

AURA: Intent-Directed Probing for Implicit-Need Surfacing in Situated LLM Agents

Yang Li, Jiaxiang Liu, Jiang Cai, Mingkun Xu*

Guangdong Institute of Intelligence Science and Technology, Zhuhai, China
{liyang, liujiaxiang, caijiang, xumingkun}@gdiist.cn

Abstract

A situated query like “where is Lin Wei?” often encodes more than its literal content: the user may also want to know whether Lin Wei is free, in a good mood, or worth interrupting now. Standard tool-use agents answer the literal question and stop. AURA inserts an inference step between scene perception and tool use that produces an *IntentFrame*: a structured estimate of the implicit need with a scalar *gap* score that controls per-query probe budget and tool selection. On a 100-query four-scene implicit-intent benchmark, AURA improves implicit-need coverage over ReAct-style probing ($\Delta=+0.07$, $p<10^{-6}$); three of four scenes are individually significant, the gain reproduces on a second backbone, and a prompt ablation attributes the lift to gap calibration rather than answer memorisation. A component ablation localises the gain further: the gap-derived budget ceiling alone outperforms the full architecture ($p=0.02$), and bin-wise calibration analysis ($r=-0.33$) identifies the gap score as a probe *trigger* rather than a calibrated value estimator. On factual lookup the controller trades raw accuracy for 82% fewer probes and zero forbidden-tool violations on a privacy-sensitive slice; scope conditions are detailed in Limitations. Code, simulator, and benchmark are released at <https://github.com/innovation64/AURA>, with the implicit-intent benchmark also on the HuggingFace Hub (innovation64/aura-implicit-intent).

1 Introduction

LLM-based agents (Xi et al., 2025; Wang et al., 2024) have been deployed in social simulations (Park et al., 2023) and software teams (Hong et al., 2024), but they often answer the user’s *literal* question about an environment while missing the *implicit* information need behind it. Asking

“where is Lin Wei?” may be a request for location, but it may also mean “is she free to chat?”. Agents then decide with incomplete context, invent details instead of grounding in current state, and rarely translate environment state into what the user needs next.

Prior approaches answer this only partially. **ReAct** (Yao et al., 2023) interleaves reasoning and tool calls during answer generation, but tools fire only when the surface query explicitly demands them — the loop has no step that asks “what does the user actually want to know?”. **Plan-and-Solve** (Wang et al., 2023) pre-plans tool calls from the literal query, with no mechanism to bridge to an implicit need that the literal query does not name. **Generative Agents** (Park et al., 2023) inject all environment state as passive context but expose no control over which private state to surface for a given query. Each treats the surface query as if it fully specified what the user wants. AURA adds the missing step: implicit-need inference as an *independent control variable*, a pre-tool routing decision that determines which private-state probes to issue and how many.

By *situated* we mean structured environments where private state sits behind tool-mediated access: public context is observed passively, hidden state must be probed. This is narrower than the “situated agent” umbrella (embodied navigation, open-world exploration, sensor fusion), which we do not address. We study this problem as implicit-intent inference over structured environment state. AURA introduces the **Environment Agent**: a pipeline (Perceive $\rightarrow$ Scene $\rightarrow$ Memory $\rightarrow$ Reason) plus an LLM-generated *IntentFrame* that estimates the gap between a user’s literal query and plausible implicit need, then uses that estimate to direct per-query probing, tool selection, and optional heads-up alerts.

Contributions. (i) We introduce the *IntentFrame* as a *pre-tool control variable*:

* Corresponding author.

before any tool fires, the agent infers the user’s implicit need and emits a scalar gap score that determines the probe budget and shortlists candidate tools. (ii) On a 100-query four-scene implicit-intent benchmark, gap-routed probing improves implicit-need coverage over ReAct-style NoIntent ($\Delta=+0.07$, $p<10^{-6}$; three of four scenes significant), reproducing on a 25-query pilot, a second backbone, and under disjoint-example ablation. (iii) We characterise the regime boundary: an access–cost Pareto point on factual grounding, not a universal accuracy win (§5.1; Limitations). (iv) We release the AURATOWN simulator, all 100 implicit-intent queries with subcategory labels (inter-annotator agreement $\kappa=0.61$, measured on the 25-query pilot subset), and per-seed run records.

2 Related Work

Agent architectures and tool use. ReAct, Reflexion, Toolformer, AgentBench, ToolLLM, T-Eval (Yao et al., 2023; Shinn et al., 2023; Schick et al., 2023; Liu et al., 2024; Qin et al., 2024; Chen et al., 2024b) treat tool use as reactive: tools fire *during* reasoning. AURA’s Explore runs *before* reasoning with a bounded budget routed by the inferred literal-vs.-implicit gap. Plan-and-Solve (Wang et al., 2023) plans tool calls from the surface query without that gap routing.

Multi-agent simulation and memory. Generative Agents (Park et al., 2023) inject environmental observations into LLM prompts; AURA adds (i) per-query access *control* by an LLM-produced IntentFrame, vs. passive top- k retrieval; (ii) a public/private state split that Park’s all-public state lacks. SOTOPIA (Zhou et al., 2024) provides the 7-dimension framework we adopt; broader multi-agent benchmarks (Zhou et al., 2024; Zhu et al., 2025; Xu et al., 2024; Vezhnevets et al., 2023; Chen et al., 2024a; Li et al., 2023; Hong et al., 2024) treat the environment as serving agents on request. Memory frameworks (Packer et al., 2023; Zhong et al., 2024; Lewis et al., 2020; Gao et al., 2023; Li et al., 2026) target long-term storage; AURA’s memory is populated by proactive probing, not passive accumulation. Proactive context agents (Yang et al., 2025b,a; Pasternak et al., 2025; Tang et al., 2026) (Appendix A7) and interactivity-native systems (Thinking Machines Lab, 2026) target *when to assist*; AURA is orthogonal, targeting *what private-state context to surface* for an

already-received query.

Pragmatic and conditional QA. Text-only QA meets the same motivation: users omit context they assume the model shares. CON-DAMBIGQA (Li et al., 2025) resolves this by condition-aware reasoning over text; in AURA the implicit need lives in another agent’s hidden private state, and the resolution is a budgeted probe over an environment registry rather than a textual condition rewrite.

Intent, plan recognition, and information value. The IntentFrame revisits three older ideas. *Plan recognition* (Kautz and Allen, 1986; Charniak and Goldman, 1993; Carberry, 2001; Baker et al., 2009) infers an agent’s goal from observed actions; our inferrer infers the *user’s* unstated need from a query and a scene, and acts on it by probing. In pragmatics, the question-under-discussion framework (Roberts, 2012) and Gricean implicature (Grice, 1975) formalise how a literal utterance underdetermines what is being asked; rational-speech-act models (Frank and Goodman, 2012) make that inference probabilistic; AURA collapses it to a scalar control input. POMDP dialogue managers (Williams and Young, 2007; Young et al., 2013) choose information-gathering actions under belief uncertainty, and value-of-information analysis (Howard, 1966), applied to clarification questions by Rao and Daumé III (2018), prices each such action. Our gap-to-budget map is a hand-tuned stand-in for that computation, and the inverted calibration we report (§5.2) is the symptom a learned value-of-information estimate would address.

Belief-state evaluations. Theory of mind in LLMs is contested: Kosinski (2024) argues spontaneous emergence; Sap et al. (2022); Ullman (2023) show failure on trivially-altered Sally–Anne tasks; Sclar et al. (2023) build an explicit symbolic tracker. AURA is *not* a ToM benchmark: of our five implicit-intent subcategories, only *second_order* probes belief tracking (stale-belief templates adapted from Ullman (2023)); the other four (*availability*, *mood*, *appropriateness*, *latent_goal*) target surfacing of non-belief private state through a budgeted controller. We position AURA as evidence about *when intent-directed probing helps surface hidden state*; belief queries appear in our 5-subcategory taxonomy as one substrate (*second_order*). Transfer to ex-

ternal belief benchmarks does not hold automatically: Appendix A21 reports a FANToM transfer null.

3 The AURA Framework

AURA factors a situated agent into two phases (Fig. 1): *deterministic context assembly* (Sense $\rightarrow$ Scene $\rightarrow$ Memory) followed by *LLM-controlled reasoning* (IntentInferer $\rightarrow$ Explore $\rightarrow$ Reason $\rightarrow$ Act $\rightarrow$ Interact). The deterministic phase establishes a passive context preview using the same passive-perception pattern as Generative Agents (Park et al., 2023) and is not the contribution. The contribution is the INTENTINFERRER, which produces an IntentFrame whose scalar gap field controls per-query probe budget and tool selection in Explore (Section 3.3); Reason, Act, and Interact then plan and emit the response, with an optional heads-up alert when the inferer’s gap value crosses a threshold. The remainder of this section formalises the problem (Section 3.1), describes the bounded-probing loop (Section 3.2), and details the IntentFrame stage (Section 3.3).

3.1 Problem Formulation and Architecture

We consider a situated multi-agent system where N agents operate in a shared environment $\mathcal{E}$ over discrete time steps. A human user h may interact with any agent via natural-language queries q . We seek to maximise *environmental grounding* of agent actions and user-facing responses:

$$\max_{\pi_E} \mathbb{E} \left[\sum_t G(\alpha_i^t, \mathcal{E}^t) + \sum_q R(r_q, \mathcal{E}_q^t) \right],$$

where G measures action-state consistency, R measures response factual accuracy, and π_E is the Environment Agent’s probing policy.

The Explore stage’s useful range is bounded by the residual uncertainty after passive perception: when Sense/Scene/Memory cover the facts the query needs, extra probes add little; when the query depends on a remote agent’s private state, one targeted probe can change the answer.

3.2 Bounded Proactive Probing

The **Explore** stage implements bounded environment probing. Unlike ReAct (Yao et al., 2023) where tool calls are interleaved *during* reasoning, our probing occurs *before* answer generation, so the final response is conditioned on a compact probe trace. The probe loop iterates up to

a budget B steps; at each step an LLM planner ϕ examines the current context and the tool registry $\mathcal{T}=\{t_1, \dots, t_K\}$ and either issues a tool call (whose result is appended to a probe trace and used to update the context) or signals stop; the trace is summarised at the end and consumed by Reason. Pseudocode in Algorithm 1 (Appendix A1).

Design. The loop is goal-directed (LLM-driven tool selection given current context, not exhaustively), bounded (step budget B caps API cost), composable (each tool result updates context for the next decision, enabling multi-hop gathering), and separable (probing runs independently from reasoning, so results can be cached or skipped). The AURATOWN instantiation registers eight base environment tools (Table 10) for the factual-grounding benchmark, and a separate five-tool scripted registry of agent-state probes for the implicit-intent benchmark (Section 5.2); both register under a pattern-based allow/deny policy.

3.3 Intent Inference: Modeling the User’s Implicit Need

Proactive probing decides *what to ask the environment*; intent inference estimates *what useful answer the user may need*. AURA factors the second decision out as an explicit LLM-mediated stage that sits between Scene/Memory and the Explore budget selector.

Given the user’s surface query q , the pre-view scene $\mathcal{S}$, and recent memories M , the IntentInferer produces an IntentFrame:

$$\text{IntentFrame} = (\ell, I, g \in [0, 1], P \subseteq \mathcal{T}, a \in \{0, 1\}, c \in [0, 1], r)$$

where ℓ is the literal-need restatement, I is the list of plausible implicit needs, g is the *gap* between literal and implicit (0 if the literal answer suffices, 1 if the user’s real need is orthogonal), P is a set of recommended probe tools drawn from the registry $\mathcal{T}$, a is an alert flag, c is the inferer’s self-reported confidence, and r a rationale.

The gap g is the control input: downstream stages use a deterministic map $B(g) = 0, 1, 2, 3, 5$ for $g \in [0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1]$ respectively, truncated by the global `explore_max_steps` budget. Importantly, $B(g)$ is a *ceiling*, not a target: the downstream Explore loop receives both the probe budget and

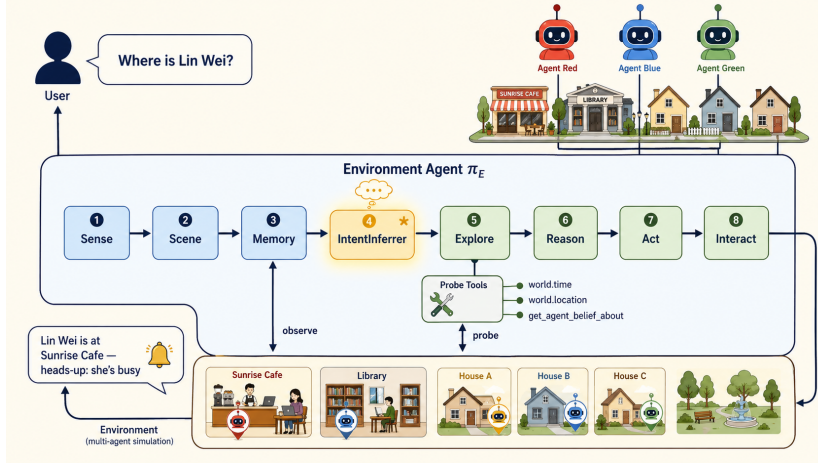


Figure 1: AURA pipeline. Sense receives the user query; Scene and Memory assemble a deterministic context preview. The **IntentInferrer** (★) outputs an IntentFrame whose scalar gap field routes probe budget and tool selection in Explore (Section 3.3). Reason plans over the enriched context; Act and Interact emit the response.

the recommended_probes hint, and we observe in Section A2.1 that the LLM typically stops short of the ceiling when one well-targeted probe has already returned actionable information. The two signals are separable, and the component ablation in Section 5.2 shows they are not equally load-bearing: the gap-derived ceiling carries the gain, while the recommended-probes shortlist is dispensable, and on our registry slightly harmful, once budget routing is in place.

Worked example. For the query “where is Lin Wei?” at 14:30 with Lin Wei present in the cafe scene, the inferer outputs ℓ =“locate Lin Wei”, I ={“is she free to chat?”, “is she taking a break?”}, $g=0.6$, P ={get_agent_private_state, get_agent_plan}, $a=1$, $c=0.7$. The $g=0.6$ maps to a budget ceiling of $B=3$, but Explore stops after two probes when get_agent_private_state(“Lin Wei”) returns availability=busy, which suffices to answer the implicit need.

Two backends and control boundary. A heuristic backend (deterministic surface-cue matching, for tests and offline execution) and an LLM backend (structured-JSON output, calibrated with a four-tier gap rubric and clean benchmark-disjoint few-shot exemplars; recommended-probes whitelisted against the live tool registry) are both provided. The architectural pipeline up to this point (Sense–Scene–Memory) is deterministic; the IntentFrame is the first stage at which the LLM affects control flow (probe

budget, tool priority, alert flag). Section A2.1 quantifies the resulting per-query adaptation, and Appendix A12 shows that the examples act as gap calibration rather than answer templates. The heuristic backend is provided for offline tests and air-gapped deployments; all reported results use the LLM backend.

Memory architecture. AURA’s memory scores each item m against query q at time t by a weighted combination of recency, importance, and lexical similarity (Eq. 1, Appendix A24, with weights 0.3:0.4:0.3 and decay 0.01). Memory types follow Tulving (1972) (observation, conversation, reflection, plan), with reflections triggered every $\theta=10$ observations. Keyword-based similarity is a deliberate reproducibility trade-off; embedding retrieval is a natural extension. A three-stage enrichment protocol routes each query through context gathering, single-step probe verification, and enriched generation (Appendix A24).

Relationship to long-context approaches. A natural alternative is to pack the full environment state into the prompt and rely on the LLM’s long-context attention. AURA’s selective probing pays one extra IntentInferrer call but issues 0–3 targeted probes whose count tracks the inferred gap, rather than the world size. We do not run a head-to-head against a stuff-everything baseline (such a baseline would also need to decide which slice of memory to include); the Static-Context baseline (§5.1) (a packed scene snapshot) is the closest in-suite analogue and reaches FA 0.450 vs. AURA

Full’s 0.640. A retrieval-packing baseline that selects a larger private-state slice under a *matched token budget* is the comparison that would directly test selective probing against context packing; we name it as open future work rather than approximate it here. Appendix A7 sketches the token-cost and prompt-content differences.

4 AURATown: A Multi-Agent Social Simulation

AURATOWN is a 60×60 grid-based social simulation with 5 named agents and 20 named locations over a 6:00–23:00 day (full setup, map, and a tick-18:00 mechanism snapshot in Appendix A4). Two decisions depart from Park et al. (2023)’s 25-agent Smallville. (i) *Scale*. 5 agents yields tractable per-query ground truth; we test a per-agent mechanism, not multi-agent emergence. (ii) *Public/private state split*. location and action are visible in the scene snapshot, but availability, emotional_state, unspoken_goal, and beliefs_about_others are hidden and only retrievable via probe tools. Each agent’s private state updates deterministically each tick via a seven-rule priority table (Appendix A3): e.g., an agent at a loaded workplace becomes busy and tired-focused; one at an empty workplace becomes available and lonely. Beliefs about other agents refresh only on co-location, producing the staleness that *second_order* queries probe. The released codebase additionally includes a chunk-based procedural world used by the demo deployment (demo/town/chunks.py); experiments here use only the fixed 60×60 / 5-agent / 20-location subset.

5 Experiments

We evaluate AURA along three axes: environment access as a boundary condition (§5.1), intent-directed probing (§5.2), and adaptive probe allocation. Primary experiments use gpt-4o-mini as both agent backbone and LLM-as-judge (full hyperparameters in Appendix A6; cross-backbone robustness on claude-haiku-4-5, qwen-plus, gemini-2.5-flash in Appendix A13). The same-family judge-policy setup limits independence; we address this with a strict precision rescore in Appendix A9. Baselines, metrics, and additional diagnostic checks (routine grounding, component ablation, SOTOPIA, human eval, budget sweep) are in Appendices A8–A17.

Regime characterisation. A scope condition governs everything that follows and should be read as a *precondition* of the method, not a footnote: the IntentFrame controller assumes an environment whose task-relevant hidden state is *tool-gated*: absent from the prompt context, but retrievable through a registered probe. AURA Intent leads the implicit-need regime (§5.2), which satisfies this condition by construction; on factual grounding (§5.1), gap-routed probing is an access-cost Pareto point rather than an accuracy winner. Where the condition fails the mechanism adds nothing: no lift when private state is already in context (LoCoMo) or structurally inaccessible (FANToM, GAIA; Appendices A21–A23).

Routine grounding: metric saturates. Under 100 simulation steps per condition $\times$ 3 seeds, all five architectures (Vanilla, Static Context, ReAct, AURA No-Probe, AURA Full) fall within 0.024 absolute GA spread, every paired *t*-test vs. vanilla gives $p > 0.5$. The metric is saturated: memory-utilisation ≈ 1.0 for every method, most actions are trivially grounded (“sleeping at home at 6 AM”). This is evidence about the metric, not the mechanism (Appendix A16).

5.1 Factual Grounding: Environment Access as a Boundary Condition

This subsection tests whether gap-routed probing is merely a generic accuracy booster (it is not). We collect 50 user queries about the environment spanning 5 categories (spatial, social, temporal, memory, planning). Responses are scored against ground-truth environment state by a gpt-4o-mini LLM-as-judge (same family as the agent backbone — see Appendix A9 for a strict precision rescore that reduces sensitivity to judge softness). We repeat across three random seeds {42, 123, 456} and report mean $\pm$ std, together with paired *t*-tests against two reference conditions: the vanilla LLM (to measure total architectural effect) and AURA (No Probe) (to isolate the marginal contribution of proactive probing on top of the Perceive/Scene/Memory pipeline).

Scope evidence: factual lookup is an accuracy-access tradeoff regime. The 50 factual queries ask for environment facts — “Where is X?”, “What time is it?” — so saturated access is a strong control. Fixed-Probe invokes all eight environment tools on every query and reaches 0.766 FA, statistically above GapRouted (0.696;

Table 1: Factual Accuracy (FA) on environment-grounded queries ($N=50$, 3 seeds). **Bold** marks within-column best by criterion: Fixed-Probe / Plan-and-Solve on raw FA; GapRouted infers a per-query gap and executes only the recommended probes; AURA GapRouted ($\star$, Pareto frontier in Fig. 2) on access cost (Probes); ReAct on contradicted-claim rate. Contrad. % is diagnostic; AURA makes no hallucination claim in this regime. Strict precision rescore in Appendix A9. † ReAct calls tools via OpenAI function-calling, bypassing the AURATown probe registry; its probe count is not directly comparable.

Method	FA	Probes	Contrad. %	Lat. (s)	$p_{vs. \text{Van}}$	$p_{vs. \text{Full}}$
Vanilla LLM	0.070 ± 0.010	0.00	93.3	2.3	—	$< 10^{-4}$
Static Context	0.450 ± 0.043	0.00	70.0	1.8	0.032	$< 10^{-3}$
ReAct Agent	0.550 ± 0.037	— †	51.3	6.0	0.100	0.033
Reflexion	0.174 ± 0.049	2.15	55.9	20.6	—	$< 10^{-4}$
Plan-and-Solve	0.764 ± 0.020	4.64	58.7	5.2	—	0.004
Fixed-Probe	0.766 ± 0.021	8.00	56.0	3.2	$< 10^{-4}$	0.006
AURA (No Probe)	0.603 ± 0.028	0.00	75.3	2.2	0.0005	0.060
AURA (Full, $B=2$)	0.640 ± 0.021	2.00	66.7	3.9	0.0010	—
AURA (GapRouted) $\star$	0.696 ± 0.026	1.40	75.3	4.3	$< 10^{-4}$	0.098

paired $\Delta=+0.070$, $p=0.031$). Plan-and-Solve is similar in raw FA (0.764; paired vs. GapRouted $\Delta=+0.068$, $p=0.056$). The gap-routed controller therefore is *not* the raw-accuracy winner on factual lookup. Its contribution is an access Pareto point (Figure 2): 1.40 probes/query vs. Fixed-Probe’s 8.00 (**82%** fewer; $p=4.9 \times 10^{-52}$), and disclosure 0.92 vs. 5.00 ($p=6.7 \times 10^{-38}$). *Disclosure* is a per-query sum, over the tools actually invoked, of a fixed sensitivity weight: 0 for public-observable tools (`world.time`, `world.location`, `world.nearby_agents`), 1 for moderate ones (`world.agents_summary`, `world.events_recent`, `agent.plan`), and 2 for `memory.recent`; unlike raw probe count, it weights *what kind* of state each call exposes. On a 30-query privacy-sensitive factual slice (Appendix A10), GapRouted ties Plan-and-Solve and ReAct in FA ($p=0.86$ and 0.65) while reducing forbidden-tool violations to **0%** (Plan-and-Solve 78.9%, ReAct 25.6%, Fixed-Probe 100%). Per-query wall-clock latency tells a more cautious story: GapRouted pays the IntentInferer’s extra LLM round trip and is slower at the median than Fixed-Probe (4.08 vs. 2.37 s; Appendix A11), so the cost-of-selectivity claim holds on probe count and disclosure rather than on latency. The mechanism’s primary accuracy contribution remains the implicit-need setting in Section 5.2; the factual-grounding regime bounds the cost of selective access.

Reflexion failure mode. Reflexion (Table 1) collapses to 0.174 FA because the reflection step withdraws probe-supported claims (Appendix A15); the Full-vs-No-Probe contrast is

near-null at $\text{gap} \approx 0$ ($p=0.060$).

Privacy as a design benefit. Gap routing’s zero forbidden-tool violation rate is not a post-hoc observation: low-gap factual queries receive $B=0$ budget and skip the probe registry entirely, so they structurally cannot invoke a forbidden tool. This makes the access-cost Pareto a privacy Pareto as well.

Strict-precision robustness. A strict-precision rescore (Appendix A9) confirms the access-cost picture: per-category gains concentrate on temporal and memory; the overall architectural margin above ReAct narrows to non-significance ($\Delta=+0.090$, $p=0.058$). The IntentFrame’s primary accuracy contribution is the implicit-need regime (§5.2).

5.2 Implicit-Need Surfacing via Intent Inference

This subsection tests whether intent inference itself surfaces the user’s implicit information need on queries specifically designed to hide that need behind a literal surface form.

Query set. We construct 25 primary implicit-intent queries stratified across five subcategories that vary the relationship between surface form and implicit need: *availability* (e.g., “where is Lin Wei?” $\rightarrow$ “is she free?”), *mood*, *appropriateness*, *latent_goal*, and *second_order* (representative queries for the remaining four and the second-order belief-vs-truth judging rule in Appendix A14). The scene exposes only public state (location, action); private-state fields (availability,

AURA GapRouted on the access-cost Pareto frontier

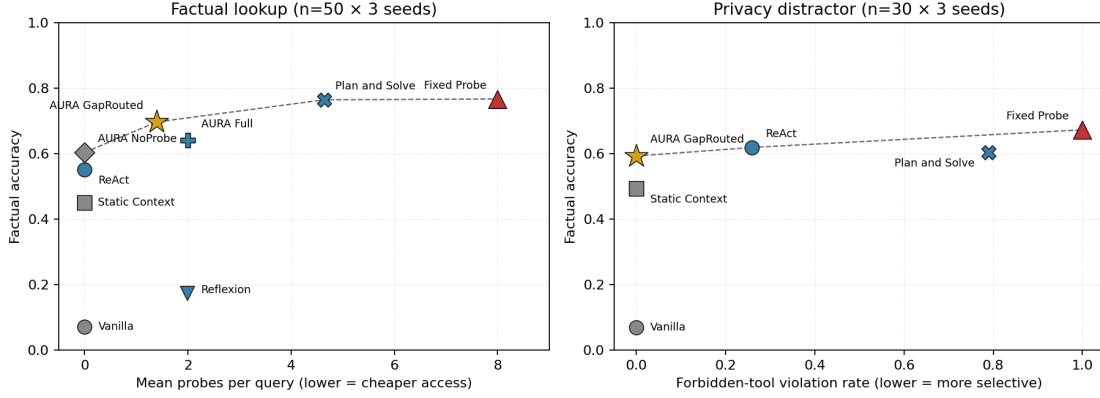


Figure 2: Factual-grounding access Pareto. Left: factual lookup ($N=50 \times 3$ seeds) — factual accuracy vs. mean probes per query. AURA GapRouted ($\star$) sits on the frontier with 1.40 probes; Fixed-Probe wins raw FA at $8\times$ the access cost. Right: privacy-sensitive distractor slice ($N=30 \times 3$ seeds) — factual accuracy vs. forbidden-tool violation rate. AURA GapRouted holds the Pareto vertex with 0% violations while approaching Fixed-Probe (100%) and Plan-and-Solve (78.9%) in FA.

emotional state, unspoken goal, beliefs about others) are only retrievable via probe tools.

Conditions. Four answer strategies share the same backbone (gpt-4o-mini, temperature 0.1) and a 5-tool agent-state registry (Appendix A5; disjoint from the factual-grounding registry): **Literal** (scene only, no tools); **NoIntent** (ReAct-style, up to $B=3$ tool calls); **Plan-and-Solve** (Wang et al., 2023) (plan/execute/synthesise, budget 3); **Intent** (IntentInferer $\rightarrow$ gap-derived probe ceiling 0–3 $\rightarrow$ directed tool loop, with heads-up prefix on high-gap queries). Each response is scored by a gpt-4o-mini judge on two dimensions in $[0, 1]$: `literal_score` and `implicit_score` (does it surface the implicit need?). We run 3 seeds (42, 123, 456) over the 25 queries and report query-level paired t -tests as primary, with seed-level paired tests and query_id cluster bootstraps as secondary checks (Appendix A14).

Overall. On the 100-query four-scene benchmark (Table 3), AURA-Intent reaches **0.804** implicit-need coverage versus 0.733 for NoIntent (ReAct-style) and 0.223 for Literal: paired $\Delta=+0.071$, $p=1.0 \times 10^{-6}$. Three of four scenes are individually significant; night scene D ties because public state already telegraphs availability. A 25-query pilot (Scene A of the four-scene set, Table 2) confirms the direction at higher absolute gain ($\Delta=+0.16$, $p=0.006$) and additionally tests Plan-and-Solve ($\Delta=+0.28$, $p=8.3 \times$

Table 2: Implicit-intent comparison ($N=25$, 3 seeds; scores in $[0, 1]$). **Bold** marks the best deployable system per column; Fixed-private and Oracle-intent are diagnostic upper bounds (require unconditional private-state access or gold-tool knowledge) and not bolded. Tool-using rows share the same 5-tool registry. Prompt-example and component ablations in Appendices A12, A17.

Condition	Lit.	Implicit	Probes	Lat.
<i>Deployable systems</i>				
Literal	0.659	0.216	0.00	1.3
NoIntent	0.947	0.640	1.04	2.2
Plan-and-Solve	0.811	0.520	1.37	11.1
AURA Intent	0.957	0.803	1.48	13.8
<i>Diagnostic upper bounds (not deployable)</i>				
Fixed-private	0.987	0.851	1.40	2.4
Oracle-intent	0.997	0.861	2.32	2.7

Table 3: 100-query four-scene implicit-intent benchmark (3 seeds, gpt-4o-mini). Implicit-need coverage scores in $[0, 1]$. **Bold** marks best per column. p values are paired tests on query-seed cells.

	Overall	A: cafe	B: library	C: garden	D: night
Literal	0.223	0.200	0.264	0.237	0.192
NoIntent	0.733	0.709	0.688	0.731	0.803
AURA Intent	0.804	0.800	0.779	0.811	0.827
Δ (Intent–NoIntent)	+0.071	+0.091	+0.091	+0.080	+0.024
p	1.0×10^{-6}	0.004	1.4×10^{-4}	0.015	0.349

10^{-5}) and two diagnostic upper bounds (fixed-private 0.851, oracle-intent 0.861; underpowered to separate from AURA at $N=25$).

Per-subcategory structure. The subcategory breakdown shows where intent inference helps.

Availability gains most ($\Delta=+0.29$, $p=2.7 \times 10^{-11}$): “where is X?” fully masks the implicit need, so the gap score routes a probe that would not otherwise fire. *Appropriateness* (+0.11) and *mood* (+0.07) follow at smaller magnitudes (both $p<2 \times 10^{-3}$). *Second_order* ties (-0.02 , NS): “does X think Y...” already cues belief-state probing. *Latent_goal* shows a deficit (-0.09 , $p=6.2 \times 10^{-4}$). The gap score adds value precisely when the surface form is maximally decoupled from the implicit need. A worked failure case for *latent_goal*, where the shortlist spends a full ceiling on one agent and misses the intent, is in Appendix A14.

Few-shot calibration is a precondition of the method. A three-way prompt ablation (Appendix A12) shows the gain does not come from memorising examples, but does depend on their presence. With clean benchmark-disjoint exemplars the Intent-vs-NoIntent contrast is $\Delta=0.163$ ($p=0.006$); removing them entirely reduces it to $\Delta=0.035$ ($p=0.44$). Replacing them with leaked examples changes little, which rules out answer-template memorisation. What the exemplars supply is calibration: without them the mean inferred gap falls and half as many queries clear the probe threshold. Any prompted control variable needs some such signal, and ours is a small, disclosed, non-answer-specific one; the method should not be expected to work without it.

Cross-backbone. The gain reproduces on claude-haiku-4.5 ($\Delta=+0.086$, $p=3.7 \times 10^{-3}$) and qwen-plus ($\Delta=+0.25$, CI [+0.14, +0.37]); per-backbone detail and a Gemini JSON-parse failure in Appendix A13.

Backend ablation: gap inference is load-bearing. Replacing the LLMIntentInferer with a deterministic HeuristicIntentInferer (rule-based gap, identical plumbing) drops implicit score from **0.803** to 0.368 (Appendix A19), most on lexically-decoupled subcategories: the lift attaches to LLM-mediated gap inference, not the scaffolding.

Human evaluation: output quality vs. NoIntent. Eight raters scored 50 paired (AURA vs. Vanilla) scenarios on four dimensions; AURA rates higher on all four, with 74% of cells reaching $\geq 6/8$ consensus for AURA and none for Vanilla (Appendix A17.3). That establishes only a preference over the weakest baseline, so we ran a sec-

Table 4: Component ablation, 100-query benchmark ($n=300$ query-seed cells). *Budget* = gap-derived ceiling; *Hints* = LLM-recommended tools P ; without hints the loop uses Fixed-Probe’s deterministic tool order. Paired t -tests vs. NoIntent.

Condition	Implicit	Probes	Δ vs. NoIntent	p
NoIntent (neither)	0.733	1.11	—	—
Budget only	0.827	1.20	+0.095	6.9×10^{-11}
Hints only	0.813	1.74	+0.080	5.6×10^{-8}
Both (AURA)	0.804	1.75	+0.071	6.4×10^{-7}

ond arm against NoIntent on the implicit-intent items, with the scene’s ground-truth state shown so that accuracy was checkable (Appendix A18). It **does not** establish a human-perceptible advantage: three dimensions favour AURA without reaching significance, and AURA rates significantly lower on factual accuracy ($p=0.008$), where no rater preferred it. Two things bear on reading that: the arm rated the full IntentFrame variant, the one budget-only routing dominates, and AURA’s answers run nearly twice as long, which raters penalised. Coverage of a stated hidden need and holistic response quality are distinct properties, and AURA claims only the first.

What the coverage gain does not buy. Two results should be read together. AURA surfaces more of the user’s implicit need, and it also carries a higher contradicted-claim rate than ReAct on factual lookup (Table 1) and rates lower on human-judged factual accuracy. We take the straightforward reading rather than the flattering one: the system says more, and a larger share of what it says is unsupported by what its probes actually returned. The coverage metric scores whether the stated hidden need was addressed; it does not penalise the elaboration surrounding that answer, and our heads-up prefix actively encourages such elaboration. Coverage and grounding are therefore not two views of one improvement, and a deployment that cares about the second should add an output constraint we do not impose: assert only what a probe returned.

Adaptive budget: ceiling, not target. Mean probes per query range 0.80–2.20 under a fixed ceiling of 3, and gap correlates only $r=0.19$ with realised probe count: the gap routes a ceiling, not a target, so the agent spends less than a fixed-budget system on most queries (Appendix A2.1).

Component ablation: the budget ceiling, not the probe hints, carries the gain. The

IntentFrame feeds two signals downstream: the gap-derived budget ceiling $B(g)$ and the recommended-probe shortlist P . Isolating them (Table 4) shows the ceiling alone, paired with Fixed-Probe’s deterministic tool order, is the strongest configuration; the full system is significantly worse, and consistently so across four of five subcategories. This is not an artifact of unstable inference: the two conditions infer near-identical gap values, yet agree on only a fifth of their realised tool sequences. The shortlist sometimes substitutes a plausible but weaker tool (`get_nearby_agents` where `get_agent_private_state` answers the need) for one the fixed order reaches reliably. The load-bearing component is therefore gap-triggered budgeting, which we recommend over a fixed probe order (Appendix A5) as AURA’s deployment configuration: it is the most accurate and cheapest arm we measure. We report the full architecture as primary only for continuity with the rest of the suite, which has no budget-only arm.

Gap calibration: a trigger, not an estimator.

If the gap score estimated the value of probing, higher-gap queries would benefit more from it. They benefit less. Across the 300 paired cells, per-query benefit correlates negatively with the inferred gap ($r=-0.33$), falling monotonically as gap rises (bin-wise values in Appendix A2.1). The cause is the subcategory structure already visible above: the inferrer assigns gap almost categorically by query type, and the types it scores highest are the ones a ReAct baseline already probes correctly. What the gap score does reliably is decide whether to cross the probe threshold at all, which the backend ablation shows requires LLM-mediated inference. What it does not do is estimate how much probing will help.

6 Conclusion

AURA inserts an inference step between scene perception and tool use: an IntentFrame whose gap score routes private-state probes before answering, improving implicit-need coverage over ReAct-style probing. Three ablations localise that gain. LLM-mediated gap inference is needed to trigger probing at all; the budget ceiling alone outperforms the full architecture; and gap correlates negatively with the benefit of probing. A small inferred signal deciding *whether* to spend probe budget carries the result, while the frame’s

finer-grained fields do not yet earn their place. We therefore recommend the leaner controller, and take learning its mapping from logs as the next step.

Limitations

Regime scope. The controller targets situated queries with tool-mediated hidden state. On factual grounding (§5.1) it acts as an access–cost Pareto point rather than an accuracy winner; cross-domain checks on FANToM, LoCoMo, and GAIA show no measurable lift when private state is already in-context or structurally inaccessible (Appendices A21–A23).

Benchmark and calibration. The 100-query four-scene benchmark is author-written; on the 25-query pilot subset, inter-annotator agreement over the 5-subcategory partition is substantial ($\kappa=0.61$, two independent annotators; the remaining 75 queries carry single-annotator labels, details in Appendix A14). The few-shot calibration examples are load-bearing for gap estimation: removing them reduces the gain to non-significance, though replacing them with benchmark-disjoint examples preserves it (Appendix A12). The gap score itself is not a calibrated estimator of probe value: bin-wise gap-vs-benefit is inverted ($r=-0.33$, §5.2), and the full architecture underperforms its budget-only ablation ($p=0.021$), so claims about the IntentFrame should be read as claims about gap-triggered budget routing, not about the frame’s finer-grained fields. Human evaluation uses $N=8$ raters (Krippendorff’s $\alpha=0.43$); directional agreement is strong (74% consensus) but magnitude estimates carry substantial uncertainty.

Judge circularity. gpt-4o-mini serves as both agent backbone and LLM-as-judge for the primary metrics, and this limitation needs two separate answers rather than one. For the implicit-need result, cross-backbone replication (where judge and agent are different models) and a directional human preference are *consistent with* a non-artifact effect but do not establish it; a fully independent judge remains future work. For factual grounding we make no accuracy claim: the strict-precision rescore narrows the margin over ReAct to non-significance ($p=0.058$; Appendix A9), which does not resolve circularity there. It confirms only that this regime is an access/privacy trade-off, not an accuracy win.

External validity. AURATown is one controlled, five-agent, deterministic-private-state setting; the implicit needs are generated by known rules and recovered by probes over the same registry. Nothing in this paper is evidence that the mechanism generalises beyond such settings, and the null cross-domain results (FANToM, Lo-CoMo, GAIA) mark where it demonstrably stops. Moving to noisier environments, with contradictory private state, tool failures, multiple users, or adversarial privacy pressure, is the necessary next test rather than an incremental extension.

Cross-backbone. Three of four tested backbones reproduce the gain; gemini-2.5-flash fails the IntentFrame JSON parser and silently falls back to a heuristic (Appendix A13).

Acknowledgments

This work is supported by the High-Level Talents Innovation Team Project of the Guangdong–Macao In-Depth Cooperation Zone in Hengqin (Grant No. 2630004018939) and the National Natural Science Foundation of China (NSFC) under Grant No. 62506084.

We thank the three anonymous ARR reviewers and the Area Chair, whose questions during the discussion period led directly to the component ablation and gap-calibration analysis reported in Section 5.2, results that changed how we describe the mechanism. We also thank the two graduate-student annotators who produced the implicit-intent subcategory labels, and the eight volunteers who completed the human evaluation.

References

- Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. 2009. [Action understanding as inverse planning](#). *Cognition*, 113(3):329–349. Reinforcement learning and higher cognition.
- Sandra Carberry. 2001. [Techniques for plan recognition](#). *User Modeling and User-Adapted Interaction*, 11(1–2):31–48.
- Eugene Charniak and Robert P. Goldman. 1993. [A bayesian model of plan recognition](#). *Artificial Intelligence*, 64(1):53–79.
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024a. [Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors](#). In *The Twelfth International Conference on Learning Representations*.
- Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and Feng Zhao. 2024b. [T-eval: Evaluating the tool utilization capability of large language models step by step](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9510–9529, Bangkok, Thailand. Association for Computational Linguistics.
- Michael C. Frank and Noah D. Goodman. 2012. [Predicting pragmatic reasoning in language games](#). *Science*, 336(6084):998–998.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.
- H. Paul Grice. 1975. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and Semantics, Vol. 3: Speech Acts*, pages 41–58. Academic Press, New York.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiaowu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. [MetaGPT: Meta programming for a multi-agent collaborative framework](#). In *The Twelfth International Conference on Learning Representations*.
- Ronald A. Howard. 1966. [Information value theory](#). *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26.
- Henry A. Kautz and James F. Allen. 1986. Generalized plan recognition. In *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, AAAI’86, page 32–37. AAAI Press.
- Hyunwoo Kim, Melanie Sclar, Xuhui Zhou, Ronan Bras, Gunhee Kim, Yejin Choi, and Maarten Sap. 2023. [FANToM: A benchmark for stress-testing machine theory of mind in interactions](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14397–14413, Singapore. Association for Computational Linguistics.
- Michał Kosinski. 2024. [Evaluating large language models in theory of mind tasks](#). *Proceedings of the National Academy of Sciences*, 121(45):e2405460121.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020.

- Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. **CAMEL: Communicative agents for "mind" exploration of large language model society**. In *Thirty-seventh Conference on Neural Information Processing Systems*.
- Yang Li, Jiaxiang Liu, Yusong Wang, Yujie Wu, and Mingkun Xu. 2026. **BMAM: Brain-inspired multi-agent memory framework**. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 39604–39626, San Diego, California, United States. Association for Computational Linguistics.
- Zongxi Li, Yang Li, Haoran Xie, and S. Joe Qin. 2025. **CondAmbigQA: A benchmark and dataset for conditional ambiguous question answering**. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 2269–2288, Suzhou, China. Association for Computational Linguistics.
- Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2024. **Agentbench: Evaluating LLMs as agents**. In *The Twelfth International Conference on Learning Representations*.
- Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. **Evaluating very long-term conversational memory of LLM agents**. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.
- Charles Packer, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph E. Gonzalez. 2023. **Memgpt: Towards llms as operating systems**. *CoRR*, abs/2310.08560.
- Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. **Generative agents: Interactive simulacra of human behavior**. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23, New York, NY, USA. Association for Computing Machinery.
- Gil Pasternak, Dheeraj Rajagopal, Julia White, Dhruv Atreja, Matthew Thomas, George Hurn-Maloney, and Ash Lewis. 2025. Beyond reactivity: Measuring proactive problem solving in LLM agents. *arXiv preprint arXiv:2510.19771*.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. **ToolLLM: Facilitating large language models to master 16000+ real-world APIs**. In *The Twelfth International Conference on Learning Representations*.
- Sudha Rao and Hal Daumé III. 2018. **Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information**. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2737–2746, Melbourne, Australia. Association for Computational Linguistics.
- Craige Roberts. 2012. Information structure: Towards an integrated formal theory of pragmatics. *Semantics and pragmatics*, 5:6–1.
- Maarten Sap, Ronan Le Bras, Daniel Fried, and Yejin Choi. 2022. **Neural theory-of-mind? on the limits of social intelligence in large LMs**. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3762–3780, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessí, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.
- Melanie Sclar, Sachin Kumar, Peter West, Alane Suhr, Yejin Choi, and Yulia Tsvetkov. 2023. **Minding language models' (lack of) theory of mind: A plug-and-play multi-character belief tracker**. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13960–13980, Toronto, Canada. Association for Computational Linguistics.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.
- Yuanbo Tang, Huaze Tang, Tingyu Cao, Lam Nguyen, Anping Zhang, Xinwen Cao, Chunkang Liu, Wenbo Ding, and Yang Li. 2026. ProAgentBench: Evaluating LLM agents for proactive assistance with real-world data. *arXiv preprint arXiv:2602.04482*.
- Thinking Machines Lab. 2026. Interaction models: A scalable approach to Human-AI collaboration. Blog post, Thinking Machines Lab. <https://thinkingmachines.ai/blog/interaction-models/>. Accessed: 2026-05-12.

- Endel Tulving. 1972. Episodic and semantic memory. In Endel Tulving and Wayne Donaldson, editors, *Organization of Memory*, pages 381–403. Academic Press, New York.
- Tomer Ullman. 2023. Large language models fail on trivial alterations to theory-of-mind tasks. *arXiv preprint arXiv:2302.08399*.
- Alexander Sasha Vezhnevets, John P Agapiou, Avia Aharon, Ron Ziv, Jayd Matyas, Edgar A Duéñez-Guzmán, William A Cunningham, Simon Osindero, Danny Karmon, and Joel Z Leibo. 2023. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*.
- Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers of computer science*, 18(6):186345.
- Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. [Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada. Association for Computational Linguistics.
- Jason D. Williams and Steve Young. 2007. [Partially observable markov decision processes for spoken dialog systems](#). *Computer Speech & Language*, 21(2):393–422.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 9 others. 2025. [The rise and potential of large language model based agents: A survey](#). *Science China Information Sciences*, 68(2):121101.
- Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See-Kiong Ng, and Jiashi Feng. 2024. [MAgIC: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7315–7332, Miami, Florida, USA. Association for Computational Linguistics.
- Bufang Yang, Lilin Xu, Liekang Zeng, Yunqi Guo, Siyang Jiang, Wenrui Lu, Kaiwei Liu, Yixuan Li, Xiaofan Jiang, Guoliang Xing, and Zhengyu Yan. 2025a. ProAgent: Harnessing on-demand sensory contexts for proactive LLM agent systems in the wild. *arXiv preprint arXiv:2512.06721*.
- Bufang Yang, Lilin Xu, Liekang Zeng, Kaiwei Liu, Siyang Jiang, Wenrui Lu, Hongkai Chen, Xiaofan Jiang, Guoliang Xing, and Zhenyu Yan. 2025b. [ContextAgent: Context-Aware proactive LLM agents with open-world sensory perceptions](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). In *The Eleventh International Conference on Learning Representations*.
- Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. 2013. [POMDP-based statistical spoken dialog systems: A review](#). *Proceedings of the IEEE*, 101(5):1160–1179.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. [Judging LLM-as-a-judge with MT-bench and chatbot arena](#). In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. [Memorybank: enhancing large language models with long-term memory](#). In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence, AAAI’24/IAAI’24/EAAI’24*. AAAI Press.
- Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. 2024. [SOTOPIA: Interactive evaluation for social intelligence in language agents](#). In *The Twelfth International Conference on Learning Representations*.
- Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. 2025. [MultiAgentBench : Evaluating the collaboration and competition of LLM agents](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8580–8622, Vienna, Austria. Association for Computational Linguistics.

A1 Bounded-Probing Algorithm

Algorithm 1 Bounded Proactive Probing (Section 3.2)

Require: Agent state s , tool registry $\mathcal{T} = \{t_1, \dots, t_K\}$, max steps B , LLM planner ϕ

Ensure: Probe result $P = (\text{summary}, \text{trace})$

- 1: $\text{context} \leftarrow \text{BUILDCONTEXT}(s)$ {agent, location, time, nearby}
- 2: $\text{trace} \leftarrow \langle \rangle$ {empty probe trace}
- 3: **for** $i = 1$ **to** B **do**
- 4: $d \leftarrow \phi(\mathcal{T}, \text{context}, \text{trace})$ {LLM decides: call a tool or stop}
- 5: **if** $d.\text{action} = \text{stop}$ **then**
- 6: **break**
- 7: **end if**
- 8: $r \leftarrow \mathcal{T}.\text{EXECUTE}(d.\text{tool}, d.\text{args})$
- 9: $\text{trace}.\text{APPEND}((d.\text{tool}, d.\text{args}, r))$
- 10: $\text{context} \leftarrow \text{UPDATE}(\text{context}, r)$
- 11: **end for**
- 12: $\text{summary} \leftarrow \text{SUMMARIZE}(\text{trace})$
- 13: **return** (summary, trace)

A2 Adaptive-Budget Evidence (Supplementary)

This appendix expands the adaptive-budget claim summarised in Section 5.2: under a fixed global `explore_max_steps=3` ceiling, the per-query probe count actually issued by AURA varies across subcategories by $2.8\times$, and is driven by the `IntentFrame`’s gap field rather than the configured ceiling. Figure 4 shows the per-subcategory implicit-need scores, Figure 3 the gap-vs-probes scatter, and Table 5 the per-subcategory means with paired latencies.

A2.1 Adaptive Budget: LLM-Directed Probe Allocation

A practical concern for any bounded-probing system is whether the configured probe ceiling simply becomes a fixed per-query cost. We therefore measure the *effective* number of probes issued after the `IntentFrame` sets a ceiling and the `Explore` loop decides whether to stop early. This analysis does not try to define autonomous-agent behavior; it checks the narrower mechanism claim that intent inference changes probe allocation across queries.

Using the same $25\text{-query} \times 3\text{-condition} \times 3\text{-seed}$ run as Section 5.2 (for AURA-full only; $N = 75$), we measure the probe-count distribution

Table 5: Per-subcategory distributions under AURA Full (fixed `explore_max_steps=3`, $N = 15$ per row: 5 queries $\times$ 3 seeds). Gap is the `IntentFrame`’s literal/implicit divergence; Probes is the number of tool calls actually issued after early stopping.

Subcategory	Gap	Probes	Score	Lat.
availability	0.48 ± 0.04	0.80 ± 0.41	0.79	4.5
mood	0.50 ± 0.00	1.40 ± 0.51	0.81	6.0
appropriateness	0.68 ± 0.06	2.20 ± 0.68	0.84	6.1
latent_goal	0.58 ± 0.07	1.47 ± 0.74	0.81	6.0
second_order	0.80 ± 0.00	1.00 ± 0.00	0.95	5.5

conditional on query subcategory, while holding `explore_max_steps=3` constant across all runs. The configured budget is therefore a ceiling, not a target.

Observed allocation patterns. (i) *Effective probe count is not constant.* Mean probe count per query ranges from 0.80 (availability) to 2.20 (appropriateness), despite the same global `explore_max_steps=3` setting.

(ii) *All four budget values are used.* Across the 75 runs, the agent issued $\{0, 1, 2, 3\}$ tool calls—the full effective range allowed by the run. This is consistent with the intended design: the configured budget is a ceiling, not a target.

(iii) *Gap is an input, not the decision itself.* The `second_order` subcategory has the highest observed gap (0.80) yet uses only 1.00 probe on average. The gap-to-budget rule maps 0.80 to a ceiling of three probes, but the `Explore` loop usually stops after one targeted belief-state call because that call already returns the needed value. The Pearson correlation between gap and actual probe count is $r = 0.19$, so the gap is not a proxy for the final number of tool calls.

Taken together, these traces support the mechanism claim that `IntentFrame`-conditioned probing changes the cost and evidence gathered for different queries under the same run configuration.

Scope of the claim. The architecture remains a hybrid system: `Perceive`, `Scene`, and `Memory` are code-determined, while intent inference, probe selection, and response generation use LLM calls. The adaptive-budget result should therefore be read as evidence about one control point in the system, not as a broad claim about every stage of the pipeline.

Gap-vs-benefit calibration (bin-wise). Section 5.2 reports that the gap score is a probe trigger rather than a calibrated estimate of probe

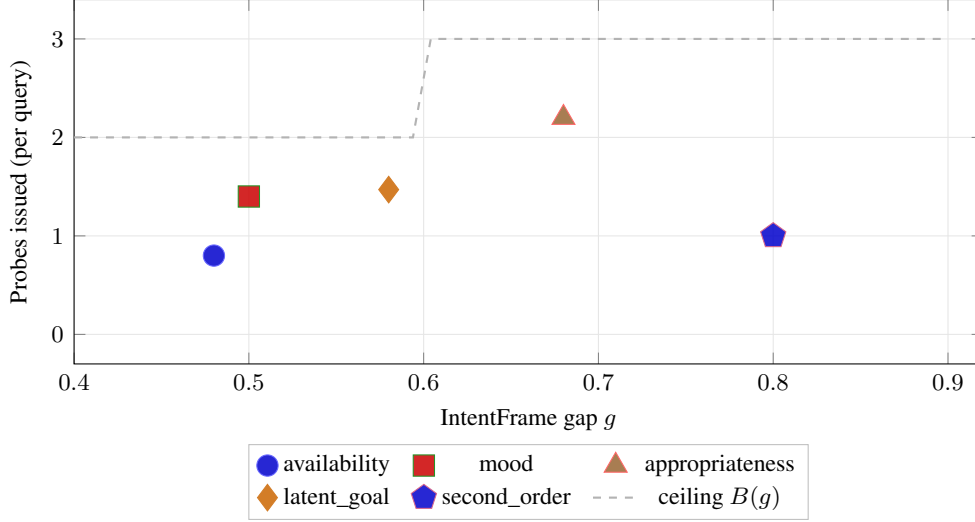


Figure 3: Per-subcategory mean of IntentFrame .gap vs. probes actually issued by the LLM (5 queries $\times$ 3 seeds per marker; explore_max_steps=3). The dashed line is the deterministic ceiling map $B(g)$ from Section 3.3, capped at the configured ceiling 3. Probe count varies $2.8\times$ (0.80–2.20) under identical configuration; all four budget values $\{0, 1, 2, 3\}$ appear; Pearson $r(g, \text{probes})=0.19$. The *second_order* subcategory has the highest gap but issues only 1 probe because one targeted get_agent_belief_about call already returns the belief.

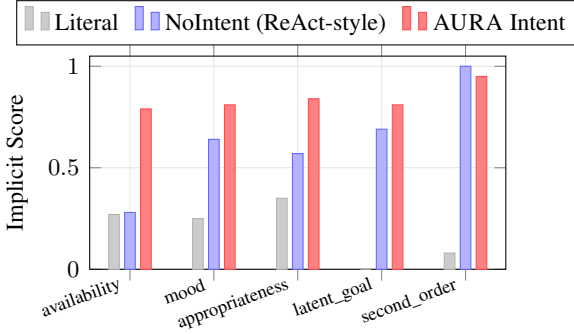


Figure 4: Per-subcategory implicit-need surfacing on the implicit-intent benchmark (25 queries, 3 seeds, $N=15$ per cell). AURA Intent improves most on lexically decoupled private-state queries (e.g. *availability*); on *second_order* queries (“does X think Y?”) the surface already cues belief retrieval and NoIntent ties.

value. Table 6 gives the bin-wise evidence: over the 300 paired query–seed cells of the 100-query benchmark, the mean benefit of intent-directed probing (tom – no_intent implicit score) falls monotonically as the inferred gap rises, and the highest-gap bin is negative. Pearson $r=-0.33$, Spearman -0.32 . Regenerate with scripts/gap_calibration_analysis.py.

A3 Private-State Evolution Rules

Private fields on each agent (availability, emotional_state, unspoken_goal, beliefs_about_others) update each sim-

Table 6: Mean benefit of intent-directed probing by inferred-gap bin ($n=300$ query–seed cells; 95% CI from the cell distribution).

Gap bin	n	Mean benefit	95% CI
[0.4, 0.5)	108	+0.167	[+0.116, +0.217]
[0.5, 0.6)	84	+0.048	[+0.004, +0.092]
[0.6, 0.7)	77	+0.010	[−0.042, +0.063]
[0.7, 0.8]	31	−0.045	[−0.096, +0.005]

Table 7: Per-subcategory implicit score ($n=15$ per cell, 5 queries $\times$ 3 seeds). Intent inference’s marginal gain concentrates on availability queries (surface hides implicit need) and ties NoIntent on second-order queries (surface cues “X thinks Y”).

Subcategory	Literal	NoIntent	AURA Intent
availability	0.27	0.28	0.79
mood	0.25	0.64	0.81
appropriateness	0.35	0.57	0.84
latent_goal	0.00	0.69	0.81
second_order	0.08	1.00	0.95

ulation tick under a deterministic, transparent rule table (source: demo/town/private_state_evolution.py). The intent is to make AURATown’s private state a live function of agent action and environment context rather than a static lookup. Rules are evaluated in order; the first match wins.

Beliefs-about-others refresh. Each agent’s beliefs_about_others dictionary entries re-

Table 8: Private-state evolution rules (first match wins). Action keywords match against `current_action` via simple regex; workplace map: Lin Wei→Sunrise Cafe, Chen Mei→General Store, Zhang Hao→home/library, Liu Yang/Wang Jun→Library.

Rule	Trigger	Outcome
sleep	action matches sleep	DND / resting / no goal
workplace_loaded	at workplace, ≥ 3 peers, busy-action keyword	busy / tired-focused / “close out the rush”
deep_focus	action matches writ draft stud research meditat	DND / creatively-flowing / writing-milestone
workplace_empty	at workplace, 0 peers, in opening hours	available / lonely / “hoping a regular drops by”
relaxed	action matches walk read tai chi eat sleep	available; emo recovers from stress, persists from loneliness
recent_stress	last 8 events contain argument emergency failed broke worried	available / stressed
default	none of the above	available / neutral

fresh *only* when this agent is co-located with the target agent in the same simulation tick. Beliefs about non-co-located agents remain at the most recently observed value, so they go stale when peers move. This is the substrate the *second_order* subcategory of the implicit-intent benchmark (§5.2) probes: a query like “does Lin Wei think Zhang Hao is at home?” asks for the believer’s stale memory, not the target’s current ground-truth.

The full rule table is unit-tested at `tests/test_private_state_evolution.py` (14 cases, all rules + co-location/staleness invariants). The evolution itself is pure-function and zero-LLM, so it is reproducible across seeds and is not a confound when ablating other mechanisms.

A4 AURATown Agent Profiles and Map

AURATOWN is a 60×60-grid simulation of five named agents living in twenty named locations (homes, commerce, civic, parks). The five agents (Table 9) and their starting locations (Figure 5) are fixed across all experiments. Each agent has both public state (location, current action, nearby agents) visible to all queries and private state (availability, emotional_state, unspoken_goal, beliefs_about_others) accessible only via probe tools.

A5 Probe Tool Descriptions

Factual grounding (§5.1) uses an eight-tool environment registry (Table 10); each tool returns a structured fragment of the simulation state. The implicit-intent setup (§5.2)

Table 9: Agent profiles in AURATOWN.

Name	Age	Occupation	Personality
Lin Wei	32	Cafe Owner	Warm, social
Zhang Hao	28	Writer	Introverted, observant
Chen Mei	45	Shop Owner	Practical, connector
Liu Yang	20	Student	Curious, idealistic
Wang Jun	68	Retired Prof.	Wise, mentor-like

Table 10: Environment tools available to the probe planner.

Tool	Args	Description
<code>world.time</code>	–	Sim time and day
<code>world.location</code>	–	Agent’s location details
<code>world.nearby_agents</code>	limit	Agents at same location
<code>world.agents_summary</code>	limit	All agents’ locations
<code>memory.recent</code>	limit	Agent’s recent memories
<code>world.events_recent</code>	limit	Recent global events
<code>agent.plan</code>	–	Agent’s current plan
<code>world.location_info</code>	location	Named-location details

uses a separate five-tool scripted registry (`get_all_agents`, `get_nearby_agents`, `get_agent_plan`, `get_agent_private_state`, `get_agent_belief_about`) over a fixed scene snapshot rather than event-history. The two registries are disjoint by design — the factual-grounding registry probes the world state through scene-aware tools, the implicit-intent registry probes individual agents’ public and private states. The chat/demo deployment additionally exposes both registries, so a deployed agent has access to both base environment tools and private-state/belief probes.

A6 Implementation Details

This section records the LLM, memory, simulation, and infrastructure parameters that hold throughout the experiments. All numbers report mean±std over the three seeds {42, 123, 456} unless explicitly noted.

LLM Configuration. All LLM calls use `gpt-4o-mini` via the OpenAI API with:

- Action decision: temperature 0.7, default max tokens

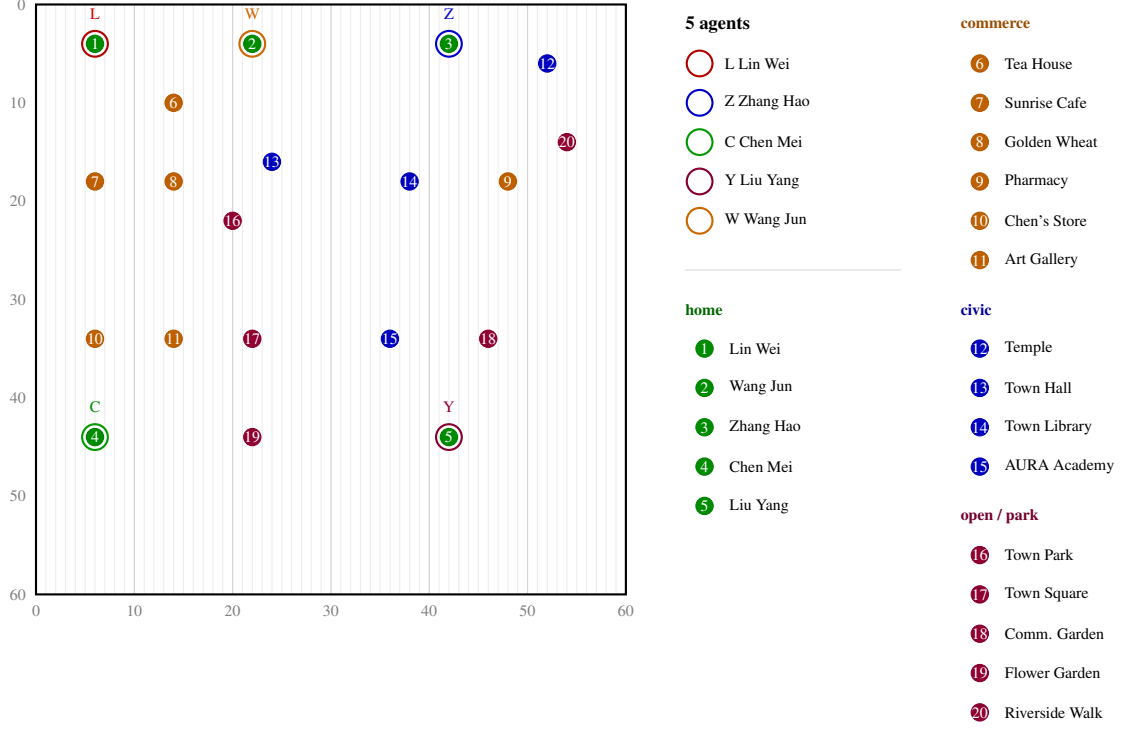


Figure 5: AURATOWN — 60×60 grid with 20 named locations at their real coordinates (from demo/town/assets/town_map.json), coloured by type (home 5, commerce 6, civic 4, open/park 5). Rings labelled L/Z/C/Y/W mark the five agents at their starting homes per demo/town/agents.py. Public state (location, action, nearby agents) is visible in the scene snapshot; private state (availability, emotional_state, unspoken_goal, beliefs_about_others) is hidden and only retrievable via the probe tools listed in Table 10.

- Probe planning: temperature 0.2, max tokens 200
- Conversation generation: temperature 0.8, max tokens 600
- Reflection: temperature 0.5, default max tokens
- Importance scoring: temperature 0.1, max tokens 50

Memory Configuration. Maximum 200 items per agent, retrieval weights $w_r = 0.3$, $w_p = 0.4$, $w_v = 0.3$, recency decay $\lambda = 0.01$, reflection threshold $\theta = 10$.

Simulation Parameters. 30-minute ticks, 6:00–23:00 day cycle, conversation cooldown 2 ticks, probe cooldown 2 ticks, movement speed 3 grid units/tick.

Infrastructure. Python backend serving HTTP API on port 7861 with threading for concurrent state/chat/step requests. React frontend with canvas-based pixel-art rendering, viewport camera system, and real-time state updates.

A7 Concrete Examples, Comparisons, and Diagrams

This appendix collects the figures and the proactive-frameworks comparison referenced from the body but moved out for space.

Motivating contrast (Figure 6). A vanilla LLM and AURA on the same query in the same scene state. Vanilla returns the literal location only. AURA infers an implicit availability need, probes the cafe scene for nearby agents and the target’s private state, and returns the location with a heads-up alert that she is currently busy.



Figure 6: Motivating contrast on “Where is Lin Wei?”. Vanilla (left) vs. AURA (right) on the same scene state.

IntentFrame example (Figure 7). An example IntentFrame produced by the LLM-backed IntentInferer on a single AURATown query. Six fields are emitted (literal_need, implicit_need, gap, recommended_probes, should_alert, confidence) and consumed by the Explore and Interact stages (§3.3).



Figure 7: The IntentFrame produced for a single user query.

Long-context prompt-content sketch (Figure 8). Figure 8 sketches the token-cost difference between packing the full environment state into the prompt and AURA’s selective probing on the same AURATOWN scene. Token counts from the AURATOWN prompt log; latency and FA from Tables 1, 2. This is an illustrative sketch, not a head-to-head experiment: the Static-Context baseline (FA 0.450) is the closest in-suite analogue but does not include private state or memory, so a true full-state long-context baseline is left for future work.

Proactive frameworks comparison (Table 11). Table 11 positions AURA relative to recent proactive and tool-using agent frameworks along three orthogonal axes (decision target, trigger, selection) and reports per-query latency for frameworks we re-implemented under our backbone. “–” marks frameworks that address an orthogonal decision (*when to assist* from sensor/behaviour patterns) and were not re-run on our factual benchmark.

A8 Experimental Setup (Full)

Baselines. We compare AURA against six baselines on the factual-grounding benchmark (Table 1) and four on the implicit-intent benchmark (Table 2):

1. **Vanilla LLM:** direct gpt-4o-mini generation with no environmental context (factual grounding).

(A) Long-context model

Pack everything into the prompt; let the LLM attend.

Prompt content (per query):

<scene snapshot>	~ 180 tok
<5 agents × public state>	~ 300 tok
<5 agents × private state>	~ 420 tok
<recent events / memories>	~ 650 tok
<system + few-shot>	~ 500 tok

→ ~ 2050 prompt tokens / query

→ $O(\#agents)$ growth as the world scales

Trade-off: pays for *all* state every query, even when the user’s literal need is one location lookup. Long-context attention dilution can lower precision on packed prompts.

Static-Context baseline (factual grounding): FA 0.450, 1.8 s.

(B) AURA (gap-routed selective probing)

Infer the gap; fetch only what closes it.

Prompt content (per query):

<scene snapshot>	~ 180 tok
<IntentFrame: literal/implicit/gap>	~ 210 tok
<probe results × 1.0~2.2>	~ 120–280 tok
<system + few-shot>	~ 500 tok

→ ~ 1010–1170 prompt tokens / query

→ $O(\#probes)$ = 0–3, gap-conditioned

Trade-off: pays one extra LLM call (the IntentInferer) but skips fetching irrelevant private state. Targeted probe returns are more easily decoded than a packed prompt.

AURA Full (factual grounding): FA 0.640, 3.9 s;

AURA Intent (implicit-intent): implicit-score 0.803, 13.8 s.

Figure 8: AURA versus a long-context approach on AURATOWN. A long-context model pays the full per-agent public+private+memory cost per query (panel A); AURA’s IntentFrame routes selective probing so the prompt grows with the inferred gap, not the number of agents (panel B). Token estimates are from the AURATOWN prompt log; FA / latency from Tables 1, 2. Static-Context (the long-context lower-bound in our suite) reaches FA 0.450; AURA Full reaches 0.640.

2. **Static Context:** LLM receives a packed scene snapshot (location, time, nearby agents) but no proactive information gathering (factual grounding).
3. **ReAct Agent (Yao et al., 2023):** Thought–Action–Observation interleaved *during* reasoning. Same eight base environment tools as AURA on factual grounding; same five-tool scripted registry on the implicit-intent benchmark.
4. **Reflexion (Shinn et al., 2023):** ReAct + self-reflection, up to 2 retry rounds (factual grounding).
5. **Plan-and-Solve (Wang et al., 2023):** Phase-1 plan-from-surface-query, Phase-2 execute, Phase-3 synthesise (factual grounding + implicit-intent benchmark).
6. **AURA (No Probe):** AURA architecture with

Table 11: Where AURA sits among proactive / tool-using agent frameworks. “Trigger” = when the framework decides to act; “Selection” = how it decides what to fetch; “Latency” = mean per-query wall-clock when run on our 50-query factual benchmark with backbone gpt-4o-mini, or “–” if not re-implemented here.

Framework	Decision target	Trigger	Selection	Latency
ContextAgent (Yang et al., 2025b)	when to surface a hint	sensor stream change	multi-dim. context extract.	–
ProAgent (Yang et al., 2025a)	predict user need	AR-glass sensor + profile	hierarchical context fusion	–
PROBE (Pasternak et al., 2025)	search–identify–resolve	idle / unspecified issue	three-stage pipeline	–
ProAgentBench (Tang et al., 2026)	benchmark (timing+content)	N/A (real-user sessions)	N/A	–
ReAct (Yao et al., 2023)	during-reasoning act	per LLM step	LLM picks per step	6.0 s
Reflexion (Shinn et al., 2023)	during + retry	per step + reflect	LLM picks per step	20.6 s
Plan-and-Solve (Wang et al., 2023)	plan-then-execute	once, from surface query	surface-query plan	5.2 s
AURA (ours)	what to fetch	post-query, routed	gap- gap → probe set	3.9 s

the probe budget forced to 0 (factual-grounding ablation).

7. AURA (Full): complete pipeline with IntentFrame-derived budget and tool selection.

We do not re-run Generative Agents (Park et al., 2023) or ContextAgent (Yang et al., 2025b) as paired baselines: Generative Agents’ passive observation string is subsumed by our Static Context; ContextAgent targets a different decision (*when to assist* from sensor streams) and is discussed as related work in Appendix A7 rather than an in-suite comparator.

Models and metrics. Backbone gpt-4o-mini, temperature 0.7; gpt-4o-mini LLM-as-judge at temperature 0.1 (Zheng et al., 2023), combined with a rule-based pre-filter on location and time consistency. Judge and agent share the same model family (gpt-4o-mini), which reduces judge-policy independence; we mitigate by reporting a strict precision rescore (Appendix A9) that uses only the judge’s CORRECT/CONTRADICTED claim-level classifications and drops the soft completeness component. Metrics: *Grounding Accuracy* (GA, 5 sub-dimensions, inspired by AgentBench (Liu et al., 2024)), *Factual Accuracy* (FA, 50 environment-grounded queries in 5 categories), *SOTOPIA Social Score* (7 dimensions from Zhou et al. (2024)), *Context Utilization* (CU), *Latency*.

Table 12: Strict rescore over the same 50 factual-grounding queries $\times$ 3 seeds (per-query, per-seed cells). *StrictP* = correct/(correct + contradicted) excluding cells where the judge produced only unverifiable claims. *Halluc. %* = fraction of cells with ≥ 1 contradicted claim. *Perfect %* = 0 contradicted AND ≥ 1 correct. *Lenient FA* = the original $0.7 \cdot P + 0.3 \cdot C$ from Section 5.1 for comparison.

Method	StrictP	Recall	Halluc. %	Perfect %	Lenient FA
Vanilla LLM	0.036	0.033	93.3	1.3	0.070
Static Context	0.459	0.396	70.0	26.7	0.450
ReAct	0.578	0.534	51.3	38.0	0.550
AURA (NoProbe)	0.581	0.519	75.3	24.0	0.603
AURA (Full)	0.658	0.540	66.7	28.0	0.640

A9 Strict Precision Rescore for Factual Grounding (Robustness Check)

The factual accuracy (FA) reported in Section 5.1 uses an LLM-as-judge whose final score combines a hard precision component (correct/(correct+contradicted) over the judge’s claim-level classifications, weight 0.7) with a soft *completeness* component (a 0/0.5/1 rubric for whether the response addressed the question, weight 0.3). Soft components are sensitive to phrasing and judge mood.

We rescore the same per-query records with strict precision only, drop the completeness term, and additionally report the per-query *hallucination rate* (fraction of (query, seed) cells with ≥ 1 contradicted claim) and *perfect-response rate* (0 contradicted AND ≥ 1 correct). The rescore script and output JSON are listed in Appendix A25.

Three honest takeaways relative to the lenient table in Section 5.1:

Table 13: Per-category strict precision (mean across query-seed cells, $n \leq 30$ per cell).

Category	Vanilla	Static	ReAct	NoProbe	Full
memory	0.000	0.347	0.547	0.695	0.740
planning	0.034	0.730	0.453	0.728	0.691
social	0.069	0.275	0.583	0.291	0.470
spatial	0.034	0.448	0.591	0.390	0.472
temporal	0.036	0.491	0.717	0.798	0.892

- **vs. Vanilla:** the architectural-effect headline holds and gets larger (+0.609 strict precision, $p < 10^{-4}$).
- **vs. ReAct (the fair-tools baseline):** the strict precision gap is +0.090 ($p=0.058$, query-level paired t -test, $n=49$ complete pairs) — not significant. AURA also has a higher hallucination rate than ReAct (66.7% vs. 51.3%, $\Delta = +15.3$ pp, $p=0.013$). Reading these two together, the architectural pipeline’s contribution *above* a tool-using ReAct baseline is small under strict scoring.
- **vs. AURA NoProbe:** strict precision rises by +0.062 ($p=0.032$); the lenient version of this contrast was $p=0.299$. The probing mechanism is more clearly supported under strict scoring than under the soft FA, while the overall architectural margin shrinks.

The per-category strict precision (Table 13) localises the gains: AURA (Full) wins on temporal (0.892) and memory (0.740) categories and loses to ReAct on social and spatial. The original “social win for probing” from the lenient analysis does not survive strict scoring; we update Section 5.1’s scope condition accordingly in the discussion.

A10 Privacy-Sensitive Distractor Slice

The factual-grounding primary FA metric rewards factual completeness but does not penalise unnecessary access. We therefore authored a 30-query factual slice whose gold answers require public facts only. Each query carries a forbidden_tools list over high-disclosure tools (memory.recent, world.events_recent, agent.plan) plus query-specific extras; a violation is any fired forbidden tool in the query-seed cell. We run the same three seeds as the factual-grounding benchmark.

Against GapRouted, Fixed-Probe gains +0.080 FA only marginally ($p=0.056$) while adding +100 pp forbidden-tool violations. Plan-and-Solve and

Table 14: Privacy-sensitive factual slice (30 queries $\times$ 3 seeds). *Viol. %* is the fraction of query-seed cells where any forbidden tool fired.

Method	FA	Viol. %	Probes/q
Fixed-Probe	0.672	100.0	8.00
ReAct	0.618	25.9	1.87
Plan-and-Solve	0.603	78.9	4.16
AURA (GapRouted)	0.592	0.0	0.73
Static Context	0.493	0.0	0.00
Vanilla LLM	0.069	0.0	0.00

Table 15: Wall-clock per query (seconds; median / mean). Factual columns aggregate the 50q factual benchmark; Privacy is the 30q distractor slice; Implicit-Intent v1 is the 25q implicit-need set under clean prompt. Probe count is the per-condition mean tool-call count (factual / Privacy use the eight-tool registry; implicit-intent uses the five-tool probe registry). AURA GapRouted pays an extra IntentInferer LLM round-trip (≈ 2 s) over a one-shot baseline; the cost does not scale with the probe count.

Condition	Factual		Privacy slice		Implicit-Intent v1	
	Lat	Probes	Lat	Probes	Lat	Probes
Vanilla LLM	2.14 / 2.28	0	1.83 / 2.01	0	—	—
Static Context	1.64 / 1.77	0	1.54 / 1.59	0	—	—
Literal	—	—	—	—	1.23 / 1.29	0
NoIntent (ReAct)	5.10 / 6.01	1.05	3.18 / 4.87	1.87	1.95 / 2.20	1.05
Plan-and-Solve	4.75 / 5.19	4.64	4.46 / 4.51	4.16	—	—
Reflexion	16.45 / 20.64	2.15	—	—	—	—
Fixed-Probe	2.37 / 3.23	8.00	1.47 / 1.65	8.00	—	—
AURA GapRouted	4.08 / 4.34	1.40	3.97 / 3.99	0.73	5.31 / 13.85[†]	1.48

ReAct are tied in FA with GapRouted ($p=0.856$ and $p=0.655$) but incur +78.9 pp and +25.6 pp violations. This slice makes the factual-grounding tradeoff explicit: saturated access is often accurate, but it violates stated access constraints by construction; gap routing is not FA-dominant, but it operates on the low-violation, low-probe side of the Pareto frontier.

A11 Cost and Latency Across Conditions

The Pareto framing in Section 5.1 ranks conditions on access cost (probes) and disclosure, not wall-clock. Table 15 reports median and mean per-query latency from the same multi-seed runs so the wall-clock side of the tradeoff is auditable. Medians guard against rare upstream-API timeouts that inflate means (most pronounced for the implicit-intent v1 *tom* row, where one 604 s API hiccup on seed 456 pulls mean to 13.85 s while median stays at 5.31 s).

[†] Median is more representative; mean is inflated by one 604 s upstream-API outlier across the 75 query-seed cells.

Two observations qualify the Pareto picture.

First, AURA GapRouted is *not* the fastest condition despite firing the fewest probes: the IntentInferer is a single extra LLM call (≈ 2 s on gpt-4o-mini), so GapRouted’s median latency sits above Fixed-Probe (4.08 vs. 2.37 s on factual grounding; 3.97 vs. 1.47 s on the privacy slice). The probe-count win does not translate into a wall-clock win at small budgets. Second, Reflexion’s median latency (16.5 s) is the regime outlier; its retry loop pays the cost without buying accuracy. We treat latency as a reportable axis rather than a paper claim: the cost-of-selectivity story holds on probe count and disclosure, not on wall-clock.

A12 Prompt Ablation for IntentFrame Calibration

We audit whether the IntentFrame gain comes from benchmark-overlapping few-shot examples or from calibrated gap estimation. Table 16 compares three prompt variants on the same 25 implicit-intent queries and three seeds. The leaked row uses the original benchmark-overlapping examples and is reported only for diagnosis; the clean row is the final system; the no-few-shot row keeps the same rubric but removes all examples.

Table 16: Prompt ablation for the IntentFrame controller. p_q is the query-level paired test for Intent vs. NoIntent.

Variant	Intent	NoIntent	Δ	p_q
Leaked few-shot	0.840	0.637	+0.203	7.2×10^{-4}
Clean few-shot	0.803	0.640	+0.163	0.006
No few-shot	0.677	0.643	+0.035	0.44

Clean few-shot is only 0.037 below the leaked prompt, so the main effect is not driven by memorising names or locations. In contrast, removing examples reduces Intent to 0.677 and makes the Intent-vs.-NoIntent gain non-significant. Mechanistically, the no-few-shot prompt underestimates the gap: mean inferred gap drops from 0.589 to 0.476, and high-gap cells ($g \geq 0.60$) drop from 43/75 to 22/75. The few-shot examples are therefore load-bearing calibration for gap-to-budget routing.

A13 Cross-Backbone Intent-vs.-NoIntent (Robustness)

To check that the Intent-vs.-NoIntent gain reported on gpt-4o-mini is not specific to one backbone, we re-ran the 25 implicit-intent queries on three additional production-grade LLMs (each via its

vendor’s official API; judge fixed at gpt-4o-mini to keep the scoring rubric constant). Table 17 summarises the result. The gpt-4o-mini row is the clean calibrated prompt used in the main paper; the other backbones are single-seed robustness probes. Three of four backbones reproduce the gain; gemini-2.5-flash regresses, but the regression is explained by JSON-schema parse failure on 23/25 IntentFrame calls and silent fallback to the deterministic heuristic — a format-compliance boundary.

Table 17: Cross-backbone Intent-vs.-NoIntent contrast on implicit score (25 implicit-intent queries, query-level paired comparison with cluster bootstrap on query_id). The gpt-4o-mini row is the clean calibrated prompt and averages over 3 seeds; the other rows are single-seed robustness probes. Three of four backbones reproduce the gain; gemini-2.5-flash fails the JSON parser on 23/25 IntentFrame calls (run log_run_gemini25flash.log) so its row reflects heuristic-fallback under a Gemini tag.

Backbone	Vendor	NoIntent	AURA Intent	Δ (95% CI)
gpt-4o-mini	OpenAI	0.640	0.803	+0.16 [+0.06, +0.27]
claude-haiku-4-5	Anthropic	0.680	0.920	+0.24 [+0.12, +0.36]
qwen-plus	Alibaba	0.736	0.984	+0.25 [+0.14, +0.37]
gemini-2.5-flash	Google	0.600	0.400	-0.20 [-0.37, -0.03]

A14 Expanded Implicit-Intent v2 Details

To test whether the 25-query scene is over-specialised, we authored an expanded implicit-intent v2 set with 4 scene snapshots $\times$ 5 subcategories $\times$ 5 queries (100 queries; 300 scored cells per condition). Scene A preserves the original 25 queries; scenes B–D vary location, time of day, agent rosters, private states, and stale belief-vs-truth mismatches. Each v2 query records gold_required_tools and forbidden_tools; all second-order queries forbid direct get_agent_private_state access because the correct evidence is the believer’s recorded belief, not the target’s ground truth.

Subcategory examples. Representative surface queries by subcategory: *availability* “where is Lin Wei?” (implicit: “is she free?”); *mood* “how is Chen Mei today?” (implicit: “is she in a receptive emotional state?”); *appropriateness* “is now a good time to invite Lin Wei for coffee?” (requires integrating schedule and private state); *latent_goal* “what is Lin Wei up to?”; *second_order* “does Lin Wei think Zhang Hao is free?” (the correct answer must report the believer’s recorded belief, not the target’s ground truth).

Inter-annotator agreement. Two independent annotators (computer-science graduate students, distinct from the authors and naive to the AURA architecture) re-labelled the 25-query pilot set under the 5-subcategory definitions given in the task instructions. They reached Cohen’s $\kappa=0.61$ (substantial under Landis-Koch; raw agreement 68%, 17/25). All 8 disagreements concentrate on two boundaries: (i) *mood* vs. *appropriateness/availability* (4/8; e.g., “Does Zhang Hao look busy?” admits both an availability reading and a mood reading), and (ii) *appropriateness* vs. *literallavailability* (3/8; e.g., “Can I ask Wang Jun for a favor right now?” splits between a literal-permission reading and a context-aware appropriateness reading). Per-class agreement is highest on *second_order* (24/25 between annotators) and *latent_goal* (24/25); collapsing *mood* into a single *context-aware availability* super-category yields $\kappa=0.68$. Raw label dumps are released alongside the queries.

Table 18: Expanded implicit-intent v2 check (100 queries $\times$ 3 seeds, gpt-4o-mini). Scores are implicit-need coverage. The final two rows report AURA Intent minus NoIntent; p values are paired tests over query-seed cells.

	Overall	A: cafe	B: library	C: garden	D: night
Literal	0.223	0.200	0.264	0.237	0.192
NoIntent	0.733	0.709	0.688	0.731	0.803
AURA Intent	0.804	0.800	0.779	0.811	0.827
Intent–NoIntent Δ	+0.071	+0.091	+0.091	+0.080	+0.024
p	1.0×10^{-6}	0.004	1.4×10^{-4}	0.015	0.349

Scene-level read. Three of four scenes are clearly positive (A/B/C), and the post-event night scene D ties: agents are spatially distributed in D and the public state already telegraphs availability, so the gap mechanism has nothing to add.

Subcategory breakdown (Intent vs. NoIntent, cell-level $n=60$ per cat). *availability* +0.29 ($p=2.7 \times 10^{-11}$), *appropriateness* +0.11 ($p=8.2 \times 10^{-4}$), *mood* +0.07 ($p=1.7 \times 10^{-3}$), *second_order* –0.02 ($p=0.32$, NS) after a synthesis-prompt fix that drops the public-state dump on belief-state queries and adds a strict “report belief, not actual state” instruction, and *latent_goal* –0.09 ($p=6.2 \times 10^{-4}$, residual deficit acknowledged in Limitations).

Cross-backbone v2. Re-running the 100-query set on claude-haiku-4.5 (seed 42 only, via

OpenRouter) reproduces the gain at *larger* magnitude: Intent 0.876 vs. NoIntent 0.790, paired $\Delta=+0.086$, $p=3.7 \times 10^{-3}$. The scene-D tie pattern replicates ($\Delta=+0.008$, $p=0.88$, NS), confirming the night-scene null is a property of the regime rather than a single-backbone artifact.

A worked failure case (*latent_goal*). The deficit has a consistent shape, visible in a single query. Asked “is anyone setting up something for tomorrow that I should know about?”, with one agent privately intending to invite another on a hike the next day, AURA inferred a high gap, which bought it a full probe ceiling, and then spent that ceiling on a single agent, missing the intent entirely. The NoIntent baseline enumerated the roster first and found it. Naming no subject is what makes such a query hard, and it is also what the shortlist handles worst: a narrow, confident probe plan is a liability when the question ranges over everyone. Across the queries that name no subject, AURA probes fewer distinct agents than either the baseline or its own budget-only ablation, so the deficit is one of allocation rather than of budget, and raising the ceiling does not address it.

A15 Factual-Grounding Per-Category Paired Contrast

The aggregate AURA Full vs. AURA No-Probe contrast on factual grounding is a near-null (+0.038 FA, $p=0.060$ paired query-level), but this aggregate hides a heterogeneous per-category structure. Splitting the 50 queries into 5 categories of 10 (spatial / social / temporal / memory / planning), Table 19 shows that probing produces a significant FA improvement only on the *social* category. The other four categories are saturated by the Perceive/Scene channel, so probing has nothing to add. This is the empirical scope condition for proactive probing referenced in the main text.

Table 19: Per-category paired t -test of proactive probing (AURA Full vs. AURA No-Probe), $N=10$ queries per category per seed, 3 seeds. Probing produces a significant effect only on the *social* category.

Category	Full FA	NoProbe FA	Δ	p
spatial	0.457 ± 0.027	0.453 ± 0.130	+0.004	0.96
social	0.459 ± 0.056	0.322 ± 0.070	+0.137	0.010*
temporal	0.846 ± 0.060	0.813 ± 0.016	+0.033	0.51
memory	0.707 ± 0.092	0.684 ± 0.031	+0.023	0.72
planning	0.734 ± 0.031	0.741 ± 0.072	–0.007	0.90

A16 Routine-Action Grounding (Null Result, Full Table)

The routine-grounding check measures whether richer environmental access changes Grounding Accuracy on routine daily simulation. The protocol is 100 simulation steps $\times$ 5 conditions $\times$ 3 seeds = 1,500 judgments per condition; the metric averages four GA sub-dimensions (location consistency, time appropriateness, social awareness, plan adherence). Memory utilisation saturates at ≈ 1.0 for every method and is omitted from Table 20. The paired t -test column reports two-sided p vs. Vanilla. All five methods fall within 0.024 absolute GA spread; every contrast is non-significant. We report this null openly: when most actions are trivially grounded, GA cannot distinguish architectures.

Table 20: Grounding Accuracy (GA) on 100-step routine daily simulation, 3 seeds, mean $\pm$ std. Dimension scores: location consistency, time appropriateness, social awareness, plan adherence.

Method	GA	Loc.	Time	Social	Plan	Lat. (s)	p vs. Van.
Vanilla LLM	0.659 $\pm$ 0.030	0.425	0.371	0.877	0.621	9.8	—
Static Context	0.659 $\pm$ 0.007	0.401	0.384	0.835	0.676	12.9	0.99
ReAct Agent	0.652 $\pm$ 0.015	0.537	0.288	0.911	0.527	29.3	0.73
AURA (No Probe)	0.676 $\pm$ 0.008	0.459	0.405	0.837	0.677	16.4	0.51
AURA (Full)	0.665 $\pm$ 0.005	0.421	0.375	0.858	0.671	25.2	0.76

All pairwise paired t -tests vs. Vanilla give $p > 0.5$; a non-parametric Wilcoxon signed-rank test gives the same conclusion. The GA spread across five different architectures is 0.024, well inside per-condition seed variance. This is evidence the metric is saturated on this workload, not that probing fails; factual grounding (Section 5.1) reaches a $9.1\times$ gap on queries that stress grounding.

A17 Additional Diagnostic Checks

A17.1 Component Ablation (Saturated-Workload Null)

We ablate each pipeline component on a routine-action workload of 100 simulation steps and 50 chat queries per configuration, three seeds, source data evaluation/results/rq3_ablation_study_multiseed.json. Table 21 gives the multi-seed mean GA / FA / latency deltas relative to AURA Full. (An earlier draft of this table reported single-seed numbers measured at 20 steps and 20 queries; we replace it here with the full multi-seed configuration to match the actual run.)

Table 21: Component ablation, multi-seed mean (Δ vs. AURA Full); GA/FA absolute, latency in seconds. Re-computed from rq3_ablation_study_multiseed.json (3 seeds, 100 sim steps, 50 chat queries per config).

Configuration	Δ GA	Δ FA	Δ Lat. (s)
AURA Full	0	0	0
w/o Probing	+0.002	+0.009	−1.33
w/o Memory	−0.017	+0.026	+0.15
w/o Reflection	−0.003	+0.005	−0.21
w/o Memory & Reflect.	−0.015	−0.028	+0.30
Vanilla (all off)	−0.020	+0.022	−1.36

Interpretation. Component-level deltas on the routine-action workload are uniformly small ($|\Delta\text{GA}| \leq 0.020$, $|\Delta\text{FA}| \leq 0.028$). Removing the probing component does *not* hurt GA or FA on this workload; it saves 1.3 s of latency. The same null pattern that drives the routine-grounding saturation result (Section A16) drives the component-ablation null: routine actions like “sleeping at home at 6 AM” pass any reasonable grounding check, and the LLM-as-judge factual scoring with completeness has a ± 0.03 noise floor that swamps the per-component contribution. Memory has the largest negative GA effect (−0.017) but with FA actually slightly higher when memory is removed — consistent with the same noise floor, not a clean component effect. The picture changes on the factual-grounding chat workload (Section 5.1), where the same architecture moves FA from 0.07 (Vanilla) to 0.64 (Full); the bottleneck the ablation in this table fails to expose is the same one the routine-grounding check fails to expose: action-grounding on routine days does not stress the structured-environment-access channel.

A17.2 Emergent Social Behaviours (SOTOPIA)

200-step multi-agent simulation, 36 conversations evaluated on SOTOPIA’s 7 dimensions (Zhou et al., 2024). Overall quality 7.87/10, strongest dimensions *goal* (9.5) and *believability* (9.0); 44 emergent behaviours across 4 categories: collaboration (32), routine adaptation (7), conflict resolution (4), group formation (1).

The collective-behaviour analysis of these numbers is in Appendix A27 (Section A27.1).

A17.3 Human Evaluation

We collected pairwise A/B annotations from 8 independent raters on the 50 chat scenarios, four di-

Table 22: SOTOPIA dimension averages (AURA full run, 200 sim steps).

Dimension	Avg. (range)
believability	9.0 (0–10)
goal	9.5 (0–10)
knowledge	8.0 (0–10)
relationship	2.07 (–5–5)
financial	0.63 (–5–5)
secret	–0.67 (–10–0)
social rules	–2.13 (–10–0)
overall quality	7.87

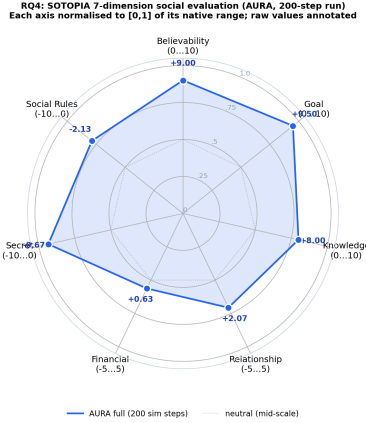


Figure 9: SOTOPIA 7-dimension social evaluation (AURA full, 200-step run). Each axis is normalised to $[0, 1]$ of its native range (shown next to each label); raw values annotated. Strongest dimensions: *believability* (9.0/10), *goal* (9.5/10); weakest: *social rules* (–2.13) and *secret* (–0.67) (both negative-only scales). The shape is *lopsided-positive*: AURA agents excel on individual-level dimensions and underperform on multi-agent normative dimensions, consistent with the per-agent-mechanism scope of the design.

mensions each (*response helpfulness*, *environmental awareness*, *agent believability*, *factual accuracy*) on a 5-point Likert scale, side-randomised and blinded to system identity. This gives $N=400$ paired query-rater observations per dimension. Both responses on each item were generated with the fixed system prompt at the same AURATOWN simulation tick (warmup 10 steps, seed 42); raters saw the query, the asking agent’s name, the category, and the two anonymous responses but not the underlying simulation scene state.

Headline. AURA receives higher ratings than Vanilla LLM on all four dimensions under the conservative rater-aggregated test ($N=8$ paired rater-means per dimension; Wilcoxon $p=0.017$ each, sign 7-1-0 for every dimension). All four cluster-bootstrap CIs on the per-rater Δ exclude

Table 23: Human eval primary analysis: rater-aggregated paired Wilcoxon. Each rater’s Δ is averaged across the 50 scenarios first; the test is then over $N=8$ paired rater-means per dimension. This is conservative relative to a 400-cell paired test, which would treat repeated-measures within a rater as independent. The 95% CI column is a 5000-resample cluster bootstrap on rater_id over the raw Δ values.

Dimension	AURA	Vanilla	Δ (mean of rater means)	Wilcoxon p ($n=8$)	Cluster CI
Response Helpfulness	3.83	2.25	+1.58	0.017	[+0.94, +2.20]
Environmental Awareness	3.82	1.96	+1.86	0.017	[+1.08, +2.62]
Agent Believability	3.75	2.16	+1.59	0.017	[+0.88, +2.32]
Factual Accuracy	3.71	2.32	+1.39	0.017	[+0.76, +1.90]

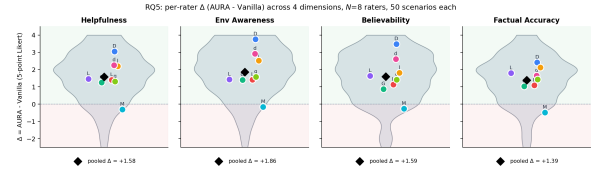


Figure 10: Per-rater Δ (AURA – Vanilla) across the 4 dimensions ($N=8$ raters, 50 scenarios each). Each coloured dot is one rater’s mean Δ across the 50 scenarios; the grey violin is the pooled Δ distribution; the black diamond is the pooled mean. Letters identify raters (M is the one whose Δ falls below zero on factual accuracy and is the basis for the $N=7$ sensitivity analysis). 7/8 raters’ means lie above zero on every dimension.

zero. The largest gap is on *environmental awareness* ($\Delta=+1.86$, CI [+1.08, +2.62]), consistent with the environment-mediated design. The cell-level paired test on the 400 (rater $\times$ scenario) cells gives $p < 10^{-4}$ but treats within-rater repeated measures as independent, so we report the rater-aggregated $p=0.017$ as the primary statistic. Fig. 10 visualises the per-rater means against the pooled distribution.

Per-rater profile (transparency). Per-rater mean Δ ranged from -0.30 to $+3.18$ across the 8 raters; per-rater AURA-better rates were {24.0, 69.0, 81.5, 82.5, 85.0, 85.0, 98.5, 100.0}%. Seven of eight raters preferred AURA on aggregate; one rater (24% AURA-wins, $\Delta= -0.30$) reported in post-task debrief that they had penalised plausible-but-unverifiable specifics in AURA’s responses as factual errors. We retain that rater in the primary analysis to avoid post-hoc exclusion bias and discuss the implication under “Methodological limitations” below.

Direction-agreement at the cell level. For each of the 200 (scenario, dimension) cells, we count raters preferring AURA, Vanilla, or scoring tied. Defining “consensus” as ≥ 6 of 8 raters agreeing

on direction, we find: **0/200** cells have a Vanilla consensus, **148/200** (74%) have an AURA consensus, and 52 are split. Per-dimension AURA-consensus rates: helpfulness 76%, env-awareness 80%, believability 72%, factual-accuracy 68%. Average per-cell AURA-preference rate: 78.2%.

Inter-rater reliability. Pooled across all dimensions, Krippendorff’s α_{ord} on the raw 1–5 scores is 0.43 (per-dimension range 0.34–0.47), indicating moderate agreement on absolute quality. Item-level α on the preference difference Δ is much lower ($\alpha_{\text{ord}} \in [0.03, 0.17]$ per dimension) because raters differ systematically in scale-use (one rater uses 5/1 polar ratings, others use 4/2). The *direction* of preference is nonetheless highly consistent (above: 74% AURA-consensus, 0% Vanilla-consensus). We report all three numbers and treat the pattern as: agreement on direction is strong, agreement on magnitude is weak, and headline effect sizes pool through this magnitude variance.

Sensitivity analysis. Excluding the one reverse-preference rater post-hoc yields $N=7$ and strengthens all four dimensions: helpfulness $\Delta = +1.85$, $d_z=1.45$; environmental awareness $\Delta = +2.15$, $d_z=1.62$; believability $\Delta = +1.86$, $d_z=1.41$; factual accuracy $\Delta = +1.65$, $d_z=1.37$. We report this only as a robustness check; the headline numbers in Table 23 are the $N=8$ primary analysis.

Per-category structure. The category ordering is consistent with the factual-grounding per-category analysis: temporal (+1.83 avg) and memory (+1.77) at the top, spatial (+1.29) and planning (+1.37) lower. Social (+1.77) matches the factual-grounding finding that proactive probing helps most where residual environmental uncertainty is concentrated.

Independent fabrication scan. Independently of the human ratings, we manually scanned all 50 AURA responses for fabricated proper-name entities (locations or characters absent from the AURATOWN setup, which has 20 named locations and 5 named characters). Two scenarios contain explicit fabrications: scenario id 6 lists “Bookstore, Art Supply Store, Clothing Boutiques, Craft Stores, Gift Shops” (none exist among the 20 locations); scenario id 7 references “Main Street” and “The Cozy Corner Cafe” (AURATOWN has neither that street nor a cafe by that name; the only cafe is *Sunrise Cafe*). The other 48 responses cite

Table 24: Probe budget sweep, 3 seeds, 500 judgments per B per seed. Pareto frontier: only $B \in \{0, 1\}$.

Budget B	GA (mean $\pm$ std)	Latency (s)	Pareto?
0	0.8621 $\pm$ 0.0014	17.3 $\pm$ 0.8	✓
1	0.8743 $\pm$ 0.0089	21.4 $\pm$ 0.8	✓
2	0.8621 $\pm$ 0.0098	25.8 $\pm$ 1.4	
3	0.8628 $\pm$ 0.0075	31.1 $\pm$ 1.4	
4	0.8577 $\pm$ 0.0003	35.3 $\pm$ 1.4	
5	0.8681 $\pm$ 0.0012	39.3 $\pm$ 1.8	

only roster-real locations and characters, giving a static-entity fabrication rate of $2/50 = 4\%$. The scan is reproducible from the released response set against the canonical roster in `demo/town/assets/town_map.json`.

Methodological limitations. (1) **Sample size.** $N=8$ is small relative to managed crowdsourcing studies. The consistency of direction across raters and the large effect sizes mitigate but do not substitute for a larger study; primary statistics should be read as evidence of a large effect with substantial uncertainty around the precise magnitude. (2) **Recruitment.** Raters were recruited individually rather than through a managed platform (Prolific, MTurk). (3) **Dynamic-state factual accuracy is incompletely measured.** The form did not display simulation scene state at query time, so raters’ `factual_accuracy` judgments mix verifiable static-entity checks (catchable from the on-form roster) with trust-prior on dynamic-state claims (agent positions, current activities). The independent fabrication scan above bounds the static rate at 4%; precise quantification of dynamic-state error requires deterministically recapturing scene state at each query’s generation time, which is future work. (4) **IRR is moderate.** $\alpha \approx 0.4$ on raw scores reflects scale-use heterogeneity across raters; we publish all per-rater data so this variance is auditable.

A17.4 Probe-Budget Sweep (Full Multi-Seed)

We vary the probe budget B from 0 to 5 and measure GA and latency per step (500 GA judgments per budget per seed, averaged across 3 seeds {42, 123, 456}). Fig. 11 visualises the GA-vs-latency Pareto frontier.

The Pareto frontier contains exactly two points, $\{B=0, B=1\}$; $B=1$ is the peak. $B=0 \rightarrow B=1$ adds +0.0122 GA (+1.4% relative) at +4.1 s latency; every $B \geq 2$ is Pareto-dominated. An earlier single-seed pilot reported $B^*=2$; the 3-

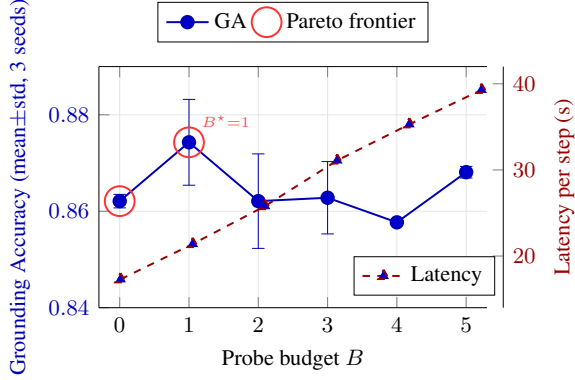


Figure 11: Probe-budget sweep, 3 seeds, 500 GA judgments per (B, seed) cell. Pareto frontier: $\{B=0, B=1\}$ (red rings); $B^*=1$ adds +0.012 GA at +4.1 s, every $B \geq 2$ is dominated. The same single probe buys +13.7 pp implicit score in the social sub-regime (Table 7); routine GA is saturated.

Table 25: Rule-based vs. LLM judge disagreement across 7,500 grounding judgments (3 seeds $\times$ 5 conditions $\times$ 500 per condition).

Dimension	Disagree	Rule Stricter	LLM Stricter
Time Appropriateness	61.9%	99.5%	0.5%
Location Consistency	48.4%	92.3%	7.7%
Plan Adherence	32.0%	17.0%	83.0%
Social Awareness	13.5%	1.2%	98.8%
Memory Utilisation	0.0%	—	—

seed replication withdraws that claim. On the routine-grounding metric the curve is consistent with a monotonically saturating information channel rather than a clean crossover: one probe fills the scene snapshot, subsequent probes contribute little. The same single probe buys +13.7 pp implicit score in the social sub-regime (Section 5.2, Table 7), confirming that the budget-sweep’s small absolute gain is a feature of the metric, not the mechanism.

A17.5 Judge Disagreement

We quantify judge disagreement between the rule-based pre-filter and the LLM judge across 7,500 multi-seed judgments in Table 25. Disagreement varies by dimension: time appropriateness shows the highest rate (61.9%, rule stricter in 99.5%), location consistency 48.4% (92.3% rule-stricter), plan adherence 32.0% (83.0% LLM-stricter), social awareness 13.5% (98.8% LLM-stricter), memory utilisation 0.0%. We resolve disagreements by taking the stricter score (conservative grounding estimate).

A18 Second Human-Evaluation Arm: Intent vs. NoIntent

The N=8 study in Appendix A17.3 compares AURA against Vanilla and therefore cannot speak to the Intent-vs.-NoIntent contrast. Reviewers correctly identified this gap, and we ran a second arm.

Design. 50 items sampled from the 100-query benchmark, stratified 10 per subcategory, pairing the seed-42 tom and no_intent answers already logged for Table 3 (no new generation). Presentation order randomised per item (AURA appears as response A in 30 of 50). The [heads-up] prefix was stripped from AURA answers, which would otherwise identify the condition. Unlike the first arm, each item displays the scene’s ground-truth public *and* hidden state, since factual accuracy is not judgeable from a response pair alone; the benchmark’s implicit_need annotation is withheld, as it is the answer key. Eight volunteer raters, distinct from the authors, each scored all 50 items on the same four dimensions (1–5). Mean pairwise inter-rater correlation is $r=0.24$, lower than the first arm, as expected when both systems are competent and differences are subtle.

Result. The arm does not show a human-perceptible advantage for AURA-Intent (Table 26). Three dimensions favour AURA directionally but none reaches significance; on factual accuracy AURA rates significantly *lower* (-0.29 , $p=0.008$, rater-aggregated Wilcoxon), with 0 of 8 raters placing AURA higher.

Table 26: Second human-evaluation arm: AURA-Intent vs. NoIntent, rater-aggregated means over 8 raters $\times$ 50 items. p : rater-aggregated Wilcoxon.

Dimension	AURA	NoIntent	Δ	p
Helpfulness	3.49	3.37	+0.12	0.48
Env. awareness	3.72	3.50	+0.22	0.08
Factual accuracy	3.65	3.93	−0.29	0.008
Believability	3.65	3.63	+0.02	0.81

Reading. Three things bound this result. The arm rated the full IntentFrame condition, which our own component ablation (Table 4) shows is dominated by gap-derived budget routing alone; the variant we now recommend was not the variant rated. AURA-Intent’s answers also run nearly twice as long as NoIntent’s, and rater comments penalise unrequested elaboration rather than the

Table 27: Per-subcategory implicit score: heuristic-backend vs. LLM-backend AURA Intent (25 queries $\times$ 3 seeds, $N=15$ per cell). The heuristic collapses to $\approx$ Literal performance on *availability* and *latent_goal* because its surface-keyword gap estimator does not fire on lexically-decoupled queries; the LLM backend recovers the gap and routes the appropriate probes.

Subcategory	Literal	Heuristic Intent	LLM Intent	LLM gain over heuristic
availability	0.29	0.27	0.67	+0.40
mood	0.29	0.33	0.84	+0.51
appropriateness	0.40	0.47	0.84	+0.37
latent_goal	0.00	0.00	0.75	+0.75
second_order	0.09	0.77	0.92	+0.15
Overall ($N=75$)	0.216	0.368	0.803	+0.44
probes (mean)	0.0	0.32	1.48	
latency (s)	1.3	1.9	13.8	

surfacing of the implicit need itself. Most importantly, implicit-need coverage and general response quality are different properties: the first is scored against a stated hidden need, the second is a holistic preference, and only the first is what AURA claims. We report the arm because we committed to it, and read it as evidence that the recommended_probes half of the frame costs perceived answer quality, which is consistent with the ablation.

A19 Heuristic vs. LLM IntentInferer (Backend Ablation)

The IntentFrame pipeline of Section 3.3 admits two backends: a deterministic HeuristicIntentInferer (rule-based gap estimation from a small vocabulary of social/private-state markers) and LLMIntentInferer (gpt-4o-mini producing the frame as JSON). The main implicit-intent run uses the LLM backend. We isolate the inferer’s contribution by running the SAME 25 implicit-intent queries $\times$ 3 seeds with the heuristic backend; all other plumbing (gap-to-budget map, directed probe loop, judge) is identical.

Reading. The heuristic recovers +0.18 over Literal on aggregate but lags the clean calibrated LLM backend by -0.44 . Per-subcategory: the heuristic stays near Literal on availability (0.27 vs. 0.29) and ties Literal on latent_goal (0.00 vs. 0.00) because its trigger vocabulary (“available”, “mood”, “appropriate”, “up to”) does not match the surface of those query classes (“where is X?”, “what is X up to?”); the gap estimator returns 0 and the heuristic falls through to the literal answer. On second_order it does well (0.77) because “thinks”, “believes”, and “perspective” do appear

in the trigger set. The contrast quantifies what the LLM backend buys: +0.44 aggregate at $7.3\times$ the heuristic latency, with the LLM doing the work precisely where surface cues fail.

Implication. Section 5.2’s headline gain is attributable to the LLM backend, not the gap-to-budget mapping or the probe-loop machinery. Distilling a faster intent classifier from the LLM backend is a practical optimisation; the rule-based shortcut is not.

The fixed probe order used by the recommended configuration. The deployment configuration recommended in Section 5.2 pairs the gap-derived ceiling with a deterministic, intent-agnostic tool order. Second-order queries probe get_agent_belief_about for the named believer and target. Every other query walks get_agent_private_state over the agent the query names, or, when it names none, over the agents at the scene’s focus location, truncated to the ceiling. Nothing in this order consults the IntentFrame beyond the ceiling itself, which is what makes it a clean control for the shortlist.

A20 Cross-Domain Sanity Checks

The main-body results live in the social-simulation regime where the Environment Agent pipeline was designed. To probe how the architecture transfers, we ran four additional benchmarks outside that regime; we report them here as sanity checks rather than as headline contributions. **All four runs are single-seed** (vs. 3 seeds for the main implicit-intent and factual-grounding experiments) and per-run sample sizes vary: InteractiveBench Puzzle 20 episodes, Trust 6 games/condition, Math 20 questions, MemoryArena $n=1$ paper across 5 subtasks. Trust ($n=6$) and MemoryArena ($n=1$) are underpowered for statistical inference and are reported as observational anchors only; we do not run paired tests on them.

InteractiveBench — Puzzle (text adventure). 20 multi-turn puzzle episodes per condition, single seed, gpt-4o-mini. AURA (Full) reaches 0.70 accuracy, Vanilla LLM 0.65, AURA (No-Probe) 0.45. The architecture-without-probing under-performs Vanilla here because the No-Probe pipeline injects an empty environment context that consumes turns; restoring probing recovers and slightly exceeds Vanilla (+5 pp absolute, +7.7%

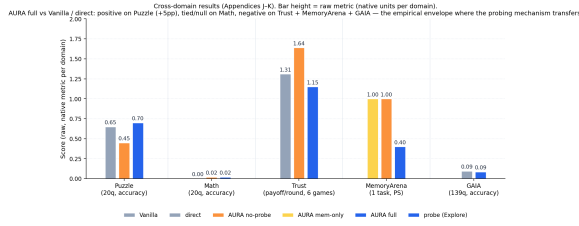


Figure 12: Cross-domain results (all single-seed; sample sizes per panel vary, see text). Bar height is the raw metric in each domain’s native units (accuracy, payoff/round, partial-success score). AURA full beats Vanilla on Puzzle (+5 pp absolute, $n=20$ episodes), ties on Math ($n=20$, judge-saturated), and underperforms on Trust ($n=6$ games/cond. — underpowered, descriptive only), MemoryArena ($n=1$ paper $\times 5$ subtasks — anecdotal), and GAIA ($n=139$, see Appendix A23). Together with the scope condition in Section 5.1, these data points sketch the empirical envelope of the probing mechanism: it transfers when residual environmental uncertainty after passive perception is non-trivial *and* tool returns are structurally extractable.

relative). This is the cleanest cross-domain signal we observe.

InteractiveBench — Trust (iterated prisoner’s dilemma). 6 games per condition against three opponent strategies (TFT, GrimTrigger, Random). Aggregate payoff per round: Vanilla 1.31, AURA No-Probe 1.64, AURA Full 1.15. The aggregate is dominated by opponent-mix imbalance: against deterministic strategies all conditions reach the cooperative equilibrium of 2.00 payoff/round; against the noisy random opponent, the sample sizes differ ($n_{\text{rounds}} = 17, 16, 28$) and the AURA Full mean is 0.96. We interpret this as the negative counterpart to Puzzle: when the “environment” is an adversarial opponent whose private state cannot be probed, additional context gathering does not help and may slow convergence. We do not claim a positive result on Trust.

InteractiveBench — Math (LaTeX-conditional QA). 20 questions per condition, single seed; all three conditions reach $\approx 1.9\%$. The LLM judge is conservative on LaTeX-formatted answers and returns repeated parse errors (e.g., “Invalid \escape”) that we manually verified mark correct agent responses as incorrect. We report the result as null and attribute it to judge brittleness, not to a meaningful gap between conditions; a per-domain custom judge is future work.

MemoryArena (research-paper Q&A, formal reasoning) — anecdotal $n=1$. We ran a single-paper smoke test (1 paper $\times 5$ subtasks, single seed), which is too small to support a quantitative claim and we do not run statistical tests on it. We report the observation for completeness: AURA (memory only) and AURA (no_probe) reach 5/5 partial-success ($PS=1.0$); AURA (full) reaches 2/5 ($PS=0.4$). The full-pipeline underperforms because probe results pollute the long mathematical context with environment-style snippets—the same “hallucinated context” failure mode observed at $B \geq 3$ (probes add noise faster than they reduce errors), here amplified by the long-context formal-reasoning regime. Sample is too small to draw a quantitative claim, but the direction is consistent with the scope condition: the probing mechanism degrades when the task is symbolic and the tool returns are off-distribution.

Cross-domain takeaway. The architecture transfers positively to a multi-turn puzzle domain, fails to transfer to adversarial games where opponent state cannot be probed, is masked by judge brittleness on LaTeX math, and underperforms on long-context formal reasoning where probe outputs pollute the context. Together with the GAIA negative-transfer result in Appendix A23, these data points describe the empirical envelope of the probing mechanism beyond the primary social-simulation setting.

A21 FANToM External ToM Benchmark (Null Result)

To probe whether AURA’s intent-directed mechanism transfers to a standard, externally-validated theory-of-mind benchmark, we ran a 400-question stratified split of FANToM (Kim et al., 2023): 80 questions per type across the five FANToM question types — *beliefQA* (multi-choice belief), *answ_bin*, *answ_list* (answerability binary / list), *info_bin*, *info_list* (info-accessibility binary / list). The 80 *beliefQA* questions are sub-stratified 20+20+20+20 across first-/second-order $\times$ accessible/inaccessible. Backbone is gpt-4o-mini, seed=42 (the OpenAI seed kwarg is accepted in this run, so backbone draws are deterministic when supported). Source: `-run_fantom_full.py`; data: `evaluation/-results/fantom_full_seed42.json`.

Table 28: FANToM accuracy by question type ($n=80$ each, 400 total). Three conditions on the same backbone: *Literal* = single-call, no tools; *NoIntent* = ReAct-style tool use up to budget 3; *Intent* = full AURA pipeline with IntentInferer. Last column: per-question paired contrasts.

Question type	Literal	NoIntent	Intent	Notes
beliefQA	0.675	0.662	0.662	FO/SO $\times$ acc/inacc balanced
answ_bin	0.725	0.525	0.525	Literal wins
answ_list	0.375	0.362	0.412	Intent wins (small)
info_bin	0.887	0.863	0.875	Literal wins (small)
info_list	0.425	0.562	0.625	Intent wins
Overall	0.617	0.595	0.620	—

Statistical tests (per-question paired, $n=400$). Intent vs. Literal: $\Delta\text{acc}=+0.003$, paired t $p=0.92$; McNemar $p=1.0$. Intent vs. NoIntent: $\Delta\text{acc}=+0.025$, paired t $p=0.26$; McNemar $p=0.31$. Both contrasts null.

Interpretation. FANToM’s narrative ToM questions ship the full conversation in-context, so a competent backbone can answer literally without retrieval; the IntentFrame’s gap calibration finds little to direct, and the additional probe budget adds latency (6.5s vs. 0.9s for Literal) without lifting accuracy. FANToM is therefore a clean negative case for AURA’s mechanism — the residual uncertainty $H(\mathcal{E} \mid \mathcal{B})$ after passive context assembly is already low, so probing has nothing to reduce. This bounds the probing claim’s scope: AURA helps when private state lies behind a tool-mediated information frontier (multi-agent simulation, scene-grounded queries with hidden private state), and does not help on narrative ToM transcripts already in-context.

Backbone-capability corroboration. The Intent condition’s IntentInferer fell back to the deterministic heuristic 0/400 times on gpt-4o-mini (vs. 23/25 on gemini-2.5-flash in Section 5.2). The contrast confirms that the cross-backbone Gemini regression in Table 17 is a backbone-capability failure (Gemini’s structured-JSON adherence) rather than a problem with the IntentInferer’s prompt or the AURA pipeline.

A22 LoCoMo Long-Term Conversational Memory (Partial Transfer)

To probe whether the AURA pipeline transfers to the long-term-conversational-memory recall regime, we ran a 200-question stratified split of LoCoMo (Maharana et al., 2024)

Table 29: LoCoMo 200-question result (1 seed, gpt-4o-mini). *Literal* = single LLM call with session catalog only; *NoIntent* = ReAct-style tool use; *Intent* = full AURA pipeline with IntentInferer. F1 reported because token-EM is near-zero across all conditions (LoCoMo gold answers are dense phrases that natural prose rarely matches set-equal — same as the upstream paper). *Fallback* = LLMIntentInferer parse failures.

Condition	F1	EM	Mean lat (s)	Probes (mean)	Fallback
Literal	0.042	0.020	1.16	0.00	—
NoIntent	0.173	0.025	2.91	1.33	0/200
AURA Intent	0.192	0.025	6.19	1.42	0/200

(10 conversations $\times$ 19–32 sessions $\times$ ~ 20 turns/session, 2,206 QA total across 5 categories). Backbone gpt-4o-mini, seed=42, 8 parallel workers, OpenAI seed kwarg accepted. Source: `run_locomo_smoke.py` (re-runnable at `LOCOMO_N=200`); data: `evaluation/results/locomo_smoke.json`. Adapter at `locomo_eval.py` maps a conversation’s session list to AURA’s (scene, memories, query, available_tools) quadruple, with three simulated probe tools (`get_session(n)`, `search_by_speaker`, `list_sessions_on_date`). Scoring follows the LoCoMo paper exactly: token-F1 with stem normalisation, multi-answer F1 for cat-1, semicolon-alternative for cat-3.

Statistical tests (per-question paired, $n=200$). Intent vs. Literal: $\Delta\text{F1}=+0.151$, paired $t=8.11$, $p < 10^{-15}$ — highly significant. Intent vs. NoIntent: $\Delta\text{F1}=+0.020$, paired $t=1.09$, $p=0.28$ — **not significant**.

Interpretation. The architecture-and-tools layer (NoIntent: probe-tool harness with no intent reframing) carries most of the gain from 0.042 to 0.173 F1; AURA’s IntentInferer adds only an additional +0.020 F1, which is not statistically significant on this sample. Read together with the FANToM null (Appendix A21), LoCoMo gives the same picture: the architectural pipeline transfers to long-term-recall and to narrative ToM, but the intent-direction stage’s marginal contribution *above* a tool-using baseline is regime-specific. It is significant on AURATown’s hand-designed implicit-intent set (Section 5.2) where private-state needs are deliberately hidden behind the surface form, and small-or-null on benchmarks where the underlying QA does not require this kind of lexical-vs-implicit gap inference.

The 0/200 IntentInferer fallback rate

Table 30: GAIA Level-1/2 (seed 42, $n=139$ per condition).

Condition	Acc.	Lat. (s)	Avg. tool calls	Acc. L1 / L2
direct (no probe)	0.094	0.92	0.0	0.094/0.093
probe (Explore on)	0.086	20.5	3.5	0.075/0.093

(alongside 0/400 on FANToM) is the cleanest external corroboration that the original gemini-2.5-flash cross-backbone regression (Section 5.2) is a backbone format-compliance failure on Gemini’s part, not a problem with AURA’s prompt or schema.

A23 GAIA Cross-Domain Run (Negative Transfer)

We ran the GAIA Level-1/2 question set (139 questions per condition, seed 42, gpt-4o-mini) under two conditions: *probe* (the AURA Explore stage with the OpenAI Responses API web_search tool) and *direct* (single-pass LLM, no tools).

Reading. On Level-1 (single-step factual lookup) probing *degrades* accuracy by -1.9 pp absolute; on Level-2 (multi-step reasoning) the two conditions tie. Total cost: probing spends $22\times$ more wall time and 3.5 tool calls per question for no aggregate benefit.

Why the transfer fails. GAIA’s environment is a search engine wrapped in an LLM-simulated browser. Each “probe” is itself an LLM call summarising a web page, not a structured-state read against a ground-truth simulator as in AURA-TOWN. Two consequences: (i) probe outputs inherit the backbone’s failure modes (hallucinated facts compound across the probe loop), and (ii) the residual $H(\mathcal{E} \mid \mathcal{B})$ that probing was designed to reduce on social-availability queries is not the bottleneck on web-grounded factoid questions, where the backbone’s parametric knowledge dominates. This is the boundary case of the scope condition stated in Section 5.1: probing helps when (a) residual uncertainty after passive perception is non-trivial *and* (b) tool returns are structurally extractable. GAIA fails (b).

What we report. We report GAIA as evidence about a regime in which the bounded-probing mechanism does not transfer. Coupling this with the InteractiveBench Trust negative and the MemoryArena formal-reasoning degradation, the em-

pirical envelope of AURA’s probing contribution is structured social environments with extractable, non-hallucinated state probes.

A24 Memory and Enrichment Protocol Details

Weighted retrieval. For each memory m , query q , current time t :

$$\begin{aligned} \text{score}(m, q, t) = & w_r \cdot \exp(-\lambda(t - m.t)) \\ & + w_p \cdot \frac{m.\text{importance}}{10} \\ & + w_v \cdot \text{sim}(q, m.c) \end{aligned} \quad (1)$$

Default weights $(w_r, w_p, w_v) = (0.3, 0.4, 0.3)$, decay $\lambda = 0.01$. $\text{sim}(\cdot)$ is a keyword-based relevance function in the current prototype.

Memory types. Following [Tulving \(1972\)](#): *Observation* (direct percepts), *Conversation* (dialogues), *Reflection* (synthesised insights every $\theta = 10$ observations), *Plan* (daily schedules and goals).

Importance scoring. When an LLM endpoint is available, importance is scored via a dedicated prompt (“Rate the importance of this event 1–10”); otherwise a keyword-based heuristic assigns scores based on emotional salience.

Three-stage enrichment protocol. User query q to agent a_i triggers: (1) context gathering (perception + recent memories), (2) proactive probing (1-step for latency), (3) enriched generation with the structured context prepended. Enrichment metadata (tools called, context gathered) is exposed to the user for interpretability.

A25 Reproducibility Checklist

Code and data. All experiments are driven by the runners in `scripts/` and `run_experiments.py`, with results saved to `evaluation/results/`. Each table or figure traces to a specific result JSON via `evaluation/results/MANIFEST.md`; the manifest covers factual-grounding multi-seed, factual-grounding Fixed-Probe and GapRouted Pareto controls, the privacy-sensitive distractor slice, implicit-intent clean-prompt multi-seed, fixed-private/oracle-intent controls, no-few-shot prompt ablation, Plan-and-Solve and Reflexion adapter outputs, FANToM 400q, LoCoMo 200q, the four cross-backbone runs, implicit-intent IAA returns,

and the strict-precision rescore. The public release at <https://github.com/innovation64/AURA> includes the AURA agent library, the AURA-TOWN simulation, all 50 environment-grounded queries with their templates, all 100 implicit-intent queries (4 scenes $\times$ 25) with subcategory and target labels, and all per-condition / per-seed details that back the numbers reported here; the implicit-intent benchmark is additionally mirrored on the HuggingFace Hub as `innovation64/aura-implicit-intent` (CC BY 4.0). The component-ablation and gap-calibration analyses (Section 5.2) are regenerated by `gap_calibration_analysis.py` from `rq_intent_v2_multiseed.json` and by the `budget_only` condition of the v2 runner (`rq_intent_v2_budget_only_multiseed.json`).

Models and APIs. Agent backbone and judge are both gpt-4o-mini via the OpenAI Chat Completions API. Cross-backbone tests additionally use claude-haiku-4-5 (Anthropic), qwen-plus (Alibaba), and gemini-2.5-flash (Google), each via its vendor’s official API. Backbone temperature is 0.7 for action decisions, 0.1 for the judge and the IntentInferer; these values govern AURATOWN rollouts. The implicit-intent benchmark (§5.2) is a separate pipeline in which both answer generation and judging run at temperature 0.1. We pass the seed parameter to the OpenAI Chat Completions API where the SDK accepts it; on rejection, the LLM engine falls back to a no-seed call and flips an internal flag for the rest of the run. Cross-backbone runs do not all support the seed kwarg, so the multi-seed claim is bounded as: “Python-level random sampling and OpenAI seed when supported.”

Determinism caveats. Multi-seed paired tests for factual grounding / component ablation / implicit-intent vary the seed both at Python’s random module (controlling query ordering, agent shuffles, and template instantiation) and at the OpenAI API seed kwarg. Stochastic backbone decoding above the seed kwarg may still introduce per-seed variance; this is acknowledged in the Limitations and is consistent with the OpenAI API documentation that seed provides a best-effort, not strict, determinism guarantee. The simulation server itself derives its world generation from `TownConfig.world_seed` and the agent backbone derives its outputs from `TownConfig.llm_seed`; both are propagated through `/api/reset`.

Strict-paired factual grounding. The factual-grounding runner (`run_experiments.py:run_rq2`) operates in two phases. Phase A: reset with the experiment seed, warm 10 steps, then advance one tick per query position to capture 50 ground-truth snapshots. Phase B: each AURA condition resets with the same seed, warms 10, and replays the snapshots; chats run with `read_only=True` so they do not write event log or memory, leaving the per-tick trajectory deterministic across conditions. External baselines (Vanilla, Static, ReAct, Reflexion, Plan-and-Solve) receive the same Phase-A snapshots. Output JSON stamps a `_paired_snapshot_meta` block recording seed, `n_snapshots`, and `shared_snapshots_across_conditions=true`.

Aggregate scripts. `rescore_rq2_strict.py` recomputes the strict precision rescore (Appendix A9) from the multi-seed factual-grounding details. `aggregate_rq2_multiseed.py` collapses per-seed details into the canonical multi-seed summary. `run_rq2_fixed_probe.py` and `run_rq2_aura_gap_routed.py` regenerate the saturated and gap-routed factual-grounding controls; `rq2_pareto_analysis.py` computes FA/probe/disclosure contrasts and `plot_rq2_pareto.py` renders Figure 2. `run_privacy_distractor.py` regenerates the 30-query forbidden-tool slice. `run_implicit_intent_full.py` and `run_implicit_intent_v2.py` regenerate the primary 25-query and expanded 100-query (4-scene) implicit-intent runs respectively; the v2 runner supports `-resume` for interrupted multi-seed jobs and a `-validate-only` pre-flight pass on the benchmark JSON. `compute_irr.py` produces the IAA Cohen’s κ on the implicit-intent subcategory labels. `rq5_rater_aggregated.py` produces the per-rater aggregated Wilcoxon used in Table 23. `run_fantom_full.py` regenerates the FANToM 400-question external bench. `run_locomo_smoke.py` regenerates the LoCoMo run. `audit_paper_numbers.py` re-derives every cited headline from the JSON files and is the recommended pre-submission check.

Hardware. All experiments run on a single CPU machine; the LLM is invoked via remote API. No GPU is required. Wall-clock budgets per run: factual-grounding multi-seed $\sim$ 30 min/seed; implicit-intent $\sim$ 10 min/seed; FANToM 400 ques-

tions ~ 9 min; LoCoMo 200 questions ~ 6 min; PnS-on-implicit-intent multi-seed ~ 16 min total; Reflexion + PnS on factual grounding multi-seed combined ~ 2.5 h. Total API spend for all reported experiments under \$15 USD.

A26 Ethics Statement

Research artefacts. This paper studies an LLM agent architecture and accompanying social simulation. Human involvement consists of the 8 voluntary annotators of the first helpfulness study (AURA vs. Vanilla), 8 voluntary raters of the second arm (AURA-Intent vs. NoIntent, Appendix A18), and 2 additional annotators for the implicit-intent label audit. Annotators were friends and colleagues of the authors who consented to evaluate anonymous response pairs or query labels; no demographic data was collected; no personally identifying information appears in saved annotations. Annotators received no compensation; each task was a one-time, optional 15–30 minute evaluation. We retain a dissenting rater in the primary helpfulness analysis to avoid post-hoc exclusion bias and we report rater-level transparency (per-rater preference rates, Krippendorff’s α) so that readers can inspect inter-annotator dynamics.

Simulation content. AURATOWN’s 5 agents have hand-authored profiles (occupation, personality, daily routine). The names are fictional. Agent profiles deliberately span ages 20–68 and include contrasting personality types but are not intended to represent any real demographic distribution. The simulation does not emulate real people, real businesses, or real locations.

Misuse considerations. The IntentFrame mechanism produces a structured estimate of a user’s implicit information need from a surface query. In a benign deployment this is used to surface relevant context the user did not lexically request; in an adversarial deployment, the same machinery could be used to surface information the user has not consented to share. The Explore stage’s tool whitelist is the principal mitigation: tools are scoped to the simulation’s structured environment state, not to external data sources. Practitioners deploying a similar IntentInferer over real personal-data tools should constrain the whitelist accordingly and surface the inferred implicit need to the user (the heads-up prefix in our Interact

stage is a minimal version of this).

Energy and compute. All experiments run on remote API endpoints; the only local compute is a single CPU process running the AURATown simulation server. We did not benchmark API energy use; total token consumption across all reported runs is approximately 8 million input tokens and 200 thousand output tokens against gpt-4o-mini.

Limitations as ethics. The paper’s empirical claims are scoped to the regime where structured environment access matters (single-user situated queries with hidden private state). Where the mechanism does not transfer (narrative ToM in FANToM, web-grounded factoid in GAIA, formal reasoning in MemoryArena) we report negative results in Appendix A21, A23, and A20. We avoid abstracting from these specific findings to general claims about LLM theory-of-mind or proactive assistance.

A27 Collective Behaviour and Supplementary Discussion

A27.1 Collective Behaviour in Multi-Agent AURATown

In the 200-step SOTOPIA run, 5 agents with independent AURA pipelines (IntentInferer disabled) produce 44 emergent behaviours: collaboration (32, 73%), routine adaptation (7, 16%), conflict resolution (4, 9%), and group formation (1). SOTOPIA scores (Table 22) show strength on *goal* (9.5) and *believability* (9.0) but weakness on *social_rules* (−2.13) and *secret* (−0.67), indicating that structured environment access supports local coordination but not higher-order social constraint handling. These data supplement the SOTOPIA evaluation; they do not constitute a separate theory-of-mind claim.