDL arguments, so the grounding features measure agreement with a task hint rather than with the scene inventory. The four grounding features are strongly collinear – they are four readings of one quantity – so their individual signs are not separately interpretable and the three-feature version is the one to read for direction; the nine-feature formula is a predictor, not a decomposition. Finally, the transfer is across planner families on one benchmark, not across benchmarks, and the gold-free judge was only ever run on the four Qwen2.5 planners, so on those three families there is no judge number to compare the composite against.

Q Asking the judge the criterion's own question

The rubric asks for coverage, validity, equivalence and efficiency and predicts execution through that composite. A reviewer will ask whether the deficit is the rubric's fault. So we asked the same primary 7B judge, with no rubric, for a single probability that the plan achieves its instruction in the scene – the task's declared object and receptacle arguments, the instruction and the plan in the prompt; in one variant the gold plan as well – at temperature 0, on the primary pool.

Table 26: Execution AUROC of the direct probability against the rubric judge and overlap, with paired trajectory-cluster intervals, and what the direct score adds to LCS out of fold.

Variant	direct	EMBER†	LCS	direct − EMBER	direct − LCS	added to LCS
reference-free	0.619	0.710	0.890	-0.090 $[-0.112, -0.069]$	-0.270 $[-0.297, -0.245]$	0.000 $[-0.002, +0.003]$
gold plan shown	0.662	0.786	0.890	-0.124 $[-0.151, -0.096]$	-0.228 $[-0.255, -0.200]$	$+0.003$ $[+0.000, +0.007]$

Asked the criterion's own question, the judge does worse, not better: 0.619 reference-free against the rubric's re-elicited reference-free 0.710, and 0.662 with the gold plan shown against the rubric's gold-conditioned 0.786 (overlap: 0.890); each row's rubric comparator saw the same reference the direct prompt did, and the direct score adds nothing to LCS. The 7B model answers with only 12 and 11 distinct probabilities. The rubric is not the source of the deficit; it is what makes the judge's execution signal usable at all. Whether a frontier judge answers the direct question better is untested here.

R Contamination probe

ALFRED has been public since 2020, so the frontier judge's reference-free advantage invites the objection that it is recall rather than judgment. The reference-free condition removes the gold plan from the prompt but not from the model's weights, so the objection needs data.

By compositional split. The advantage holds in both halves of our compositional hold-out, seen ($1{,}237$ episodes) and unseen composition (404), with overlapping intervals (Table 27). This is our

object–receptacle split, not ALFRED's floorplan split (Appendix K). We report it for completeness but do not lean on it: both halves are equally public text, so novelty of an object–receptacle pairing to our planners says nothing about novelty to a pretrained LLM.

Table 27: Reference-free advantage over the symbolic baseline, by compositional split.

Judge	Split	n	Judge / symbolic AUROC	Δ (95 % CI)
gpt-5.1	`seen`	4,948	0.777 / 0.652	$+0.125$ $[+0.103, +0.147]$
gpt-5.1	`unseen_composition`	1,616	0.759 / 0.666	$+0.093$ $[+0.032, +0.151]$
Qwen2.5-7B	`seen`	4,948	0.691 / 0.652	$+0.039$ $[+0.017, +0.059]$
Qwen2.5-7B	`unseen_composition`	1,616	0.768 / 0.666	$+0.102$ $[+0.062, +0.140]$

Direct recall, the discriminating test. ALFRED's object vocabulary is idiosyncratic (`butterknife`, `sinkbasin`, `countertop`) and its routing choices are arbitrary, so a model that had memorized the corpus would reproduce them exactly. Asked for the gold plan from the instruction alone, the judge reproduced 0 of 150 (0.0 %), and paraphrased precisely those arbitrary details – `knife` for `butterknife`, `sink` for `sinkbasin`, and `goto(mug)` where the gold plan routes `goto(shelf)`. Our strongest local planner exact-matches the gold plan on 2.7 % of the same episodes. This is evidence against contamination, not proof of its absence. Only the exact-match figure is cited: the verb-sequence rate is unstable between runs because the hosted judge rejects `temperature`$= 0$.

What this test rules out is verbatim recall of gold plans. It does not rule out the weaker and more likely form: familiarity with ALFRED's object vocabulary and its routing idioms, acquired from the dataset's wide circulation, which would help a judge recognise a well-formed ALFRED plan without reproducing any particular one. We cannot separate that from competence with the data we have, and it would inflate the frontier judge's standalone number rather than the deficit, which is measured against a reference drawn from the same distribution.

S A PERMUTATION TEST OF THE CAPABILITY TREND

Six points cannot carry a correlation on their own, so we give the trend an exact permutation test over all $6! = 720$ orderings, and ask whether family still predicts anything once capability is held fixed. Degrees of freedom are small and we read this as weak support, not proof. Table 28 lists the six judges and Table 29 the permutation test.

Table 28: The six judges behind the trend. Weak *same*-family judges lose the advantage exactly as the cross-family judge does, and the frontier cross-family judge has the largest gains of all, which is why we read the trend as capability rather than family. Gains are execution-AUROC deltas over LCS (gated) and over the symbolic baseline (reference-free). The reference-free column is the post-hoc reconstruction, which the two re-elicited judges show inflates the advantage by different factors ($2.4\times$ for Qwen2.5-7B, $1.4\times$ for gpt-5.1), so it is ordinal support only and not comparable to the re-elicited $+0.119$ in the main text. Exploratory.

Judge	Family	Standalone	Gated	Ref.-free
Qwen2.5-1.5B	same	0.429	$+0.002$	-0.151
Qwen2.5-3B	same	0.660	$+0.009$	-0.060
Mistral-7B	cross	0.694	$+0.002$	-0.092
Qwen2.5-7B	same	0.786	$+0.017$	$+0.132$
Qwen2.5-32B	same	0.810	$+0.016$	$+0.134$
gpt-5.1	cross	0.837	$+0.021$	$+0.167$

Table 29: Capability trend across the six judges, and family's contribution once capability is controlled.

Gain	Pearson r	exact perm. p	Spearman ρ	capability β (s.e.)	family β (s.e.)
Gated	$+0.831$	0.0278	$+0.886$	$+0.0492$ (0.0167)	$+0.0038$ (0.0049)
Reference-free	$+0.889$	0.0069	$+0.943$	$+0.8825$ (0.2279)	$+0.0600$ (0.0664)

Family adds nothing once capability is in the model, but with three degrees of freedom that absence is not itself decisive.

S.1 A PUBLISHED RUBRIC JUDGE: PROMETHEUS 2 ON THE SAME ROLLOUTS

Every judge in the body is a general-purpose base model under one hand-written rubric, and a reviewer can reasonably ask whether the execution deficit is a property of LLM judging or of that rubric on those models. Prometheus 2 (7B) (Kim et al., 2024b) is a model fine-tuned to grade responses against a rubric, and it is the system Related Work already names as the form EMBER shares. So we ran it, on the same $6{,}564$ rollouts, under its own published absolute-grading template with a 1–5 rubric that restates EMBER's four dimensions in the plain-language form it expects. The rubric, the template and the reading rule were fixed in the preregistration (`amendments_prometheus_judge`) before a single verdict existed. The rule: on the reference-conditioned condition, an interval for (Prometheus − LCS) entirely below zero with a point estimate at or below -0.052 means the deficit is not specific to our rubric or base models; anything else means a published judge closes most of the gap and the paper's framing narrows.

Table 30: Prometheus 2 (7B) against the same criterion. AUROC for ranking execution success on the primary pool; paired trajectory-cluster intervals, $10{,}000$ resamples. Regenerated by `analysis/prometheus_judge.py`.

Condition	Prometheus 2	EMBER (7B)	LCS	Prom. − LCS	Prom. − EMBER
Reference shown	0.757	0.785	0.890	-0.133 $[-0.151, -0.114]$	-0.028 $[-0.052, -0.005]$
Reference withheld	0.574	0.710	0.890	-0.316 $[-0.338, -0.294]$	-0.136 $[-0.159, -0.112]$

With the reference shown, Prometheus 2 reaches 0.757 against LCS's 0.890, a paired -0.133 $[-0.151, -0.114]$, on $6,559$ rollouts (5 verdicts unparseable and excluded, 0 parsed from a loosely formatted answer). Its scores fall on the five rubric levels as 1: 2,095, 2: 1,033, 3: 71, 4: 2,961, 5: 399. Under the preregistered rule this is the first outcome: the deficit is not specific to our rubric or our base models. A judge trained to grade against rubrics, run under its own template, trails the one-line overlap metric on the same criterion.

The reference-withheld score is close to chance. That is not the template breaking: every verdict in that condition parsed, none needed a loose parse, and all five rubric levels were used, with the mass shifting to 4 and 5. Prometheus 2 without a reference rates most plans as good, which is the failure the reference-free EMBER arm shares in milder form.

Two things this arm does not settle. It is one published judge at one size, and Prometheus 2 was trained on natural-language responses rather than action plans in a DSL, so a domain-tuned grader could behave differently. And the rubric levels are ours, restating EMBER's dimensions; a rubric written without reference to EMBER might grade differently, which is why the levels are printed in the preregistration rather than only described here.

T JUDGE AUDITS IN FULL

If the judge is used at all, its biases must be known. **Self-preference:** a 2×2 planner-family $\times$ judge-family design finds the Qwen judge inflates its own family's advantage over the Mistral judge's (the 2×2 means and the test are below); both still rank the Qwen planner first, so rankings survive. The raw interaction is a difference of EMBER levels across two judges, which this appendix elsewhere says must not be compared, so we also report it after normalising within each judge: $+0.240$ of that judge's standard deviations (p=5.7×10^{-11}) and $+0.042$ in within-judge quantiles (p=2.1×10^{-5}). The effect is not an artifact of scale, and it is large enough to warrant a judge disjoint from all planners under test. **Compositional generalization gap:** The seen-minus-unseen gap measures how much harder novel compositions are, not when a capability appears; it is present in Qwen2.5 with no counterpart at any Qwen3 scale, so we report it as a difficulty measure and retract any emergence reading; lenient/strict variants of the same judge also move its onset, making that a property of the configuration, not the planner. **Degenerate inputs:** given an empty plan, smaller judges hallucinate – the 7B judge scored 41 of the 47 one planner emitted at ≥ 0.5 and Mistral 38, while the 32B judge scored all 47 at zero. (The primary pool holds 49 empty plans, 47 from that planner and 2 from another; the audit uses the 47 so the comparison is within one planner.) We score

empty plans deterministically and corrected 510 stored judgments (counts from the decision log; the pre-guard caches were rewritten in place); robustness to degenerate inputs is itself capability-dependent. **Cross-judge agreement:** continuous scores correlate only weakly across the three judge pairs (Table 38) even though within-judge planner rankings replicate, so absolute EMBER levels must never be compared across judges and a benchmark release must fix one.

The compositional gap, by planner scale and by judge. Table 31 gives the seen-minus-unseen gap for every planner scale under each judge; the reading it supports is in the paragraph after it.

Table 31: The compositional generalization gap (seen minus unseen-composition EMBER, 95 % trajectory-cluster CI) by planner scale and judge. It appears under the 7B judge at 1.5B only, moves under the 32B judge, and has no counterpart at any Qwen3 scale, which is why we retracted the emergence reading.

Judge	Planner	gap	95 % CI
Qwen2.5-7B	Qwen2.5-0.5B	$+0.016$	$[-0.026, +0.059]$
Qwen2.5-7B	Qwen2.5-1.5B	$+0.053$	$[+0.010, +0.095]$
Qwen2.5-7B	Qwen2.5-3B	$+0.036$	$[-0.004, +0.074]$
Qwen2.5-7B	Qwen2.5-7B	$+0.026$	$[-0.010, +0.062]$
Qwen2.5-32B	Qwen2.5-0.5B	-0.008	$[-0.039, +0.021]$
Qwen2.5-32B	Qwen2.5-1.5B	$+0.007$	$[-0.036, +0.050]$
Qwen2.5-32B	Qwen2.5-3B	$+0.077$	$[+0.026, +0.126]$
Qwen2.5-32B	Qwen2.5-7B	$+0.068$	$[+0.014, +0.121]$
Qwen2.5-7B	Qwen3-0.6B	$+0.041$	$[-0.002, +0.082]$
Qwen2.5-7B	Qwen3-1.7B	-0.001	$[-0.047, +0.046]$
Qwen2.5-7B	Qwen3-4B	-0.006	$[-0.044, +0.033]$
Qwen2.5-7B	Qwen3-8B	$+0.019$	$[-0.019, +0.057]$

Table 32: Gap onset under lenient, default and strict variants of the same 7B judge (Qwen2.5 planners). The scale at which the gap's interval first excludes zero moves with the rubric's stringency, so onset is a property of the evaluator's configuration.

Variant	Planner	gap	95 % CI	EMBER mean
lenient	Qwen2.5-0.5B	$+0.025$	$[-0.021, +0.071]$	0.437
lenient	Qwen2.5-1.5B	$+0.010$	$[-0.028, +0.047]$	0.569
lenient	Qwen2.5-3B	$+0.019$	$[-0.011, +0.049]$	0.676
lenient	Qwen2.5-7B	$+0.035$	$[+0.008, +0.064]$	0.698
default	Qwen2.5-0.5B	$+0.016$	$[-0.026, +0.059]$	0.294
default	Qwen2.5-1.5B	$+0.053$	$[+0.010, +0.095]$	0.471
default	Qwen2.5-3B	$+0.036$	$[-0.004, +0.074]$	0.578
default	Qwen2.5-7B	$+0.026$	$[-0.010, +0.062]$	0.614
strict	Qwen2.5-0.5B	$+0.005$	$[-0.041, +0.051]$	0.238
strict	Qwen2.5-1.5B	$+0.042$	$[-0.010, +0.094]$	0.394
strict	Qwen2.5-3B	$+0.034$	$[-0.014, +0.080]$	0.518
strict	Qwen2.5-7B	$+0.022$	$[-0.020, +0.065]$	0.570

Self-preference, the 2×2. Mean EMBER over the same 506 trajectory clusters: Qwen planner under the Qwen judge 0.614, under the Mistral judge 0.821; Mistral planner under the Qwen judge 0.511, under the Mistral judge 0.784. The interaction (difference in differences) is $+0.069$, cluster Wilcoxon $p{=}2.1 \times 10^{-13}$.

U CAPABILITY-LEVEL SCALING, AND WHY ITS SHAPE BELONGS TO THE JUDGE

The rubric decomposes plan quality into coverage, validity, equivalence and efficiency ($e = 1 - r/2$), all scored here by the primary Qwen2.5-7B judge. Aggregating each by planner scale on the unseen-composition split asks a different question from the rest of the paper: not whether the judge *ranks* plans correctly, but how the capabilities it scores move with model size. Episode-level discriminative failure and a population-level trend are separate things, and the second can be informative even where the first is not.

What one judge shows. Under the primary 7B judge (Table 33) coverage, validity and equivalence rise with scale in both Qwen2.5 and Qwen3, and efficiency is flat in both. We present this analysis in full and then show, below, that its *shape* does not survive a change of judge; the reader should hold the 7B reading provisionally throughout. The seen-minus-unseen *gap* did not replicate across families; the direction of these curves does, under this judge.

Table 33: Capability means by planner scale on unseen composition (n=404 per cell), with trajectory-cluster bootstrap 95 % intervals.

Family	Scale (B)	Coverage	Validity	Equivalence	Efficiency
Qwen2.5	0.5	0.589 [0.530, 0.646]	0.745 [0.695, 0.794]	0.464 [0.437, 0.491]	0.608 [0.581, 0.634]
Qwen2.5	1.5	0.757 [0.713, 0.804]	0.896 [0.864, 0.925]	0.539 [0.517, 0.565]	0.660 [0.630, 0.690]
Qwen2.5	3.0	0.901 [0.867, 0.931]	0.931 [0.901, 0.958]	0.613 [0.587, 0.639]	0.639 [0.605, 0.674]
Qwen2.5	7.0	0.928 [0.900, 0.955]	0.975 [0.953, 0.992]	0.632 [0.607, 0.655]	0.647 [0.614, 0.679]
Qwen3	0.6	0.616 [0.559, 0.673]	0.891 [0.860, 0.921]	0.493 [0.470, 0.517]	0.637 [0.611, 0.665]
Qwen3	1.7	0.792 [0.742, 0.843]	0.926 [0.898, 0.953]	0.595 [0.569, 0.621]	0.646 [0.615, 0.676]
Qwen3	4.0	0.916 [0.881, 0.947]	0.983 [0.969, 0.995]	0.627 [0.602, 0.652]	0.636 [0.607, 0.666]
Qwen3	8.0	0.918 [0.889, 0.945]	0.990 [0.976, 1.000]	0.642 [0.617, 0.667]	0.650 [0.618, 0.683]

Smooth scaling or a transition? We fitted each capability twice over the pooled eight scale points: once as a single trend linear in $\log_2$ parameters, and once as a continuous piecewise-linear trend with one breakpoint. A segmented fit is preferred in every case, and under this judge every slope change is negative (Table 34): rising-then-flat, the shape of a ceiling, not the flat-then-rising shape a threshold transition would show (Wei et al., 2022) – a shape whose reality is itself contested (Schaeffer et al., 2023). We do not read this as a finding about the planners, for the reason the 32B re-scoring below makes plain.

Table 34: Smooth versus segmented fits over eight pooled scale points. ΔAIC favours the segmented fit throughout, but every slope change is negative.

Capability	R^2 linear	R^2 segmented	ΔAIC	slope before	slope after (per doubling)
Coverage	0.931	0.995	−19.71	+0.121	+0.022
Validity	0.751	0.966	−13.95	+0.511	+0.029
Equivalence	0.914	0.963	−4.86	+0.057	+0.014
Efficiency	0.296	0.747	−6.18	+0.136	+0.001

This comparison is under-powered by construction – eight points against a four-parameter segmented model – so it cannot locate a threshold, and we do not report one. It is reported for the sign of the slope change, which is unambiguous and identical across all four capabilities.

Table 35: Curvature within each family, without pooling. Leading coefficient of a quadratic in $\log_2$ parameters fitted to that family's four points; negative is decelerating. Intervals resample each planner's trajectory clusters and refit.

Family	Capability	curvature (95 % CI)	reading
Qwen2.5	coverage	−0.0152 [−0.0284, −0.0023]	decelerating
Qwen2.5	validity	−0.0130 [−0.0226, −0.0031]	decelerating
Qwen2.5	equivalence	−0.0055 [−0.0125, +0.0021]	indistinguishable from zero
Qwen2.5	efficiency	−0.0065 [−0.0155, +0.0027]	indistinguishable from zero
Qwen3	coverage	−0.0204 [−0.0353, −0.0058]	decelerating
Qwen3	validity	−0.0014 [−0.0088, +0.0062]	indistinguishable from zero
Qwen3	equivalence	−0.0115 [−0.0200, −0.0029]	decelerating
Qwen3	efficiency	+0.0006 [−0.0089, +0.0102]	indistinguishable from zero

The pooled fit treats nominal parameter count as a common axis across two families that differ in training recipe and generation, an assumption we have not validated. So Table 35 asks the same question inside each family, where a segmented model cannot be fitted (four points, four parameters) but a quadratic can. No cell shows positive curvature with an interval excluding zero; four of eight are negative with intervals excluding zero; and the efficiency curvature is indistinguishable from zero in both families, as a flat trajectory should be. The 7B reading therefore does not rest on the pooling; whether it rests on the judge is the next question.

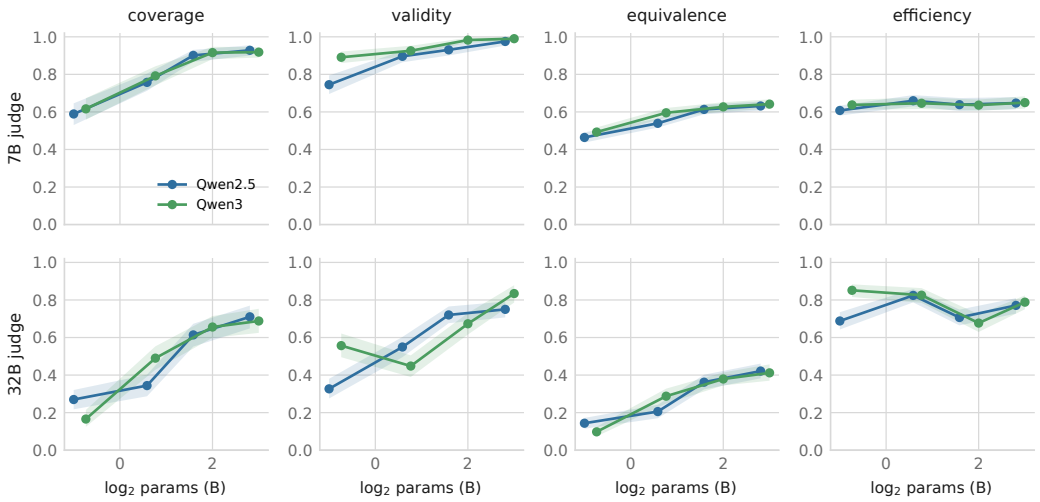

Figure 7: Capability means by planner scale under the 7B judge (top) and the 32B judge (bottom), both families, with trajectory-cluster intervals. The 7B validity curves sit at their ceiling; under the 32B judge Qwen2.5 coverage is flat-then-rising and Qwen3 validity non-monotone.

The shape does not survive a change of judge. Everything above is scored by the primary 7B judge. Absolute component levels are judge-specific (Appendix T), so we re-scored the same eight planners' plans with the 32B judge and repeated the analysis.

Table 36: The same plans under the 32B judge: capability means by planner scale on unseen composition (n=404 per cell), with the same trajectory-cluster intervals as Table 33, so a non-monotone step can be read against its noise.

Family	Scale (B)	Coverage	Validity	Equivalence	Efficiency
Qwen2.5	0.5	0.270 [0.22, 0.32]	0.327 [0.28, 0.38]	0.144 [0.11, 0.17]	0.688 [0.64, 0.73]
Qwen2.5	1.5	0.344 [0.29, 0.40]	0.550 [0.50, 0.60]	0.206 [0.17, 0.24]	0.824 [0.79, 0.86]
Qwen2.5	3	0.614 [0.55, 0.67]	0.720 [0.68, 0.77]	0.362 [0.32, 0.40]	0.707 [0.66, 0.75]
Qwen2.5	7	0.710 [0.65, 0.77]	0.750 [0.71, 0.79]	0.421 [0.38, 0.46]	0.771 [0.73, 0.81]
Qwen3	0.6	0.166 [0.12, 0.21]	0.557 [0.50, 0.62]	0.098 [0.07, 0.12]	0.851 [0.82, 0.88]
Qwen3	1.7	0.490 [0.43, 0.55]	0.448 [0.39, 0.50]	0.288 [0.25, 0.33]	0.825 [0.78, 0.86]
Qwen3	4	0.656 [0.60, 0.71]	0.673 [0.62, 0.73]	0.379 [0.34, 0.41]	0.677 [0.63, 0.72]
Qwen3	8	0.688 [0.62, 0.75]	0.834 [0.79, 0.88]	0.412 [0.37, 0.46]	0.788 [0.75, 0.82]

The two judges barely agree episode by episode: Spearman correlations of the component scores on the same 3,232 unseen episodes are 0.40 (coverage), 0.25 (validity), 0.47 (equivalence) and 0.26 (efficiency), and the 32B judge is far stricter (coverage mean 0.49 against 0.80). Under it the reading reverses in places. Pooled, coverage's slope under the 32B judge goes from -0.245 before the breakpoint to $+0.147$ after – accelerating, where the 7B judge decelerates (Table 34) – and the segmented model is preferred for efficiency only. Within families, Qwen3 validity and Qwen3 efficiency accelerate with intervals excluding zero, where under the 7B judge no cell did. Qwen2.5 coverage reads 0.27, 0.34, 0.61, 0.71 across scale, flat-then-rising, the very shape the 7B judge showed nowhere.

We therefore withdraw the saturation reading as a finding about the planners. The 7B judge scores validity at 0.75–0.99 across scale and above 0.89 for every planner but the smallest, at its ceiling, and a ceiling decelerates every curve mechanically; the 32B judge, with room above it, shows something else. What is invariant is weaker and we state only that: under both judges and in both families, coverage, validity and equivalence are higher at the largest scale than at the smallest, and efficiency shows no consistent direction. The *shape* of capability scaling – saturation or transition – is a property of the judge, not of the planners, and we draw no conclusion about it in either direction. This is the same fact as the gap onset moving with judge stringency (Appendix T), now shown for

the curves themselves: which is the paper's thesis, not an exception to it. Four points per family remain enough to read a slope direction and not to place a transition, and the 0.5–8B range would need more sizes within each family, and a multimodal family alongside, before any shape claim could be made at all.

V SENSITIVITY: DISCRIMINATION, CROSS-JUDGE AGREEMENT, RUBRIC ROBUSTNESS

Discrimination is shared by every surface metric and tracks neither execution validity nor judge identity: EMBER separates 9 of 10 planner pairs, action F1 separates 10 and exact match 5 (Table 37). Cross-judge agreement is in Table 38.

Table 37: Pairs separated (of 10, from the five preregistered planners, n=1,641 each) under trajectory-cluster correction with Holm adjustment, and median paired $|d|$. Discrimination is shared by surface metrics and does not track execution validity.

Evaluator	Sig. pairs	Median $\|d\|$
EMBER (7B judge)	9/10	**0.72**
Action F1 (token-set)	10/10	0.52
Plan length error	10/10	0.49
Plan order (LCS)	9/10	0.61
Embedding cosine (MiniLM)	8/10	0.58
Exact match (EM)	5/10	0.11
Subgoal prefix	3/10	0.04

Table 38: Cross-judge agreement on the full evaluation set (n=8,205 pooled judgments per pair). Continuous scores correlate weakly and component agreement is moderate, yet planner rankings replicate under every judge.

Judge pair	Spearman ρ	Validity agr.	Coverage agr.
7B vs. 32B	0.466	0.75	0.61
7B vs. Mistral	0.316	0.87	0.82
32B vs. Mistral	0.397	0.70	0.50

RUBRIC ROBUSTNESS AND SELECTIVE COST

EMBER's discrimination is insensitive to the rubric: re-scoring under alternative weights, an additive form, or a reference-free form preserves 9–10 of ten pair directions (Table 40). A symbolic checker agrees with judge validity on 87–94 % of episodes for competent planners, 65–66 % for the two smallest (Table 41). A *selective hybrid* that calls the judge only when the symbolic checker is uncertain preserves all ten discrimination pairs (median $|d|$=0.71) while saving 21 % of judge calls on average (49 % for the weakest planner). Preregistered claim 3 had two halves. The noninferiority half passes at its 0.02 margin, and by more than noninferiority: the hybrid predicts execution at AUROC 0.831 against the full judge's 0.786 (Table 1); the paired difference is +0.045 (CI [+0.039, +0.052]), excluding zero. The call-reduction half asked for $\geq$ 50 % and reached 21 %, and the decision rule turns on it, so we execute the stated consequence – keep the full judge, report both halves, and make no cost-saving recommendation.

RUBRIC WORDING VARIANTS

The ablations above vary the rubric's *arithmetic* and leave its *wording* fixed, so they cannot separate the score's shape from the prompt that elicits it. We re-elicited the primary 7B judge under two rewordings of the same four capabilities – a terse checklist and a free-form instruction that names the capabilities without prescribing a form – on the primary pool at temperature 0, and recomputed the execution deficit under each. The reading rule – the deficit is robust to wording if both variants trail LCS with the interval's upper bound below zero – was fixed before the run, as amendment `amendments_rubric_robustness` in the preregistration file, and both variants meet it in

both conditions. Neither rewording rescues the judge: both also lose to the published rubric with intervals excluding zero, so the rubric we report is the strongest of the three rather than a weak one chosen to lose.

Table 39: Rubric wording variants, primary pool (n=6,564 per cell, 7B judge, temperature 0, zero parse failures in 26,256 calls; the 49 empty plans are scored 0 as in the main rubric). LCS is 0.890 in every row. − EMBER is against the published rubric *under the same reference condition*: 0.786 with the gold plan shown, the re-elicited 0.710 reference-free. Spearman ρ is against that same comparator's scores on the same episodes.

Wording	Condition	AUROC	− LCS	− EMBER	ρ
checklist	gold shown	0.607	−0.283 [−0.309, −0.254]	−0.178 [−0.213, −0.142]	0.30
	reference-free	0.571	−0.319 [−0.345, −0.292]	−0.139 [−0.162, −0.115]	0.44
free-form	gold shown	0.673	−0.217 [−0.240, −0.194]	−0.113 [−0.142, −0.083]	0.41
	reference-free	0.596	−0.293 [−0.321, −0.264]	−0.113 [−0.139, −0.085]	0.40

Two caveats on how to read the gap. The checklist is five boolean checks combined as a gated product and emits only 4 distinct values, so part of its loss is resolution rather than judgment. The free-form variant – two sentences of critique and then a 0–100 score – has no such excuse, emitting 19 distinct values in the gold-shown condition, and still trails LCS by 0.217. Both correlate only weakly with the published rubric (ρ 0.30–0.41), which is the cross-judge lesson of Appendix T restated within a single judge: absolute rubric levels are a property of the wording as much as of the plan.

Table 40: Rubric ablations: of the ten planner pairs, how many keep their direction and how many stay significant under each variant of the score.

Variant	directions kept	significant
default	10/10	9/10
weights 1.0/0.0	10/10	9/10
weights 0.5/0.5	10/10	9/10
weights 0.3/0.7	10/10	9/10
additive	9/10	9/10
no efficiency term	10/10	9/10
reference-free	9/10	9/10
selective hybrid	9/10	10/10

Table 41: Agreement between the judge's validity verdict and the symbolic checker, per planner, with the two disagreement directions.

Planner	n	agreement	judge lenient	judge strict
oracle-grounded skeleton	1,641	0.940	0.000	0.060
Qwen2.5-0.5B	1,641	0.655	0.270	0.075
Qwen2.5-1.5B	1,641	0.653	0.293	0.054
Qwen2.5-3B	1,641	0.871	0.082	0.048
Qwen2.5-7B	1,641	0.896	0.080	0.024

V.1 Discrimination against validity, on every evaluator

Sec. 5.7 states the dissociation in one line. Table 42 is the evidence: both coordinates on all seven evaluators, from the same rows for each. Pairs separated and median paired $|d|$ follow the procedure of Table 37; execution AUROC is over the primary pool, as in Sec. 5.1.

The two axes are *not* unrelated, and we report that rather than the cleaner claim. Across the seven, Spearman's ρ between pairs separated and execution AUROC is $+0.58$ (exact permutation $p = 0.18$), and between median $|d|$ and execution AUROC $+0.75$ ($p = 0.066$). Seven points support a direction and nothing finer.

One caution about populations. Discrimination in Table 37 is measured over five planners including the oracle-grounded skeleton, which every deficit claim excludes, while execution AUROC is over

the four primary planners. Correlating one against the other mixes populations, so the table also carries discrimination recomputed on the same four. It moves the answer: matched, the correlation with pairs separated falls to $+0.35$ ($p = 0.47$) while the one with median $|d|$ rises to $+0.89$ ($p = 0.012$), and the largest effect size passes from EMBER (7B judge) to Plan order (LCS). Seven evaluators over four planners will not settle a correlation; the ordering argument below does not depend on one.

What fails is the ordering, which is what a practitioner choosing an evaluator actually uses, and the clearest case survives either population. Plan length error separates *every* planner pair on both – 10 of 10 and 6 of 6 – and still reaches only 0.784, below every overlap metric here and below the judge it is meant to be a crude stand-in for. On the published population the largest effect size belongs to EMBER (7B judge), which ranks 4th of seven on execution. Sensitivity buys a lower bound on how coarse an evaluator is, not a claim about what it is sensitive *to*.

Table 42: Discrimination and execution validity for the same seven evaluators. Pairs separated and median $|d|$ are over the five preregistered planners (n=1,641 each) with trajectory clustering and Holm adjustment; execution AUROC is over the primary pool of four Qwen2.5 planners (n=6,564). Rows are ordered by execution AUROC. Regenerated by `analysis/discrimination_validity.py`.

Evaluator	Sig. pairs	Median $\|d\|$	Execution AUROC
Plan order (LCS)	9/10	0.61	0.890
Action F1	10/10	0.52	0.866
Embedding cosine	8/10	0.58	0.825
EMBER (7B judge)	9/10	0.72	0.786
Plan length error	10/10	0.49	0.784
Exact match (EM)	5/10	0.11	0.543
Subgoal prefix	3/10	0.04	0.525

W CALIBRATION, AND THE HUMAN-ANCHOR DESIGN

RANKING AND CALIBRATION ANSWER DIFFERENT QUESTIONS

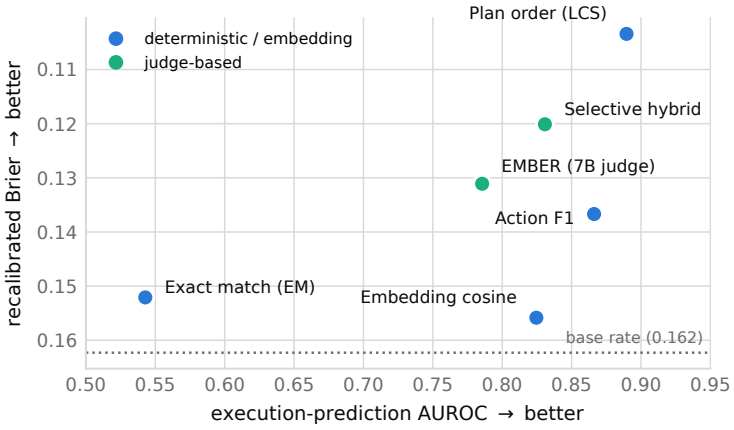

Figure 8: Table 1's six evaluators on one plane: execution-prediction AUROC against Brier after out-of-fold Platt recalibration, the latter axis inverted so better is up on both. The two axes disagree – EMBER ranks fifth of six on AUROC and third on recalibrated Brier, ahead of the two evaluators that outrank it on AUROC – which is the point of Appendix W. Built from the two artifacts Table 1 is built from.

Thresholding needs "how likely is *this* one," not a ranking. Read as probabilities (Table 1, right) none is usable as-is – which is arithmetic rather than a finding, since none of these scores was built to be a probability, and it is why the raw Brier column in that table should not be read as a result.

The informative comparison is the recalibrated one: after an out-of-fold Platt fit the ordering partly reshuffles, LCS stays best, but EMBER edges out action F1 despite ranking well below it (Table 43).

Calibration in full. Table 1 gives Brier scores; the expected calibration error (ten equal-width bins) separates miscalibration from sharpness. Platt recalibration is fitted out of fold on trajectory-grouped folds. No raw score beats predicting the base rate (0.204; Brier 0.162).

Table 43: Brier score and ECE, raw and after out-of-fold Platt recalibration (lower is better; n=6,564).

Evaluator	Brier raw	ECE raw	Brier Platt	ECE Platt
Plan order (LCS)	0.171	0.246	0.103	0.059
Selective hybrid	0.194	0.225	0.120	0.052
EMBER	0.245	0.303	0.131	0.050
Action F1	0.467	0.577	0.137	0.106
Exact match (EM)	0.187	0.187	0.152	0.017
Embedding cosine	0.620	0.687	0.156	0.082

Ranking and calibration are distinct axes: action F1 outranks EMBER on AUROC yet is far worse calibrated raw and slightly worse after recalibration.

What a human anchor could still decide

Our criteria are symbolic and behavioural, not preferential. A blinded human plan-quality anchor would answer what neither does: whether the judge's semantic signal, real but a worse execution predictor than overlap (Sec. 5.1), tracks what a person calls a good plan. We designed but did not run that study (655 stratified items, two raters each over five annotators, powered at 0.87 for a 0.10 Spearman advantage, preregistered); design and code are released, stratified by EMBER tercile – by the metric under test – so the stratification must enter as a covariate. A 120-item pilot packet, a prefix of the same design with its rater assignment kept, is released alongside it. Note the ceiling: a judge tracking human judgment while losing on execution would establish a *dissociation* between criteria, not rescue it as an execution predictor.

X Failure localization in detail

Preregistered claim 2 asked whether the rubric's components localize failure better than deterministic scalars. Macro-F1 is 0.286 for the rubric components, 0.508 for the five deterministic scalars together, 0.527 for both sets together, and 0.122 for the majority-class baseline; against the five together the components lose by -0.221 $[-0.248, -0.194]$. The registered endpoint is narrower than that: `macro_f1_gain_over_best_scalar_baseline_classifier`, the components against the best *single* deterministic scalar fitted alone. Fitted alone the scalars reach 0.238 (action F1), 0.288 (LCS), 0.208 (length error), 0.234 (predicted length) and 0.175 (gold length); the best is LCS, and the registered difference is -0.002 $[-0.024, +0.020]$, so the claim fails: the registered superiority of $+0.08$ is not achieved, and the interval excludes a gain of that size, but it does not show the components are worse than the one overlap scalar – only than the five combined. The comparator is the scalar observed best on these data, so the estimate is conditional on that choice; selecting the best of five can only favour the comparator, which makes it conservative for the rubric. An earlier version of this script reported only the five-feature comparison under the claim-2 label; the registered endpoint is the one the verdict now rests on, and both are kept.

Table 44: Per-class F1 under cluster-grouped five-fold cross-validation.

Class	n	components	deterministic
`dsl_vocabulary`	659	0.428	0.518
`goal_not_met`	1,778	0.606	0.691
`missing_referent`	789	0.216	0.448
`physical`	655	0.000	0.399
`precondition`	168	0.182	0.482

Table 45: Feature-set ablation, macro-F1. Two overlap features alone beat the four rubric components, so the result is not a plan-length artifact; adding the components to the deterministic set gains little.

Feature set	macro-F1
rubric components (4)	0.286
overlap only (F1, LCS)	0.314
overlap + length error	0.379
lengths only	0.336
all deterministic (5)	0.508
components + lengths	0.432

Y LIBERO: PROTOCOL GENERALITY WITHOUT AN EXECUTION ANCHOR

LIBERO enters the paper as a check that the protocol – converter, DSL, judge, metrics – runs unchanged on a second benchmark. It has no execution anchor, so nothing here bears on the deficit; it is reported for completeness. LIBERO also ships no skill-level gold plans, so our converter synthesizes a canonical minimal plan from each task's goal predicates. Scores here therefore measure conformance to that template, not agreement with a human-authored reference.

We ran the Qwen2.5-7B planner with the 7B judge on 650 paraphrase episodes over 130 tasks, with zero judge failures. Exact-match success rate is $0.0\,\%$: instance-level exact match never fires on LIBERO plans, so SR is uninformative there, while the judge reads 0.543 and action F1 0.563.

Table 46: LIBERO by suite. The judge separates suites that exact match cannot: pick-and-place compositions score high, predicate-style goals low.

Suite	n	EM	Action F1	EMBER
object	50	0.000	0.596	0.728
spatial	50	0.000	0.543	0.629
goal	50	0.000	0.588	0.367
LIBERO-10	50	0.000	0.528	0.481
LIBERO-90	450	0.000	0.563	0.540

Across the five paraphrases of each task the intraclass correlation of the judge's score is $\mathrm{ICC}(1) = 0.622$: most of its variance is attributable to task identity rather than wording. LIBERO's unseen pairs skew toward its easier object suite, so no compositional conclusion is drawn from it.

Z REPRODUCIBILITY

PREREGISTRATION AND REPRODUCIBILITY

Thresholds, gates and decision rules were frozen before outcomes were observed and are released verbatim, together with both amendments and their timestamps, the failed gates, and a decision log; the preregistration's commit field is left empty on purpose, the git history being the timestamp. Every number in this paper traces through a single facts sheet to a stored artifact. Three pieces of evidence back that, each a stored file rather than a sentence: `verify_artifacts.py` re-runs 54 GPU-free analysis scripts and diffs each artifact byte for byte (67 files identical, none differing, none skipped, in the stored report), the seven it cannot cover being listed in it with a reason; `independent_headline.py` re-derives the headline with a Mann–Whitney AUROC and its own trajectory bootstrap, sharing no code with the analysis package, and obtains EMBER 0.7855, LCS 0.8896, $\Delta = -0.1040$, CI $[-0.126, -0.083]$; and 193 unit tests pin the primitives against external references (AUROC and average precision under ties, cluster rather than row resampling, out-of-fold leakage) and pin each stored result the text quotes, so a regenerated artifact that changed a claim would fail a test rather than go unnoticed. Seven steps are outside that loop, each listed with a reason in the verifier script and in `REPRODUCE.en.md`: the hosted judge's recall probe and the judgments themselves (a server or an API key), the 32B capability curve (run separately), the embedding baseline (it needs the sentence-transformer weights), the AI2-THOR rollouts (they need the simulator), and the human-agreement analysis and packet builders (the study was designed but not run). The first perceptual-executor gate also predates a horizon fix; the later two gates reproduce.