

# Observe, Share, or Predict? Disentangling Partner Information in Embodied Multi-Agent Collaboration

**Adrián Szlatinicsán**  
Fondazione Bruno Kessler  
University of Trento

aszlatinicsan@fbk.eu

**Filippo Ziliotto**  
Fondazione Bruno Kessler

fziliotto@fbk.eu

**Bruno Lepri**  
Fondazione Bruno Kessler

lepri@fbk.eu

**Luciano Serafini**  
Fondazione Bruno Kessler

serafini@fbk.eu

**Massimiliano Luca**  
Fondazione Bruno Kessler

mluca@fbk.eu

**Tommaso Campari**  
Fondazione Bruno Kessler

tcampari@fbk.eu

## Abstract

Large language models (LLMs) are increasingly used as high-level controllers for embodied agents, but it remains unclear whether they support collaborative reasoning, such as anticipating a collaborator’s actions in real-time teamwork without dialogue. We ask how three ingredients of partner information, namely observing the partner’s state, exchanging object knowledge, and explicitly predicting the partner’s next action, separately shape task outcomes and coordination when agents can share percepts but never intentions, plans, or task-level messages, and whether the two agents occupy symmetric roles. Using the Habitat-based PARTNR benchmark, we introduce a Decoupled Partial Observability regime that enforces strictly individual perception, and we augment decentralized zero-shot ReAct planners with Other Agent’s State (OAS) observation, object Knowledge Exchange (KE) between co-located agents, and Other Agent’s Next Action (OANA) prediction, evaluated on the full  $2 \times 2 \times 2$  grid with four planners on 1,000 validation tasks. OAS observation primarily improves task quality, reducing extraneous effort for every planner, whereas KE primarily improves execution efficiency, cutting simulation steps by up to 25% with little change in success. The two are largely redundant for strong planners but complementary for the weakest, for which knowledge exchange without the partner state is the worst configuration. OANA prediction lowers the success rate in fifteen of sixteen model  $\times$  context pairs while raising explicit partner mentions to 55–100%, and its accuracy gains lie mainly in the coarse action type, partly because agents recognize the action the partner is already performing. On average, the robot picks up objects earlier and usually finishes its share first, while the simulated human keeps working afterwards, and shared context helps mainly by shortening this period in which only the human is still working. Coordination gains thus come from reacting to shared context, chiefly the partner’s observed state, rather than from explicit anticipation, and partner modeling in this setting is shallow and planner-dependent.

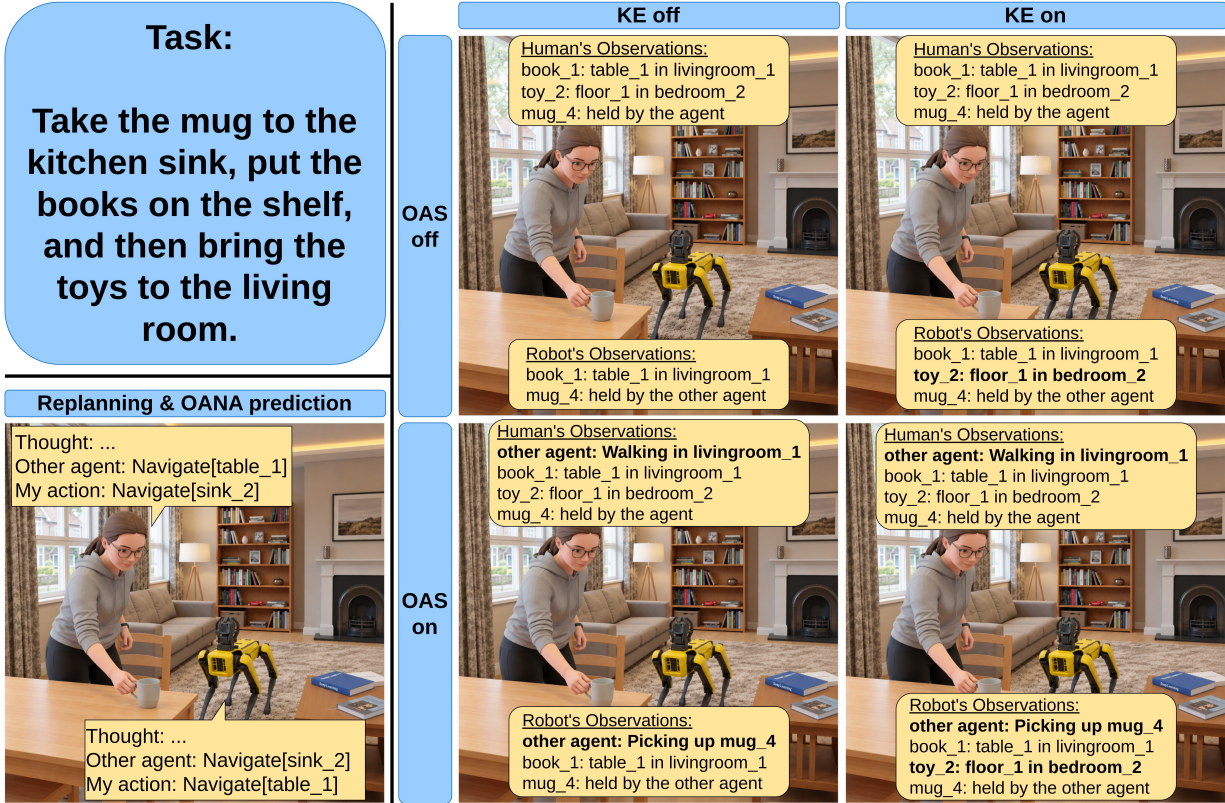


Figure 1: **Observation settings and OANA prediction.** In the object-only baseline (KE and OAS both off) of Decoupled Partial Observability, agents observe only objects and their positions (including whether an agent holds them). With object KE, co-located agents synchronize their accumulated object knowledge. With OAS observation, agents also observe the partner’s state, but only when co-located. With OANA prediction, each replanning step predicts the partner’s next action before the agent selects its own. Text in the figure is simplified relative to the prompts; the background image is AI-generated.

## 1 Introduction

Large language models (LLMs) are increasingly used as high-level controllers for embodied agents that collaborate with each other or with humans in complex, partially observable environments, such as household assistance and human–robot teamwork in realistic homes (Glocker et al., 2025; Li et al., 2026; Chang et al., 2025; Chen et al., 2024; Zhang et al., 2024a). LLM-based multi-agent systems coordinate, deliberate, and negotiate through rich communication channels and plan flexibly on long-horizon embodied tasks (Xi et al., 2025; Glocker et al., 2025; Li et al., 2026; Chen et al., 2024; Zhang et al., 2024a), and a growing literature suggests that LLMs exhibit emergent Theory-of-Mind-like (ToM-like) abilities (Kosinski, 2024; Li et al., 2023; Yang et al., 2025), which would help in decentralized control under partial observability, a hard problem in multi-agent planning and learning (Amato et al., 2019; Omidshafiei et al., 2017; DeWeese & Qu, 2024), although whether such abilities are robust, and whether they transfer from text-based question answering into behavior, remains contested (Shapira et al., 2024; Gu et al., 2026).

However, in decentralized embodied settings, LLM agents still struggle with coordination, frequently exhibiting extraneous behaviors such as redundant task execution, repeated subtasks, and irrelevant actions (Chang et al., 2025; Chen et al., 2024; Li et al., 2025; Zhang et al., 2024a). PARTNR (Chang et al., 2025) shows that access to the partner’s intent, enabled via centralized control, can improve task efficiency, but such benefits are less understood under decentralized control (Chang et al., 2025; Li et al., 2025). Current evaluations also often confound perception, planning, communication, and social or collaborative reasoning, and rarely isolate how a collaborator’s observable state and explicit partner prediction affect real-time coordination

(Chang et al., 2025; Chen et al., 2024; Zhang et al., 2024a). Three conflationations are particularly consequential: (i) what an agent knows about its partner and what it knows about the world its partner acts in are often supplied together as one notion of shared context; (ii) coordination metrics are often aggregated over the team, presupposing that agents running identical planners occupy symmetric roles; and (iii) reacting to a partner’s observable state is rarely separated from anticipating its behavior, so improvements in coordination are readily read as evidence of partner modeling. It therefore remains unclear what partner modeling LLM agents actually perform, and when it improves planning, division of labor, temporal alignment, and task success, particularly when agents can exchange percepts but no intentions or plans (Chang et al., 2025; Li et al., 2023; Zhang et al., 2024a).

To address these gaps, we propose a controlled framework<sup>1</sup> that disentangles observability, partner-state access, object-knowledge sharing, and explicit partner-action prediction. Building on the PARTNR benchmark (Chang et al., 2025), which runs in Habitat 3.0 (Puig et al., 2024), we introduce (i) a **Decoupled Partial Observability** regime that, unlike PARTNR’s partial observability, enforces strictly individual perception for both agents; (ii) a structured **Other Agent’s State (OAS)** observation and (iii) an object **Knowledge Exchange (KE)** mechanism for co-located agents that transmits object positions and states without sender attribution, which we manipulate independently of each other; and finally (iv) explicit **Other Agent’s Next Action (OANA)** prediction within decentralized zero-shot ReAct (Yao et al., 2023) planners, scored both leniently and strictly so that recognizing a partner’s ongoing action is not credited as anticipating it. We evaluate the full  $2 \times 2 \times 2$  grid of these factors with four planners (Llama-3.1-8B, 4-bit Llama-3.1-70B, Gemini 3.1 Flash Lite, and DeepSeek V3.2) on 1,000 validation tasks, add a full-observability control, and report coordination metrics per agent as well as for the team. Figure 1 illustrates the observation settings and OANA prediction. Our main findings are:

- **The two context channels act on different axes** (Section 4.2). OAS observation improves task quality, reducing extraneous effort for every planner, while KE improves execution efficiency, cutting simulation steps by up to 25% with little change in success. They are largely redundant for strong planners but complementary for the weakest, for which object knowledge without partner state is the worst configuration we measure.
- **Explicit prediction lowers success** (Sections 4.2 and 4.5). OANA prediction lowers the success rate in fifteen of sixteen model×context pairs, and in nineteen of twenty once full observability is included, while driving partner mentions to 55–100%.
- **Partner modeling is shallow** (Sections 4.4 and 4.5). Shared context improves predictions of the coarse action type much more than of its arguments. Part of the measured accuracy reflects recognizing the partner’s ongoing action rather than anticipating its next one, and even with complete world knowledge, the Llama planners name their partner’s next action type wrongly more than half the time.
- **The two agents take different roles** (Section 4.3). The robot picks up its first object earlier and usually completes its share of the work first, whereas the simulated human keeps working after the robot has finished and therefore determines when the episode ends. Shared context helps mainly by shortening this final period in which only the human is still working, while the division of labor between the two agents stays nearly the same.

The coordination gains we observe therefore come from agents reacting to shared context, chiefly structured partner state; explicit anticipation of the partner does not reliably add to them. For the debate on Theory of Mind in LLMs, this is a cautionary result: in embodied collaboration, neither coordination gains nor partner-mention frequency nor prediction accuracy that credits a partner’s ongoing action should necessarily be read as evidence of mental-state attribution.

## 2 Related Work

**Implicit Multi-Agent Coordination in Embodied LLMs.** Much of the recent research on LLM-based multi-agent systems focuses on explicit communication, such as dialogue (Zhang et al., 2024b; Estornell et al., 2025; Han et al., 2024; Li et al., 2024; Plaat et al., 2025). While these works establish that LLMs can coordinate through deliberation, they do not fully capture the constraints of embodied settings where

<sup>1</sup>Code and evaluation scripts are available at <https://github.com/szlAdrian/observe-share-or-predict>.

messaging and dialogue are limited or may be unavailable. Social conventions can emerge in LLM populations through repeated interaction alone, without explicit negotiation (Ashery et al., 2025), and LLM-generated narratives can steer cooperation in multi-agent systems (de Curtò & de Zarzà, 2025), although scaling LLM agents to populations raises validity concerns (Taillandier et al., 2025). Communication itself does not guarantee coordination: it can reinforce biases (Buscemi et al., 2026), and collaborative outcomes depend on whether agents reach common ground (Nath et al., 2026). In contrast to dialogue-based systems, we focus on embodied coordination in which agents can exchange percepts but never intentions, plans, or task-level messages, so that partner intent must be inferred from task dynamics and observation rather than declared.

**Embodied Planning and Partial Observability.** In embodied AI, LLMs serve as high-level controllers for task decomposition and coordination across agents in shared environments (Zhang et al., 2024a; White et al., 2025; Wang et al., 2025a; Seo et al., 2026). While LLMs improve flexibility in long-horizon tasks, decentralized collaboration frequently suffers from partial observability and redundant or extraneous effort (White et al., 2025; Chang et al., 2025), and an agent’s world model goes stale when its partner changes the world out of view. Modern frameworks mitigate these problems through modular agent architectures or collaborative belief models (Zhang et al., 2024a; Wang et al., 2025b). However, most evaluations assess collaboration holistically, making it difficult to isolate whether gains stem from perception, planning, or social or collaborative reasoning. Our work differs by assuming oracle action execution and privileged perception to factor out low-level errors, allowing us to directly test whether access to partners’ observable states, exchanging object knowledge, and explicit modeling improve decentralized performance.

**Theory of Mind and Partner Modeling.** A central debate is whether LLM agents possess a functional Theory of Mind (ToM), the ability to represent others’ latent mental states (Li et al., 2023; Gandhi et al., 2023; Street et al., 2025; Yang et al., 2025). While LLMs perform well on text-only ToM benchmarks, whether these abilities are robust and transfer into behavior is contested (Shapira et al., 2024; Gu et al., 2026). We do not probe mental-state attribution directly: our agents are never queried about a partner’s beliefs, and OAS observations deliberately exclude internal state. Instead, we study partner modeling at the level of behavior that ToM is ultimately invoked to explain, anticipating a collaborator’s next action, and ask whether it improves joint behavior. Related work infers a partner’s goals or intentions in order to coordinate (Wang et al., 2022; Trivedi & Huber, 2025) or reasons from a partner’s visual perspective (Patania et al., 2025). Work that equips LLM agents with explicit partner models, such as Hypothetical Minds (Cross et al., 2025), which generates and refines natural-language hypotheses about other agents’ strategies, evaluates the modeling component by ablation under a fixed observation of the other agents. We additionally manipulate that observation: observable partner-state access (OAS) and explicit action prediction (OANA) are varied independently, and predictions are scored both leniently and strictly so that recognizing what a partner is already doing is not credited as anticipating it. This yields a behavioral test of whether explicit partner modeling translates into more efficient and successful collaboration, an embodied counterpart to the gap between explicit ToM inference and its application reported by Gu et al. (2026).

### 3 Method

Our goal is to characterize how two embodied LLM agents coordinate while collaborating on a shared task. We use *collaboration* for the joint pursuit of that task and judge it by team outcomes (Success Rate, Percent Complete, Simulation Steps), and *coordination* for how the agents manage the dependencies between their actions within it (Malone & Crowston, 1994): avoiding duplicated work, aligning in time, and dividing the subtasks. *Partner modeling* refers to what an agent represents about its partner, from reacting to its observed state to anticipating its next action. We ask whether observing the OAS, sharing object knowledge through KE, and predicting the OANA reduce coordination failures such as timing conflicts, race conditions, and extraneous actions; how each affects task outcomes and coordination; and whether the two agents take different roles. Section 3.1 describes the agents and how they plan, Section 3.2 describes what each agent knows about the objects and its partner under each condition, and Section 3.3 describes how agents predict their partner’s next action and how these predictions are scored. Figure 2 illustrates the architecture, adapted from PARTNR (Chang et al., 2025).

### 3.1 Setting and Agents

A simulated human ( $h$ ) and a robot ( $r$ ) work on a shared household task, such as moving plates from a dining table to the kitchen. The tasks come from the PARTNR benchmark (Chang et al., 2025), which runs in Habitat 3.0 (Puig et al., 2024) in multi-room homes from the Habitat Synthetic Scenes Dataset (Khanna et al., 2024). We write  $i \in \{h, r\}$  for an agent and  $j$  for its partner. Control is decentralized: each agent runs its own LLM planner, and the agents send each other no messages; the only channels between them are those of Section 3.2. Both agents know the layout of the house (rooms and furniture) from the start, but have to find the objects.

**World graph.** Each agent stores what it knows in a world graph  $g_{i,t}$ , whose nodes are the house, rooms, furniture, receptacles, objects, and agents, and whose edges are the relations between them (e.g., `cup_1 on table_2`). The index  $t$  counts low-level simulator steps, the unit of the Simulation Steps metric. How  $g_{i,t}$  is updated is the subject of Section 3.2.

**Planning loop.** Each agent is a zero-shot ReAct planner (Yao et al., 2023). Let  $\tau_i^{(k)}$  denote the simulator step at which agent  $i$  replans for the  $k$ -th time; because execution is asynchronous, the replanning steps of the two agents rarely coincide. Agent  $i$  keeps a prompt history  $H_i^{(k)}$  that starts with the instructions and task, the descriptions of its action tools, and the layout. At its  $k$ -th replanning step, the agent receives a textual observation  $o_i^{(k)}$ , a summary of the objects in  $g_{i,\tau_i^{(k)}}$ , and its LLM outputs a reasoning trace  $R_i^{(k)}$  and a high-level action  $a_i^{(k)}$ :

$$(R_i^{(k)}, a_i^{(k)}) \sim \text{LLM}_i(\cdot \mid H_i^{(k-1)} \oplus o_i^{(k)}), \quad H_i^{(k)} = H_i^{(k-1)} \oplus o_i^{(k)} \oplus R_i^{(k)} \oplus a_i^{(k)}, \quad (1)$$

where  $\oplus$  appends text to the prompt. A high-level action has the form `<action_type>[<parameters>]`, e.g., `Navigate[table_0]` or `Explore[bedroom_1]`. Oracle skills execute it as a sequence of low-level simulator steps, concurrently with the partner’s ongoing action, so a single high-level action usually spans many steps  $t$ .

**Action spaces.** We use the PARTNR action tools. The robot acts only through navigation and rearrangement primitives (e.g., `Navigate`, `Pick`, `Place`, `Open`, `Close`), whereas the human additionally has the object-state actions `Clean`, `Fill`, `Pour`, `PowerOn`, and `PowerOff`, so the human must contribute whenever these are needed. This gives the human agent 14 action types and the robot agent 9 (Appendix A.1.1). An episode ends when both agents have called `Done[]`.

### 3.2 Information Conditions

**Decoupled Partial Observability.** In PARTNR’s partial observability setting, the robot also has access to the human’s perception. To remove this asymmetric advantage and keep perception realistic, we introduce *Decoupled Partial Observability*, in which each agent’s world graph is updated only from its own field of view. Let  $p_{i,t}$  be the objects in agent  $i$ ’s field of view at step  $t$ , with their positions and states (including whether an agent holds them), and let  $U(g, p)$  return  $g$  with every object in  $p$  added or moved to its perceived position and state. In the baseline,  $g_{i,t} = U(g_{i,t-1}, p_{i,t})$ . We read  $p_{i,t}$  from the simulator (privileged perception), which removes the noise of learned object detectors and isolates the effects of the manipulations below. Two optional channels change this update, and only while the agents are co-located, that is, in the same room. With OAS observation, the update also uses the partner’s current percepts  $p_{j,t}$ . With object KE, the agents then synchronize: Sync sets every object known to either agent, in both world graphs, to its true position and state in the simulator’s global world graph  $g_t^{\text{global}}$ . Writing  $\tilde{g}_{i,t}$  for the graph after the perception update,

$$\tilde{g}_{i,t} = \begin{cases} U(g_{i,t-1}, p_{i,t} \cup p_{j,t}) & \text{if OAS is enabled and the agents are co-located,} \\ U(g_{i,t-1}, p_{i,t}) & \text{otherwise;} \end{cases} \quad (2)$$

$$g_{i,t} = \begin{cases} \text{Sync}(\tilde{g}_{i,t}, \tilde{g}_{j,t}, g_t^{\text{global}}) & \text{if KE is enabled and the agents are co-located,} \\ \tilde{g}_{i,t} & \text{otherwise.} \end{cases} \quad (3)$$

**Other Agent’s State (OAS) observation.** To mimic real-world co-presence, OAS observation lets an agent see what its partner is doing. While the agents are co-located, the agent observes its partner’s visible

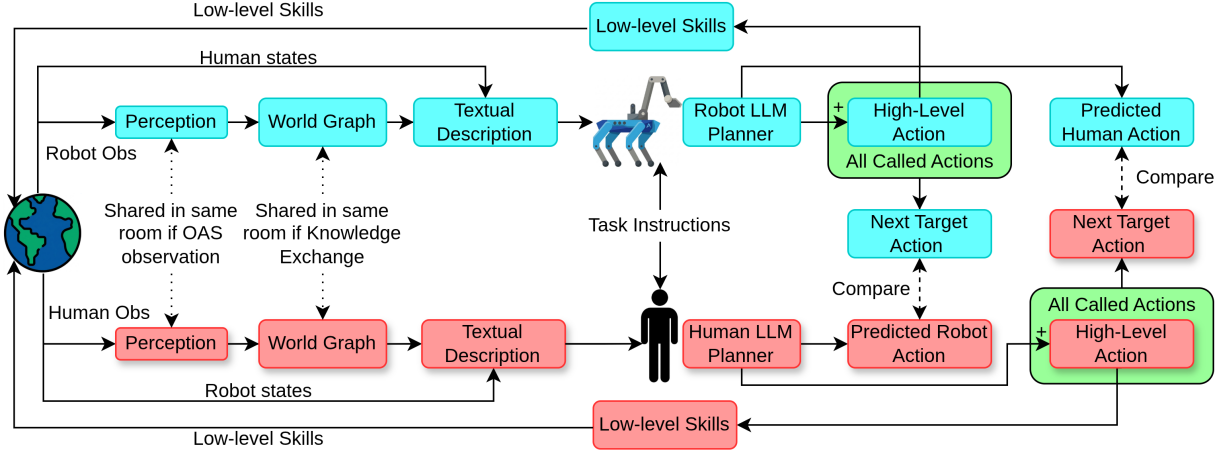


Figure 2: **Decoupled architecture with Other Agent’s State (OAS) observation, object Knowledge Exchange (KE), and Other Agent’s Next Action (OANA) prediction.** Both agents build a world graph using their own environmental object observations and their actions. The graph’s nodes represent the house, rooms, furniture, receptacles, objects, and agents. The edges represent the type of relation between them. If allowed, the agents also observe the OAS and exchange their knowledge about the objects when they are in the same room. Their high-level LLM planners call a High-Level Action (at replanning) and the corresponding low-level skills (action execution in the simulator), and predict the OANA. The predicted OANA is then compared with the other agent’s current and next actions (union target, Section 3.3), retrieved from its list of all called high-level actions.

state  $oas_{j,t}$ , such as **Walking in hallway\_1**, **Picking apple\_0**, or **Standing**; while they are in different rooms, it receives **Acting in another room**. These states contain no internal state of the partner, only what would be visible if the agent looked at it. They are read from simulator logs, so no action-recognition errors enter (a real system would need such a model, e.g., on RGB-D input), and the coordination capabilities of the planners can be studied in isolation. The observation  $o_i^{(k)}$  in Eq. (1) then consists of the object summary followed by the partner’s states during  $(\tau_i^{(k-1)}, \tau_i^{(k)})$ ; if the partner’s state did not change in this interval, the last observed state is repeated. In addition, on co-location, each agent adds its partner’s current percepts to its own world graph, as in Eq. (2). This keeps its object knowledge consistent with the manipulations it watches the partner perform (e.g., an object the partner is carrying is no longer recorded at its old location). Unlike KE, it shares only what the partner perceives at that moment, not the object knowledge the partner has accumulated earlier in the episode.

**Object Knowledge Exchange (KE).** The synchronization in Eq. (3) resolves discrepancies that arise when the agents act independently in different rooms. Although Sync reads the simulator’s global world graph  $g_t^{\text{global}}$ , it adds no information that the two agents do not jointly hold: only the agents move objects or change their states, and each agent records its own manipulations, so for every object known to either agent, the more recent of the two records is up to date. Reading the value from  $g_t^{\text{global}}$  is therefore equivalent to keeping the more recent record and spares the agents from resolving conflicting records, which they could not do without dialogue. Object KE is an information channel between the agents, and we characterize it explicitly rather than describing our setting as communication-free. It carries only object positions and states, never intentions, plans, or task-level messages (*content*); it merges exchanged knowledge into the receiver’s world graph with no sender tag, so an agent cannot tell whether it perceived a change itself or acquired it from its partner (*attribution*); it operates only while the agents share a room, as a consequence of physical co-presence (*gating*); and it is read from the simulator rather than produced and parsed by the planners, so it is lossless and instantaneous, an upper bound on what an equivalent real protocol could deliver (*fidelity*). Where KE fails to help, or harms a planner (Section 4.2), the cause therefore cannot be communication overhead, latency, or message error; it must lie in how the planner uses the information, most directly in whether it can infer agency from an unexplained change in the world.

Figure 1 shows the agents’ observations under each condition, and Appendix A.6 shows an example prompt.

### 3.3 Other Agent’s Next Action (OANA) Prediction

When OANA prediction is enabled, the planner of agent  $i$  first reasons about both agents’ next actions, then predicts the partner’s next high-level action  $\hat{a}_j^{(k)}$ , and then selects its own, so Eq. (1) becomes

$$(R_i^{(k)}, \hat{a}_j^{(k)}, a_i^{(k)}) \sim \text{LLM}_i(\cdot \mid H_i^{(k-1)} \oplus o_i^{(k)}), \quad (4)$$

and all three outputs are appended to the history. With OANA enabled, the prompt also contains the descriptions of the partner’s action tools, so part of any OANA effect could come from knowing what the partner can do. The human predicts over the robot’s 9 action types and the robot over the human’s 14, so the robot faces the larger prediction space. On average, an episode offers 11.7 ( $\pm 3.8$ ) rooms, 56.3 ( $\pm 20.3$ ) pieces of furniture, and 5.4 ( $\pm 1.9$ ) objects as possible action parameters.

**Targets.** Because the agents replan asynchronously, the partner is usually in the middle of an action when agent  $i$  makes a prediction at  $\tau = \tau_i^{(k)}$ . Let  $a_j^{\text{cur}}(\tau)$  be the action the partner is executing at step  $\tau$  and  $a_j^{\text{next}}(\tau)$  the action it selects at its first replanning step after  $\tau$ . We score each prediction against two target sets:

$$\mathcal{T}_j^{\text{union}}(\tau) = \{a_j^{\text{cur}}(\tau), a_j^{\text{next}}(\tau)\}, \quad \mathcal{T}_j^{\text{strict}}(\tau) = \{a_j^{\text{next}}(\tau)\}, \quad (5)$$

and a prediction counts as correct if it matches any element of the target set. The union target allows for the asynchrony: an agent that observes `Walking in living_room_1` may name the ongoing `Navigate[table_1]`, which the union target accepts, although the partner’s next action is `Place[cup_1, on, table_1]`. Because this credits recognizing an ongoing action as if it were anticipation, we also report the strict target (Section 4.5). Under both targets, once the partner has called `Done[]`, the only correct prediction is `Done[]`.

**Matching.** Predictions are compared at the class level: instance IDs are masked (e.g., `spoon_3`  $\rightarrow$  `spoon`, giving `Rearrange[spoon, on, table, next_to, plate]`), so that an agent is not penalized for referring to an object it has not yet seen or to another instance of the same class. Matches are evaluated at several granularities, from the action type alone to all parameters (Section 4.4; protocol in Appendix A.2). For the Llama planners, the decoding grammar restricts predicted objects to the classes present in the scene, which are available from the start of the episode (Appendix A.1.2), an advantage the API-accessed planners lack. Both OAS observation and OANA prediction also lengthen the prompt. Appendix A.1 details the action space; further details are in Chang et al. (2025).

## 4 Experiments and Analysis

### 4.1 Experimental Setup

We conduct our experiments in the Habitat 3.0 (Puig et al., 2024) simulator, extending the PARTNR (Chang et al., 2025) framework with our proposed modifications. All experiments build on the PARTNR *decentralized\_zero\_shot\_react\_summary* baseline configuration, which includes decentralized control, a zero-shot ReAct planner, privileged perception, oracle skills (i.e., simulator-based execution rather than learned action policies), and summarized object observations. We keep all these configuration settings unchanged, with a maximum timeout of 50 replanning steps and 20,000 simulator steps per agent, the LLM decoding temperature set to 0, and at most 250 generated tokens per replanning step. Evaluation is performed on the PARTNR validation set, comprising 13 house scenes, a diverse set of household objects, and 1,000 tasks. Due to limited computational resources, we report results from a single run per configuration. Prior work on the PARTNR baseline (Chang et al., 2025) reports low variance across seeds. With 1,000 episodes, the binomial standard error of a single Success Rate is 1.2–1.6 points, so we interpret differences of a few points between individual runs with caution and base our conclusions mainly on effects that recur across configurations; for example, OANA prediction lowers Success Rate in 15 of 16 model $\times$ context pairs, which a sign test treating the pairs as independent would assign a two-sided probability of  $5 \times 10^{-4}$  under no effect. Multi-seed evaluation would nevertheless give more robust estimates, and we leave it for future work.

Table 1: **Task outcomes and redundant work under Decoupled Partial Observability** for every combination of Other Agent’s State (OAS) observation, object Knowledge Exchange (KE) on co-location, and Other Agent’s Next Action (OANA) prediction. SR: Success Rate (%), PC: Percent Complete (%), Steps: Simulation Steps, EE: Extraneous Effort (%). The best value per model is bolded. All coordination metrics (Planning Cycles, Other-Agent Mention Ratio, Exploration Efficiency, Task Offloading, Extra Planning, Action Success), reported per agent, are in Table 3 (Appendix A.3).

| OAS      | KE       | OANA     | Llama-3.1-8B |             |             |             | Llama-3.1-70B (4-bit) |             |             |             | Gemini 3.1 Flash Lite |             |             |             | DeepSeek V3.2 |             |             |             |
|----------|----------|----------|--------------|-------------|-------------|-------------|-----------------------|-------------|-------------|-------------|-----------------------|-------------|-------------|-------------|---------------|-------------|-------------|-------------|
|          |          |          | SR↑          | PC↑         | Steps↓      | EE↓         | SR↑                   | PC↑         | Steps↓      | EE↓         | SR↑                   | PC↑         | Steps↓      | EE↓         | SR↑           | PC↑         | Steps↓      | EE↓         |
| <b>X</b> | <b>X</b> | <b>X</b> | 60.9         | <b>77.1</b> | 4214        | 62.3        | 76.0                  | 86.9        | 3056        | 39.4        | 80.4                  | 89.8        | 4118        | 44.3        | 82.3          | 91.0        | 3857        | 38.2        |
| <b>X</b> | <b>X</b> | <b>✓</b> | 55.0         | 73.0        | 4243        | 62.0        | 72.9                  | 85.6        | 3648        | 43.5        | 76.0                  | 88.6        | 4304        | 43.5        | 75.2          | 87.1        | 5486        | 35.0        |
| <b>X</b> | <b>✓</b> | <b>X</b> | 54.3         | 72.9        | 4914        | 69.6        | 76.9                  | <b>87.6</b> | 3041        | 43.0        | 79.9                  | 89.4        | 3081        | 38.2        | 82.2          | 90.6        | 2951        | 34.8        |
| <b>X</b> | <b>✓</b> | <b>✓</b> | 56.2         | 72.9        | 4121        | 62.7        | 72.1                  | 84.6        | 3125        | 42.1        | 75.8                  | 88.3        | 3416        | 35.1        | 75.6          | 87.4        | 4249        | 31.9        |
| <b>✓</b> | <b>X</b> | <b>X</b> | 58.8         | 74.7        | 4022        | <b>58.5</b> | 76.2                  | 86.5        | 3425        | <b>36.5</b> | <b>81.2</b>           | 89.8        | 3193        | 33.0        | <b>83.6</b>   | <b>91.5</b> | 3112        | 28.1        |
| <b>✓</b> | <b>X</b> | <b>✓</b> | 56.6         | 74.2        | 4017        | <b>58.5</b> | 72.5                  | 84.3        | 3019        | 38.0        | 79.1                  | 88.6        | 3453        | 32.2        | 78.4          | 88.8        | 3927        | <b>26.7</b> |
| <b>✓</b> | <b>✓</b> | <b>X</b> | <b>62.3</b>  | 76.8        | <b>3781</b> | <b>58.5</b> | <b>77.5</b>           | <b>87.6</b> | 3095        | <b>36.5</b> | 81.1                  | <b>90.3</b> | <b>2911</b> | 33.3        | 82.4          | 90.5        | <b>2794</b> | 27.5        |
| <b>✓</b> | <b>✓</b> | <b>✓</b> | 57.2         | 74.5        | 3890        | 59.1        | 72.8                  | 84.6        | <b>2849</b> | 36.8        | 78.6                  | 89.1        | 3107        | <b>32.1</b> | 80.0          | 89.4        | 3551        | 27.0        |

**Metrics.** We report task outcomes, which measure the collaboration as a whole, and coordination metrics, which measure how the two agents’ activities fit together and are reported per agent where applicable. *Simulation Steps* measures the number of simulator steps (simulator time) required to complete a task, reflecting execution efficiency. *Success Rate* (SR) indicates whether all task constraints are satisfied (0 or 1 for an episode), while *Percent Complete* (PC) captures partial progress by measuring the fraction of completed task propositions (equal to 1 for fully successful tasks). To assess planning efficiency, *Planning Cycles* counts the number of high-level LLM calls made by each agent. We further analyze coordination through *Other-Agent Mention Ratio*, defined as the fraction of replanning steps in which an agent explicitly mentions “other agent” or “agent b” in the reasoning part (the prompts refer to the partner by these terms; Appendix A.6). It therefore measures how often an agent considers its partner in its reasoning, not the quality of that reasoning. We also measure *Exploration Efficiency* as the first low-level simulation step at which an agent successfully picks up an object. *Task Offloading* quantifies the proportion of subtasks completed by the robot, while *Extraneous Effort* measures the ratio of unnecessary rearrangements to total actions. Finally, *Extra Planning Ratio* captures the proportion of replanning steps performed after the partner has finished acting, indicating inefficiencies in coordination, and *Action Success Rate* reports the fraction of successfully executed actions. With OANA prediction enabled, the Extra Planning Ratio also gives the share of an agent’s predictions whose target is Done[] (Section 3.3), since every prediction made after the partner has finished is scored against Done[].

## 4.2 Task Outcomes and Coordination

Table 1 reports task outcomes and redundant work for all 32 runs under Decoupled Partial Observability, and Figure 3 plots each configuration’s Success Rate against its episode length; the remaining coordination metrics are in Table 3 (Appendix A.3). The Llama planners run locally with PARTNR’s grammar-constrained decoding (Chang et al., 2025), extended for OANA prediction (Appendix A.1.2); Gemini and DeepSeek are accessed through an API, where constrained decoding is unavailable, but still achieve the highest task performance.

**OAS observation primarily improves task quality; KE primarily improves execution efficiency.** Adding OAS alone to the object-only baseline reduces Extraneous Effort for every planner (Llama 8B 62.3%→58.5%, Llama 70B 39.4%→36.5%, Gemini 44.3%→33.0%, DeepSeek 38.2%→28.1%), raises Action Success for the three stronger planners, and yields the best Success Rate of all configurations for Gemini (81.2%) and DeepSeek (83.6%). KE alone leaves Success Rate essentially unchanged for the same three planners (70B 76.0%→76.9%, Gemini 80.4%→79.9%, DeepSeek 82.3%→82.2%) but shortens episodes by 25% for Gemini (4118→3081 steps) and 23% for DeepSeek (3857→2951), with fewer Planning Cycles and less replanning after the partner has finished (Table 3). This matches the semantics of the two channels: OAS

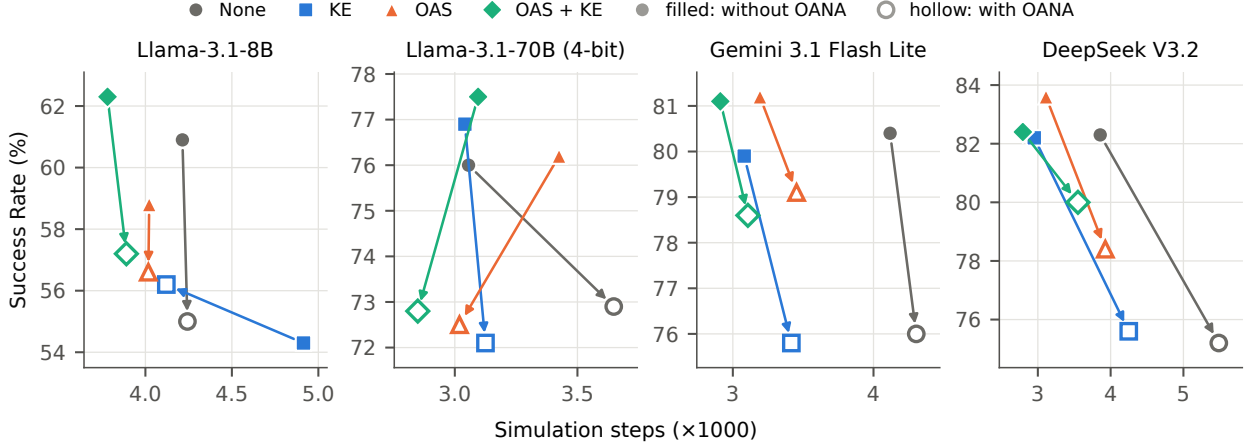


Figure 3: **Success Rate against episode length (Simulation Steps) for every configuration in Table 1.** Marker shape and color give the shared context (none, KE, OAS, OAS + KE); filled markers are runs without OANA prediction, and each arrow points to the same context with prediction enabled. For Gemini and DeepSeek, KE and OAS both shorten episodes, and OAS also raises Success Rate; all arrows but one point down, since prediction lowers Success Rate in fifteen of sixteen model $\times$ context pairs (the exception is the Llama 8B with KE only). The y-axis ranges differ per panel.

tells an agent what its partner is currently doing, and therefore what not to duplicate, whereas KE removes redundant search by synchronizing object knowledge on co-location.

**The two channels are largely redundant for strong planners but complementary for the weakest one.** For Gemini and DeepSeek, OAS-only and KE-only each recover most of the attainable efficiency gain, and combining them adds little (Gemini 3193 and 3081 $\rightarrow$ 2911 steps; DeepSeek 3112 and 2951 $\rightarrow$ 2794). For the Llama 8B, KE without OAS is its single worst configuration (Success Rate 60.9% $\rightarrow$ 54.3%, 4914 steps, Extraneous Effort 69.6%), OAS alone does not help either (58.8%), and only the combination recovers its best Success Rate (62.3%) and fewest steps (3781). We interpret this as an attribution failure: synchronized object knowledge without any signal about what the partner is doing with those objects (Section 3.2) can enlarge the world graph the 8B must reason over without giving it a way to assign newly appearing objects to the partner’s activity, so it replans more often and duplicates work. The Llama 70B moves in the same direction on Extraneous Effort (39.4% $\rightarrow$ 43.0% with KE alone) without losing Success Rate (76.0% $\rightarrow$ 76.9%; Appendix A.3.1). Shared context is therefore not uniformly beneficial; it must be interpretable by the planner that receives it.

**OAS + KE is the most efficient operating point, while peak Success Rate moves to OAS-only for the strongest planners.** Without prediction, OAS + KE gives the fewest Simulation Steps for the Llama 8B (3781), Gemini (2911), and DeepSeek (2794), comes within 2% of the fastest setting for the Llama 70B (3095 vs. 3041), and gives the lowest or near-lowest (within 0.1 points) Extra Planning ratios for every model. For Gemini and DeepSeek, OAS + KE comes within 1.2 points of the OAS-only Success Rate (81.1% vs. 81.2%; 82.4% vs. 83.6%), within the uncertainty of a single run (Section 4.1), so we read the two settings as equivalent in accuracy and separated by how efficiently that accuracy is obtained. When only one channel can be implemented, KE is the cheaper choice and OAS the more accurate one (Appendix A.3.1).

**OANA prediction lowers Success Rate and Percent Complete in fifteen of sixteen model $\times$ context pairs.** All arrows but one in Figure 3 point down. Across these fifteen pairs, the Success Rate drop ranges from  $-2.1$  points (Gemini, OAS only) to  $-7.1$  points (DeepSeek, no shared context), while the Other-Agent Mention Ratio rises from 10–53% to 55–100%: the agents reason explicitly about the partner far more, but that reasoning does not convert into completed task propositions. The behavior is consistent with premature commitment: agents act on an anticipated partner action instead of the observed situation, and a wrong prediction can leave a subtask undone as “already handled.” Duplication is not the main cost, since

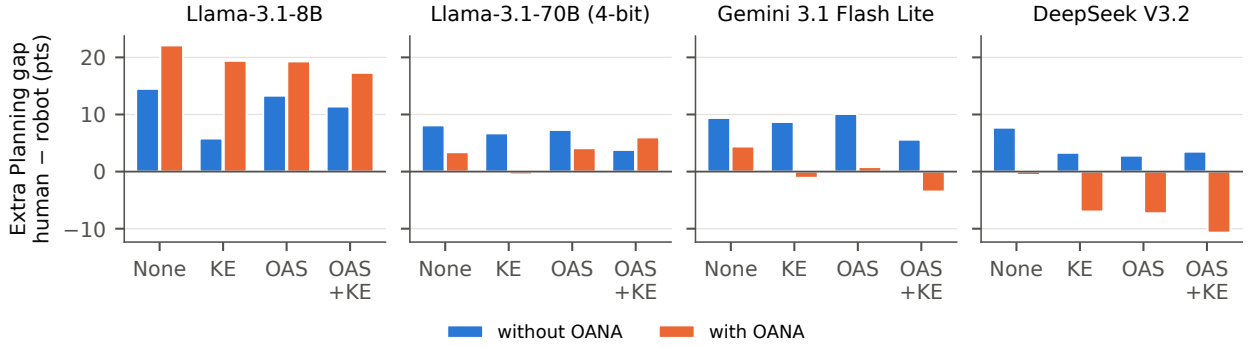


Figure 4: **Completion lag between the agents:** human minus robot Extra Planning Ratio (percentage points) for each shared context, without (blue) and with (orange) OANA prediction. Positive values mean the human keeps replanning after the robot has finished. Without prediction, the human trails in every configuration, and the full shared context (OAS + KE) shrinks the lag in every model; prediction widens it for the Llama 8B and reverses it for DeepSeek, for Gemini in two contexts, and marginally for the Llama 70B with KE only (−0.5 points).

Extraneous Effort falls under prediction in 11 of 16 pairs; prediction makes agents yield to each other more, not collide more. The failures it adds are also ordinary ones: the average completion of a failed episode,  $(PC - SR)/(100 - SR)$ , stays between 38% and 53% in all 32 runs (Appendix A.3.1). The one exception is the Llama 8B with KE but no OAS, the configuration in which KE alone was most harmful: there, prediction raises Success Rate from 54.3% to 56.2% (less than the 2.2-point standard error of a difference between two single runs) and leaves Percent Complete unchanged at 72.9%, consistent with prediction partly supplying the partner attribution that ungrounded KE lacks (Appendix A.3.1).

**For the API-accessed planners, the cost of prediction shrinks as shared context grows; for the Llama 70B, only its time cost does.** DeepSeek’s Success Rate penalty decreases monotonically with available context (−7.1, −6.6, −5.2, −2.4 points for none, KE, OAS, and OAS + KE), as does its step overhead (+1629, +1298, +815, +757); Gemini shows the same ordering more weakly (−4.4/−4.1 without OAS vs. −2.1/−2.5 with it). For the Llama 70B, prediction costs 592 and 84 additional steps without OAS but saves 406 and 246 once OAS is available, giving its fastest configuration overall (2849 steps). Anticipating a partner whose state is unobserved amounts to guessing from object positions alone; once the partner’s state is visible, predictions can build on it, and part of their overhead is recovered. How prediction changes planning volume and the quality of individual actions also differs by planner family (Appendix A.3.1).

**Partner-mention frequency tracks task success only when it is spontaneous.** Without prediction, the Other-Agent Mention Ratio rises with planner strength across our four planners (robot: 9.8%, 13.3%, 30.4%, 39.1% for the Llama 8B, Llama 70B, Gemini, and DeepSeek) and orders the models exactly as Success Rate does (60.9%, 76.0%, 80.4%, 82.3%). Once prediction is instructed, mentions reach 55–92% for the Llama and Gemini planners and essentially 100% for DeepSeek, yet Success Rate falls in all but one case, and the OAS + KE configuration without prediction, the most efficient operating point above, has comparatively low mention rates in every model (Llama 8B 21.0/30.9%, Llama 70B 18.1/24.7%, Gemini 27.5/36.5%, DeepSeek 48.6/51.4%). The metric is informative as an unprompted behavioral signal and uninformative once instructed, so on its own, it cannot separate endogenous partner modeling from prompt-compliant mentioning. Observation-grounded partner reasoning is less frequent but more useful than mandated prediction.

### 4.3 Role Asymmetry Between the Agents

The two agents differ in their action spaces, since the human additionally holds the object-state manipulation actions *Clean*, *Fill*, *Pour*, *PowerOn*, and *PowerOff* (Appendix A.1.1), and in sensor height, and the prompts inherited from PARTNR make the human the task giver and the robot the task receiver (Appendix A.6). Both run the same planner, prompt structure, and observability configuration, so systematic differences

between them reflect the action spaces, the prompted roles, and the task structure these induce, and possibly sensor height, rather than model capability; our design does not separate these sources.

**The robot acquires objects first, and the human finishes last.** The robot reaches its first object interaction earlier than the human in 28 of 32 runs (mean gap 482, 205, 180, and 18 steps for the Llama 8B, Llama 70B, DeepSeek, and Gemini), since its repertoire consists of navigation and rearrangement primitives that converge on a pickup. Conversely, the human replans more after its partner has finished than the robot does in 25 of 32 runs (Figure 4), although Task Offloading exceeds 50% in all 32 runs: the robot completes more subtasks by count while going idle first. Together with the prompted roles, a likely explanation is that the human’s exclusive state-modification actions sit at the ends of dependency chains, since an object usually must be transported before it is cleaned, filled, or switched on, so they are serialized behind the robot’s rearrangements. The division of labor itself barely moves: Task Offloading stays within 52.4–60.7% across all 32 runs while Success Rate varies from 54.3% to 83.6% (Appendix A.3.2).

**Shared context helps chiefly by shortening the human’s trailing window.** Without prediction, moving from object-only observation to OAS + KE shrinks the completion lag in every model (Llama 8B +14.5→+11.4, Llama 70B +8.1→+3.8, Gemini +9.4→+5.6, DeepSeek +7.7→+3.5 points), and the human’s own Extra Planning falls by more than the robot’s in every case. Because episode length is set by the later-finishing agent, this is what turns into the Simulation Step reductions of Section 4.2, and it explains why OAS and KE improve efficiency more reliably than Success Rate.

**Prediction reverses the lag for strong planners and amplifies it for the weakest one.** Under OANA, DeepSeek’s robot Extra Planning roughly doubles (20.3%→39.0% at baseline), turning the lag negative in all four contexts; Gemini reverses in two of four, and the Llama 70B marginally in one (KE only, −0.5 points); for the Llama 8B, the lag widens instead (+14.5→+22.1 points at baseline). Since the robot predicts over the human’s 14 action types while the human predicts over the robot’s 9, prediction imposes an asymmetric burden that strong planners absorb as robot-side deliberation. The human also executes less reliably (robot Action Success is at least as high in 28 of 32 runs), sees its Action Success change less favorably than the robot’s when prediction is enabled in 13 of 16 pairs, and mentions the partner more often in all 16 no-prediction runs; which agent replans more overall depends on the planner (Appendix A.3.2). Under partial observability, the robot thus acts as a transport layer, and the human as a finisher that determines when the episode ends. For assistive settings in which the human side is a person, this pattern calls for caution: a robot that maximizes completed subtasks can still leave its partner working alone at the end of the task, and partner-state access plus knowledge exchange address that failure more directly than raising the robot’s share of the work would. Much of the lag, however, is produced by partial observability itself (Section 4.5).

#### 4.4 Partner Prediction

Table 2 reports how accurately agents predict each other’s next action in the OANA-enabled runs of Table 1, scored against the union target (Section 3.3) at several levels of abstraction: the action type alone (e.g., Pick, Navigate), the principal parameter (e.g., cup in Place[cup, on, table]), all parameters, their combinations, and conditional accuracies scored only over predictions whose action type already matches the target. By construction,  $\text{Acc}(\text{type} \wedge \text{params}) = \text{Acc}(\text{type}) \times \text{Acc}(\text{params} \mid \text{type})$ , an identity we verified to within rounding in all 64 model×context×agent×granularity cells of Table 2. We use it to assign every change to the coarse term (what the partner will do) or the conditional term (which objects, furniture, or room it will act on). Figure 5 plots both terms for each planner, context, and agent, under both targets of Section 3.3; Section 4.5 discusses the strict one.

**Shared context tells an agent what its partner will do far more than which object it will act on.** OAS + KE gives the top Action Type accuracy for every model and agent (strictly in seven of eight cases), with gains over the object-only baseline of +17.0/+14.0 points for the Llama 70B, +14.4/+16.3 for Gemini, and +21.1/+23.5 for DeepSeek, although neither channel transmits intentions or plans, only observed states and object locations. The two channels overlap: the sum of the single-channel gains exceeds the combined gain in seven of eight model×agent cases (Gemini realizes about 60% of the sum, DeepSeek 83%). Averaged over the four planners, the conditional principal-parameter accuracy improves by only +4.2/+1.0 points, against +14.7/+14.6 for Action Type, and even the highest conditional accuracy in the table (Llama 70B

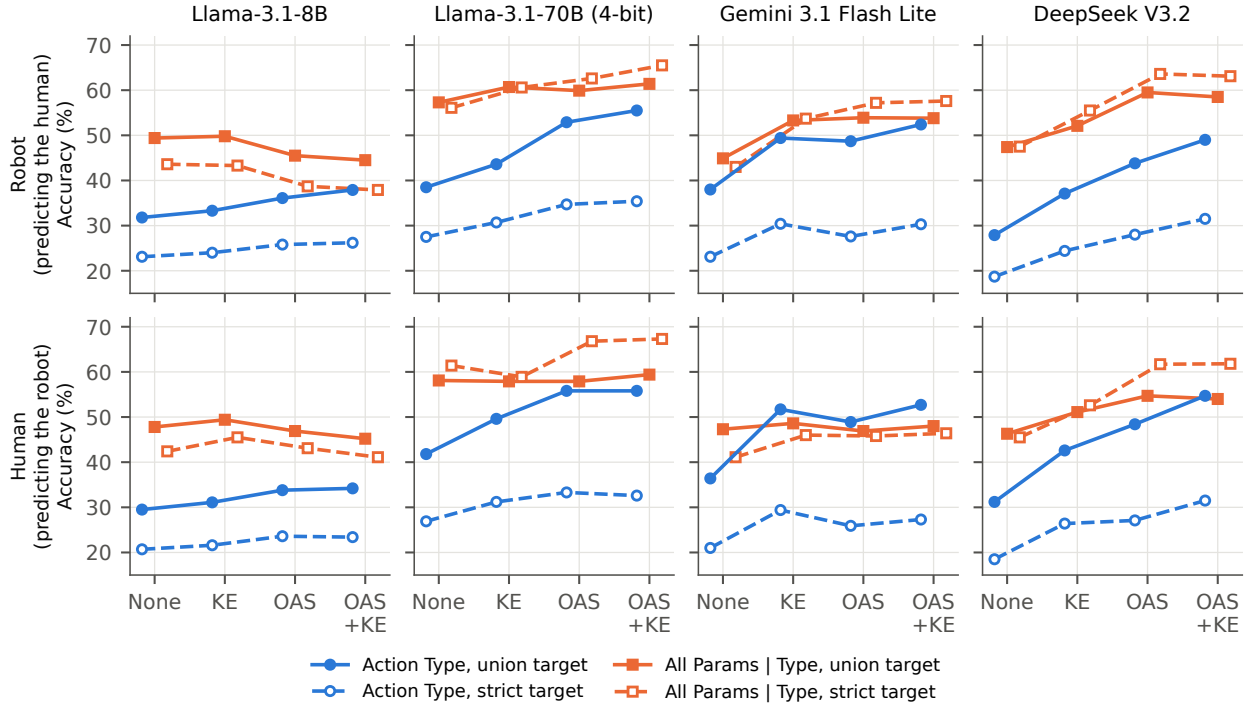


Figure 5: **Partner-prediction accuracy by shared context, for the robot predicting the human (top) and the human predicting the robot (bottom):** Action Type accuracy (blue) and all-parameter accuracy given a correct action type (orange), under the union target (solid, filled markers; Table 2) and the strict next-action target (dashed, hollow markers; Table 7). Each point is a table entry; markers are offset horizontally so that close values remain visible. From None to OAS + KE, Action Type accuracy rises more than the conditional term for every planner and agent (+14.7/+14.6 against +4.8/+1.8 points for robot/human, averaged over planners, union target). The strict target lowers Action Type accuracy by 36.5% on average (27–48% across cells), while the conditional term is unchanged on average (52.2% vs. 52.4%) but planner-dependent: lower in every cell for the Llama 8B and higher in most cells for the Llama 70B and DeepSeek.

with KE only, 67.6/64.5%) leaves about a third of correctly typed predictions with the wrong principal parameter. Predicting the arguments remains the main difficulty. For the Llama 8B, richer context even lowers the conditional term (all-parameter accuracy given the type, 49.4%→45.5% with OAS, robot) while raising the type term, so its joint accuracy barely moves (15.7%→16.4%).

**Whether object knowledge alone suffices depends on the planner.** For Gemini, KE alone matches or exceeds the Action Type gain of OAS alone (+11.4 vs. +10.7 for the robot, +15.3 vs. +12.5 for the human); for DeepSeek, it recovers about 60% of it, for the Llama 70B about a third on the robot side and about half on the human side (+5.1 vs. +14.4; +7.8 vs. +14.0), and for the Llama 8B about a third. A plausible account is self-projection: once both agents hold the same object beliefs, an agent can predict its partner by asking what it would itself do next, a route that the API-accessed planners, Gemini in particular, appear to exploit more than the Llama planners, which rely mainly on reading off the partner’s state. Constrained decoding cannot explain this difference in action type, since with KE only the unconstrained Gemini still predicts the partner’s action type better than the constrained Llama 70B (49.4/51.7% vs. 43.6/49.6%), although it may contribute to the Llama 70B’s higher conditional parameter accuracy (Appendix A.3.3).

**Task-solving strength does not predict partner-modeling strength.** DeepSeek achieves the best Success Rate of all four planners, yet at the object-only baseline its robot is the weakest partner predictor in the table (27.9% Action Type, below the Llama 8B’s 31.8%), whereas the third-best task solver, the 4-bit Llama 70B, is the best predictor under full shared context (55.5/55.8%). Strong task performance thus comes

Table 2: **Other Agent’s Next Action (OANA) prediction accuracy (%)** across action abstraction levels for robot (R) and human (H) agents. The target action is the **union of the current and next actions** of the other agent. Results highlight that Other Agent’s State (OAS) observation and object Knowledge Exchange (KE) in the same room both improve prediction, that their combination gives the best action-type accuracy for every model and agent, and that the gains are concentrated in the action type rather than in its parameters. Per-model bests are underlined; overall bests are bolded.

| Model                        | OAS | KE | Action Type (R/H) ↑       | Principal Par. (R/H) ↑    | Action Type+Princ. Par. (R/H) ↑ | All Params (R/H) ↑        | Action Type +All Params (R/H) ↑ | Principal Par. if Type (R/H) ↑ | All Params if Type (R/H) ↑ |
|------------------------------|-----|----|---------------------------|---------------------------|---------------------------------|---------------------------|---------------------------------|--------------------------------|----------------------------|
| Llama-3.1-8B-Instruct        | ✗   | ✗  | 31.8 / 29.5               | 24.6 / 21.7               | 17.2 / 15.8                     | 17.8 / 15.4               | 15.7 / 14.1                     | 54.2 / 53.5                    | 49.4 / 47.8                |
|                              | ✗   | ✓  | 33.3 / 31.1               | <u>26.1</u> / <u>24.2</u> | 18.4 / 17.8                     | <u>18.8</u> / <u>17.2</u> | 16.6 / 15.4                     | <u>55.3</u> / <u>57.2</u>      | <u>49.8</u> / <u>49.4</u>  |
|                              | ✓   | ✗  | 36.1 / 33.8               | 22.5 / 21.5               | 17.8 / <u>17.9</u>              | 18.1 / 17.0               | 16.4 / <u>15.8</u>              | 49.2 / 53.1                    | 45.5 / 46.9                |
|                              | ✓   | ✓  | <u>37.9</u> / <u>34.2</u> | 23.2 / 21.6               | <u>18.9</u> / <u>17.2</u>       | 18.5 / 16.8               | <u>16.9</u> / 15.5              | 49.9 / 50.4                    | 44.5 / 45.2                |
| Llama-3.1-70B-Instruct 4-bit | ✗   | ✗  | 38.5 / 41.8               | 34.2 / 37.7               | 24.1 / 26.5                     | 26.7 / 30.3               | 22.1 / 24.3                     | 62.5 / 63.3                    | 57.3 / 58.1                |
|                              | ✗   | ✓  | 43.6 / 49.6               | 39.1 / 41.2               | 29.5 / 32.0                     | 30.8 / 34.0               | 26.5 / 28.7                     | <b>67.6</b> / <b>64.5</b>      | 60.7 / 57.9                |
|                              | ✓   | ✗  | 52.9 / <b>55.8</b>        | 42.4 / 41.8               | 35.2 / 34.7                     | 36.1 / 36.8               | 31.7 / 32.3                     | 66.4 / 62.2                    | 59.9 / 57.9                |
|                              | ✓   | ✓  | <b>55.5</b> / <b>55.8</b> | <b>45.2</b> / <b>43.6</b> | <b>37.3</b> / <b>35.8</b>       | <b>38.5</b> / <b>38.2</b> | <b>34.1</b> / <b>33.2</b>       | 67.2 / 64.1                    | <b>61.4</b> / <b>59.4</b>  |
| Gemini 3.1 Flash Lite        | ✗   | ✗  | 38.0 / 36.4               | 28.0 / 26.4               | 19.7 / 19.1                     | 21.9 / 21.0               | 17.1 / 17.2                     | 51.7 / <u>52.6</u>             | 44.9 / 47.3                |
|                              | ✗   | ✓  | 49.4 / 51.7               | 38.5 / 36.2               | 28.8 / <u>27.2</u>              | 32.7 / 30.9               | 26.3 / 25.1                     | <u>58.3</u> / <u>52.6</u>      | 53.3 / <u>48.6</u>         |
|                              | ✓   | ✗  | 48.7 / 48.9               | 39.0 / 36.4               | 28.1 / 24.7                     | 32.6 / 29.0               | 26.3 / 23.0                     | 57.7 / 50.6                    | <u>53.9</u> / 46.9         |
|                              | ✓   | ✓  | <u>52.4</u> / <u>52.7</u> | <u>42.7</u> / <u>39.6</u> | <u>30.4</u> / 27.1              | <u>35.4</u> / <u>31.8</u> | <u>28.1</u> / <u>25.3</u>       | 58.1 / 51.5                    | 53.8 / 48.0                |
| DeepSeek V3.2                | ✗   | ✗  | 27.9 / 31.2               | 21.3 / 23.9               | 14.4 / 15.6                     | 16.8 / 18.6               | 13.2 / 14.4                     | 51.7 / 50.2                    | 47.4 / 46.3                |
|                              | ✗   | ✓  | 37.1 / 42.6               | 28.6 / 33.8               | 21.0 / 23.2                     | 23.2 / 27.0               | 19.3 / 21.8                     | 56.6 / 54.6                    | 52.1 / 51.1                |
|                              | ✓   | ✗  | 43.8 / 48.4               | 34.2 / 37.2               | 27.2 / 27.7                     | 29.7 / 31.6               | 26.0 / 26.5                     | <u>62.2</u> / 57.3             | <u>59.5</u> / <u>54.7</u>  |
|                              | ✓   | ✓  | <u>49.0</u> / <u>54.7</u> | <u>38.4</u> / <u>40.6</u> | <u>30.2</u> / <u>31.4</u>       | <u>32.9</u> / <u>34.1</u> | <u>28.7</u> / <u>29.6</u>       | 61.6 / <u>57.4</u>             | 58.5 / 54.0                |

mainly from individual planning, and partner-mention frequency is again no guide, since DeepSeek mentions its partner in essentially every replanning step under OANA. Within the API-accessed planners, however, accuracy does track the cost of prediction: as DeepSeek’s robot Action Type accuracy rises through 27.9%, 37.1%, 43.8%, and 49.0% for none, KE, OAS, and OAS + KE, its Success Rate penalty falls through  $-7.1$ ,  $-6.6$ ,  $-5.2$ , and  $-2.4$  points, consistent with agents acting on their predictions. Gemini follows this only loosely and the Llama planners not at all, so we present it as an account of the API-accessed models rather than a general law.

**The two agents predict different parts of the action.** The human predicts over the robot’s 9 action types, and the robot over the human’s 14. Under the union target, the human predicts the action type more accurately for the Llama 70B and DeepSeek in all four contexts and for Gemini in three, while the robot predicts the parameters more accurately once the type is right, a lead that appears only with shared context ( $-3.1$  to  $-4.2$  points H–R for the Llama 70B,  $-5.7$  to  $-7.1$  for Gemini,  $-2.0$  to  $-4.9$  for DeepSeek). The latter plausibly reflects arity: the robot’s Place and Rearrange take five argument slots, whereas the five human-only actions take one. The two asymmetries cancel in the joint metric, whose H–R gap stays within  $\pm 3.3$  points in all 16 cells, so joint accuracies alone would hide both. The Llama 8B inverts the type asymmetry, and neither asymmetry is robust to the strict target (Section 4.5; details in Appendices A.3.3 and A.5.1).

#### 4.5 Robustness: Strict Target and Full Observability

We test the main findings under two stricter conditions (details in Appendices A.4 and A.5). The *strict target* (Section 3.3) re-scores the same runs against the partner’s next action only, so naming an ongoing action no longer counts (Table 7). *Full observability* gives both agents the complete ground-truth world graph at all times, which removes perceptual uncertainty, makes KE inapplicable, and makes OAS available continuously rather than only on co-location; we run it with the two Llama planners (Tables 4, 8 and 9).

**Part of what the union target credits is recognition rather than anticipation.** The strict target lowers Action Type accuracy by 36.5% on average while leaving the all-parameter conditional term essentially unchanged on average (Figure 5), and the gain from enabling OAS roughly halves (Llama 70B: +14.4/+14.0→+7.2/+6.4 points), because an observation such as *Walking in hallway\_1* lets an agent name the ongoing Navigate action without anticipating anything. Under full observability, enabling OAS raises the Llama 70B’s union-target Action Type accuracy by +8.6/+6.6 points but leaves its strict accuracy unchanged or lower (−0.3/−2.8, Tables 8 and 9). The agent-level asymmetries of Section 4.4 also depend on the scoring: under the strict target, the robot matches or leads the human on action type in 14 of 16 cells, and its conditional parameter lead survives clearly only for Gemini.

**Anticipation stays weak even with complete world knowledge, and prediction still costs success.** The best strict-target Action Type accuracy is 35.4/33.3% under partial observability (Llama 70B) and 43.8/45.9% under full observability without OAS, so even an agent that knows the entire world names its partner’s next action type wrongly more than half the time. OANA prediction still lowers Success Rate and Percent Complete in all four full-observability configurations (nineteen of twenty pairs across both regimes, Table 4), so prediction does not hurt merely because agents anticipate partners they cannot see. Its cost depends on capacity: under full observability, the Llama 70B’s penalty shrinks to −1.6 and −0.6 points, whereas the Llama 8B’s penalties (−6.2, −2.3) match its partial-observability penalties (−5.9, −2.2).

**Full observability mainly shortens episodes and removes much of the role asymmetry.** Complete object knowledge shortens episodes by 43% at the object-only baseline for both planners (Table 6) while leaving the Llama 70B’s Success Rate essentially unchanged (76.9% against 77.5% at its best partial-observability setting), and about 94% (Llama 8B) and 86% (Llama 70B) of Extraneous Effort remains: most redundant work is a coordination failure that object knowledge does not fix. OAS keeps its coordination benefits, lowering Extraneous Effort and Extra Planning, but loses its task-quality benefit, which under partial observability partly compensated for missing world knowledge. The human’s completion lag halves for the Llama 70B (+8.1→+3.9 points) and reverses for the Llama 8B (+14.5→−1.8), so much of it was an exploration artifact of partial observability, whereas the robot’s execution advantage persists in all eight full-observability runs.

## 5 Limitations

Our experiments run in the Habitat 3.0 simulator on the PARTNR validation set with privileged perception and oracle skills, which may limit generalization to real deployments with perception error, actuation noise, and imperfect state estimation. We report a single run per configuration (Section 4.1), so small differences, such as the Success Rate gap between OAS-only and OAS + KE for the strongest planners, should not be over-interpreted. The Llama 70B planner is 4-bit quantized, so its results may not transfer to the full-precision model. Exact reproducibility of the API-accessed planners is outside our control: the providers determine which model version is served under a given identifier, how requests are routed and executed, and how long a model remains available, so repeated queries with identical prompts and settings may not return identical outputs over time (Appendix A.1). The API-accessed planners cannot use PARTNR’s grammar-constrained decoding (Appendix A.1.2), so prediction accuracy is not perfectly comparable across model families, although the context-dependent pattern of the differences argues against decoding alone explaining them. The study covers four planners and two-agent teams, and the full-observability control covers only the two Llama planners. The simulated human is itself an LLM agent, so the role asymmetries we report describe an LLM–LLM pair rather than an LLM–person pair. Because the PARTNR role prompts make the human the task giver and the robot the task receiver, these asymmetries are also not attributable to the action spaces alone. Several interpretations, notably the attribution-failure reading of ungrounded KE and the arity-based account of the parameter asymmetry, are consistent with the data but are not tested directly by our design. Next-action prediction is a behavioral proxy for partner modeling: we do not probe beliefs, false beliefs, or other mental-state attributions, so our results bear on the behavioral consequences usually attributed to ToM rather than on ToM itself. Finally, several design choices bound the interpretation: (i) OAS observation also merges the partner’s current percepts on co-location (Section 3.2), so part of its effect may come from shared perception rather than partner state; (ii) enabling OANA adds the partner’s action descriptions to the prompt as well as the prediction instruction; (iii) predictions are scored at the class level after masking instance IDs (Section 3.3); and (iv) both OAS observation and OANA prediction lengthen the prompt, so

---

their effects are confounded with added context length and may scale differently on longer-horizon or more constrained tasks.

## 6 Conclusions

We presented a controlled study of coordination between two embodied LLM agents in PARTNR, introducing Decoupled Partial Observability and manipulating Other Agent’s State (OAS) observation, object Knowledge Exchange (KE), and Other Agent’s Next Action (OANA) prediction independently across a full factorial grid and four planners, with coordination metrics reported per agent as well as for the team. The two context channels act on different axes: OAS mainly improves task quality and KE mainly improves efficiency, and they are largely redundant for strong planners but complementary for the weakest, for which object knowledge without partner state is the worst configuration we measure. Shared context thus helps only a planner that can make use of it, and, if our attribution reading is right, tagging shared observations with their source is a cheap and concrete design choice for real systems, since KE carries no sender identity. OANA prediction lowers the Success Rate in nineteen of the twenty model×context pairs we ran; its accuracy gains concentrate in the coarse action type, and part of that accuracy reflects recognizing the partner’s ongoing action. Even with complete world knowledge, the Llama planners name their partner’s next action type wrongly more than half the time. The two agents occupy distinguishable roles under Decoupled Partial Observability: the robot usually finishes its share first, while the simulated human keeps working afterwards and determines when the episode ends, and shared context improves throughput chiefly by shortening this final, human-only period. Embodied LLM agents therefore coordinate better when partner state is supplied in a structured way, but our evidence indicates that they do so mainly by reacting to that state rather than by anticipating the partner, and the partner modeling that does occur is shallow and planner-dependent, with agent-level asymmetries that appear only under lenient scoring. For claims about Theory of Mind in LLMs, this argues for behavioral tests that separate recognition from anticipation and anticipation from its use. Three methodological cautions follow: (i) in our data, partner-mention frequency tracks task success only when it is spontaneous; (ii) prediction accuracy that credits a partner’s ongoing action conflates recognition with anticipation; and (iii) team-averaged metrics conceal role asymmetries that per-agent reporting makes visible.

### Broader Impact Statement

This work studies coordination between two LLM agents in simulation; no human participants or personal data are involved, and the simulated “human” is itself an LLM agent. Its most direct application is assistive household robotics, where our results suggest concrete design choices: giving a robot its partner’s observable state and shared object knowledge improves coordination, whereas instructing it to predict its partner’s next action lowered task success in nearly all settings. Two risks follow. First, a robot acting on an incorrect prediction of what a person will do may leave tasks undone or get in the person’s way; deployed systems should therefore treat partner predictions as hypotheses to be confirmed by observation before acting, and should be evaluated with per-agent metrics, since our role analysis shows that a robot that completes many subtasks can still leave the person the last and hardest part of the work. Second, partner-state observation and object knowledge exchange involve monitoring people and their belongings at home. This can be limited by restricting observation to coarse, task-relevant states such as those we use (e.g., *Walking in hallway\_1*), processing them on the device without storing them, gating them by co-location as in our design, and making them visible to and switchable by the user.

## References

- Christopher Amato, George Konidaris, Leslie P. Kaelbling, and Jonathan P. How. Modeling and Planning with Macro-Actions in Decentralized POMDPs. *Journal of Artificial Intelligence Research*, 64:817–859, 2019. doi: 10.1613/jair.1.11418.
- Ariel Flint Ashery, Luca Maria Aiello, and Andrea Baronchelli. Emergent social conventions and collective bias in LLM populations. *Science Advances*, 11(20):eadu9368, 2025. doi: 10.1126/sciadv.adu9368.

- 
- Alessio Buscemi, Daniele Proverbio, Alessandro Di Stefano, The Anh Han, German Castignani, and Pietro Liò. Strategic Communication and Language Bias in Multi-Agent LLM Coordination. In *Multi-disciplinary Trends in Artificial Intelligence (MIWAI 2025)*, volume 16355 of *Lecture Notes in Computer Science*, pp. 289–301. Springer, 2026. doi: 10.1007/978-981-95-4963-4\_24.
- Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John M. Turner, Eric Undersander, and Tsung-Yen Yang. PARTNR: A Benchmark for Planning and Reasoning in Embodied Multi-agent Tasks. In *The Thirteenth International Conference on Learning Representations*, 2025.
- Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable Multi-Robot Collaboration with Large Language Models: Centralized or Decentralized Systems? In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4311–4317, 2024. doi: 10.1109/icra57147.2024.10610676.
- Logan Cross, Violet Xiang, Agam Bhatia, Daniel L. K. Yamins, and Nick Haber. Hypothetical Minds: Scaffolding Theory of Mind for Multi-Agent Tasks with Large Language Models. In *The Thirteenth International Conference on Learning Representations*, 2025.
- J. de Curtò and I. de Zarzà. LLM-Driven Social Influence for Cooperative Behavior in Multi-Agent Systems. *IEEE Access*, 13:44330–44342, 2025. doi: 10.1109/access.2025.3548451.
- Alex DeWeese and Guannan Qu. Locally Interdependent Multi-Agent MDP: Theoretical Framework for Decentralized Agents with Dynamic Dependencies. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10687–10709. PMLR, 2024.
- Andrew Estornell, Jean-François Ton, Yuanshun Yao, and Yang Liu. ACC-Collab: An Actor-Critic Approach to Multi-Agent LLM Collaboration. In *The Thirteenth International Conference on Learning Representations*, 2025.
- Kanishk Gandhi, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah D. Goodman. Understanding Social Reasoning in Language Models with Language Models. In *Advances in Neural Information Processing Systems 36 (Datasets and Benchmarks Track)*, 2023.
- Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. Grammar-Constrained Decoding for Structured NLP Tasks without Finetuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10932–10952, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.674.
- Marc Glocker, Peter Hönig, Matthias Hirschmanner, and Markus Vincze. LLM-Empowered Embodied Agent for Memory-Augmented Task Planning in Household Robotics. In *Proceedings of the Austrian Robotics Workshop (ARW 2025)*, pp. 55–60, 2025. doi: 10.34749/3061-0710.2025.9.
- Yuling Gu, Oyvind Tafjord, Hyunwoo Kim, Jared Moore, Ronan Le Bras, Peter Clark, and Yejin Choi. SimpleToM: Exposing the Gap between Explicit ToM Inference and Implicit ToM Application in LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026.
- Shanshan Han, Qifan Zhang, Yuhang Yao, Weizhao Jin, Zhaozhuo Xu, and Chaoyang He. LLM Multi-Agent Systems: Challenges and Open Problems. *arXiv*, abs/2402.03578v1, 2024. doi: 10.48550/arxiv.2402.03578.
- Mukul Khanna, Yongsan Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X Chang, and Manolis Savva. Habitat Synthetic Scenes Dataset (HSSD-200): An analysis of 3D scene scale and realism tradeoffs for ObjectGoal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16384–16393, 2024.
- Michal Kosinski. Evaluating large language models in theory of mind tasks. *Proceedings of the National Academy of Sciences*, 121(45):e2405460121, 2024. doi: 10.1073/pnas.2405460121.

- 
- Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia P. Sycara. Theory of Mind for Multi-Agent Collaboration via Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 180–192, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.13.
- Peihan Li, Zijian An, Shams Abrar, and Lifeng Zhou. Large language models for multi-robot systems: a survey. *Autonomous Robots*, 50(3):30, 2026. doi: 10.1007/s10514-026-10257-4.
- William Li, Lei Hamilton, Kaise Al-natour, and Sanjeev Mohindra. Evaluation of Habitat Robotics using Large Language Models. In *2025 IEEE High Performance Extreme Computing Conference (HPEC)*, pp. 1–8, 2025. doi: 10.1109/hpec67600.2025.11196444.
- Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1, 2024. doi: 10.1007/s44336-024-00009-2.
- Thomas W. Malone and Kevin Crowston. The Interdisciplinary Study of Coordination. *ACM Computing Surveys*, 26(1):87–119, 1994. doi: 10.1145/174666.174668.
- Abhijnan Nath, Carine Graff, and Nikhil Krishnaswamy. Collaborate, Deliberate, Evaluate: How LLM Alignment Affects Coordinated Multi-Agent Outcomes. In *Proceedings of the 25th International Conference on Autonomous Agents and Multiagent Systems*, pp. 3377–3379, Paphos, Cyprus, 2026. IFAAMAS. Extended abstract.
- Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P How, and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2681–2690. PMLR, 2017.
- Sabrina Patania, Luca Annese, Anita Pellegrini, Silvia Serino, Anna Lambiase, Luca Pallonetto, Silvia Rossi, Simone Colombani, Tom Foulsham, Azzurra Ruggeri, and Dimitri Ognibene. PerspAct: Enhancing LLM Situated Collaboration Skills through Perspective Taking and Active Vision. *arXiv*, abs/2511.08098, 2025. doi: 10.48550/arxiv.2511.08098.
- Aske Plaat, Max J. van Duijn, Niki van Stein, Mike Preuss, Peter van der Putten, and Kees Joost Batenburg. Agentic Large Language Models, a Survey. *Journal of Artificial Intelligence Research*, 84, 2025. doi: 10.1613/jair.1.18675.
- Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, et al. Habitat 3.0: A co-habitat for humans, avatars and robots. In *The Twelfth International Conference on Learning Representations*, 2024.
- SeungWon Seo, SooBin Lim, SeongRae Noh, Haneul Kim, and HyeongYeop Kang. From Assumptions to Actions: Turning LLM Reasoning into Uncertainty-Aware Planning for Embodied Agents. In *The Fourteenth International Conference on Learning Representations*, 2026.
- Natalie Shapira, Mosh Levy, Seyed Hossein Alavi, Xuhui Zhou, Yejin Choi, Yoav Goldberg, Maarten Sap, and Vered Shwartz. Clever Hans or Neural Theory of Mind? Stress Testing Social Reasoning in Large Language Models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2257–2273, St. Julian’s, Malta, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.138.
- Winnie Street, John Oliver Siy, Geoff Keeling, Adrien Baranes, Benjamin Barnett, Michael McKibben, Tatenda Kanyere, Alison Lentz, Blaise Agüera y Arcas, and Robin I. M. Dunbar. LLMs achieve adult human performance on higher-order theory of mind tasks. *Frontiers in Human Neuroscience*, 19:1633272, 2025. doi: 10.3389/fnhum.2025.1633272.
- Patrick Taillandier, Jean-Daniel Zucker, Arnaud Grignard, Benoit Gaudou, Nghi Quang Huynh, and Alexis Drogoul. Integrating LLM in Agent-Based Social Simulation: Opportunities and Challenges. *arXiv*, abs/2507.19364v1, 2025. doi: 10.48550/arxiv.2507.19364.

- 
- Bhaskar Trivedi and Manfred Huber. An Intention-Aware Agent Framework for Multi-Agent Decentralized Partially Observable Environments. *The International FLAIRS Conference Proceedings*, 38(1), 2025. doi: 10.32473/flairs.38.1.138972.
- Weizheng Wang, Ike Obi, and Byung-Cheol Min. Multi-Agent LLM Actor-Critic Framework for Social Robot Navigation. *arXiv*, abs/2503.09758, 2025a. doi: 10.48550/arxiv.2503.09758.
- Yuanfei Wang, Fangwei Zhong, Jing Xu, and Yizhou Wang. ToM2C: Target-oriented Multi-agent Communication and Cooperation with Theory of Mind. In *The Tenth International Conference on Learning Representations*, 2022.
- Zhimin Wang, Shaokang He, Duo Wu, Jinghe Wang, Linjia Kang, Jing Yu, and Zhi Wang. CoBel-World: Harnessing LLM Reasoning to Build a Collaborative Belief World for Optimizing Embodied Multi-Agent Collaboration. *arXiv*, abs/2509.21981v1, 2025b. doi: 10.48550/arxiv.2509.21981.
- Isadora White, Kolby Nottingham, Ayush Maniar, Max Robinson, Hansen Lillemark, Mehul Maheshwari, Lianhui Qin, and Prithviraj Ammanabrolu. Collaborating Action by Action: A Multi-agent LLM Framework for Embodied Reasoning. *arXiv*, abs/2504.17950, 2025. doi: 10.48550/arxiv.2504.17950.
- 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, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, Qi Zhang, and Tao Gui. The rise and potential of large language model based agents: a survey. *Science China Information Sciences*, 68(2):121101, 2025. doi: 10.1007/s11432-024-4222-0.
- Bo Yang, Jiaxian Guo, Yusuke Iwasawa, and Yutaka Matsuo. Large Language Models as Theory of Mind Aware Generative Agents with Counterfactual Reflection. *arXiv*, abs/2501.15355, 2025. doi: 10.48550/arxiv.2501.15355.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023.
- Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building Cooperative Embodied Agents Modularly with Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024a.
- Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. Exploring Collaboration Mechanisms for LLM Agents: A Social Psychology View. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14544–14607, Bangkok, Thailand, 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.782.

## A Appendix

This appendix supplements the main paper. Appendix A.1 details the agents’ action space, its generation, and the implementation; Appendix A.2 explains the action comparison protocol; Appendix A.3 gives the complete metrics and extended analyses behind Sections 4.2–4.4 under Decoupled Partial Observability; Appendix A.4 analyzes task performance and coordination under full observability; Appendix A.5 evaluates partner prediction under the strict next-action target and under full observability; and Appendix A.6 shows the prompts used in the experiments.

### A.1 Implementation details for the ReAct agents

Most of our experiments with the Llama models were conducted locally on four NVIDIA A100 GPUs, each with 64 GB of VRAM, while a smaller portion used three NVIDIA A40 GPUs with 48 GB of VRAM each.

For the 4-bit quantized Llama-3.1-70B-Instruct planner, we were able to run 2 and 1 parallel processes on these respective setups, whereas for the Llama-3.1-8B-Instruct planner, we used 5 and 3 parallel processes to complete the 1,000 validation episodes via Hugging Face Transformers inference. We used Gemini 3.1 Flash Lite and DeepSeek V3.2 through the OpenRouter API (model identifiers: google/gemini-3.1-flash-lite, deepseek/deepseek-v3.2). Some later runs used vLLM instead of Hugging Face Transformers for inference, with the same model weights and decoding settings.

Due to limited hardware and CPU hour availability, some experiments had to be split across these two types of resources, which operate at different speeds, so the runtimes below are not directly comparable across configurations. According to PARTNR’s *runtime* metric, the fastest configuration averaged 299 seconds per episode (DeepSeek V3.2 baseline experiment without OANA prediction), while the slowest required 1008 seconds per episode on average (Llama-3.1-70B-Instruct, Decoupled Partial Observability without OAS and with OANA prediction). We also observed that with OANA prediction, episodes generally required longer *runtime* (9% to 60% more), partly due to additional logging and metric computation and partly because more tokens are generated. In the case of OANA prediction, more tokens are generated at the replannings (with a 250 limit) because of the reasoning about both the other agent’s next action and the agent’s own action, as well as predicting OANA itself. Furthermore, the instruction prompt becomes longer and more complex, and incorporating OAS observations adds even more input tokens.

Despite this increase in runtime, Table 3 shows that OANA prediction often reduces the number of planning cycles and, in some cases, also decreases the number of simulation steps.

### A.1.1 Skill API Library

Below is the set of API libraries of action skill tools shared across all experiments that we use from PARTNR (Chang et al., 2025), which enable the agents to take actions in the simulator. The robot agent has no access to the state-changing tools (Clean, Fill, Pour, PowerOff, PowerOn).

- **Clean:** Used for cleaning an object.
- **Close:** Used for closing an articulated entity.
- **Explore:** Search a specific room by visiting various receptacles or furniture in that room.
- **Fill:** Used for filling an object.
- **Navigate:** Used for navigating to an entity.
- **Open:** Used for opening an articulated entity.
- **Pick:** Used for picking up an object; the agent cannot hold more than one object at a time.
- **Place:** Used for placing an object on a target location; requires object name, furniture name, and spatial relation (“on” or “within”). The object must already be held. Optionally supports placement `next_to` a reference object (which must already be on the target furniture). API: `Place[<object>, <relation>, <furniture>, <constraint>, <reference>]`, with `None` when unused.
- **Pour:** Used for pouring from one container to another; pours from the container currently held by the agent.
- **PowerOff:** Used for turning off a powered object.
- **PowerOn:** Used for turning on a powered object.
- **Rearrange:** Used for moving an object to a target location; requires object name, furniture name, and spatial relation (“on” or “within”), and automatically performs pick and place. Optionally supports `next_to` with a reference object (must already be on target furniture). API: `Rearrange[<object>, <relation>, <furniture>, <constraint>, <reference>]`, with `None` when unused.
- **Wait:** Used to make the agent stay idle for some time.
- **Done:** Used to indicate that the agent has finished the task.

### A.1.2 Constraint Generation

We adopt PARTNR’s constrained generation approach, following Geng et al. (2023), which restricts token sampling to sequences consistent with the specified grammar. For each Llama LLM call, we construct a grammar that permits only valid tool calls over observed entities. The base grammar used in all experiments is shown below, while the rules for objects, furniture, and rooms are dynamically updated based on the agent’s current world graph. This ensures that skills are invoked only for entities known to the agent. In the construction, the human agent’s grammar includes all the actions, but the robot agent’s grammar does not contain the object-state modification actions.

#### Tool Call Grammar

```

root ::= Navigate | Pick | Place | Open | Close | Rearrange | PowerOn | PowerOff |
Clean | Fill | Pour | Explore | Wait | Done
Navigate ::= "Navigate[" nav_target "]"
Pick ::= "Pick[" object "]"
Place ::= "Place[" object ", " WS spatial_relation ", " WS furniture ", " WS
((spatial_constraint " ", " WS obj_or_furniture) | (("none" | "None") WS " ", " WS ("none"
| "None")))) "]"
Open ::= "Open[" furniture "]"
Close ::= "Close[" furniture "]"
Rearrange ::= "Rearrange[" object ", " WS spatial_relation ", " WS furniture ", " WS
((spatial_constraint " ", " WS obj_or_furniture) | (("none" | "None") WS " ", " WS ("none"
| "None")))) "]"
PowerOn ::= "PowerOn[" obj_or_furniture "]"
PowerOff ::= "PowerOff[" obj_or_furniture "]"
Clean ::= "Clean[" obj_or_furniture "]"
Fill ::= "Fill[" object "]"
Pour ::= "Pour[" object "]"
Explore ::= "Explore[" room "]"
Wait ::= "Wait[]"
Done ::= "Done[]"
nav_target ::= (furniture | room | object)
object ::= "object_1" | "object_2" | ...
obj_or_furniture ::= (furniture | object)
furniture ::= "furniture_1" | "furniture_2" | ...
room ::= "room_1" | "room_2" | ...
spatial_constraint ::= "next_to"
spatial_relation ::= "on" | "within"
free_text ::= ['. ., !a-zA-Z_0-9']*
WS ::= [ ]*

```

**For OANA prediction**, the same rules apply, except that agents use only object classes rather than object classes with object instance IDs (see Section 3.3). Accordingly, the *object* rule changed to *object\_class* ::= "*object\_class\_1*" / "*object\_class\_2*" / ..., and all occurrences of *object* in other rules are replaced with *object\_class* as well. The set of object classes is available from the start of each episode based on the simulation information; thus, predictions can be made at any replanning containing the object classes, whereas specific object instances (with IDs) are only known once they are explored and observed.

Note: This type of Constraint Generation could only be applied to the locally run Llama models, where the output tensors were available, whereas for the models accessed via API, this type of Constraint Generation is not applicable. Consequently, these models (Gemini 3.1 Flash Lite and DeepSeek V3.2) can hallucinate entities as action parameters that are not in the world graph of the agents.

---

## A.2 Action comparison

As stated in Section 3.3 and shown in Table 2, we evaluated the OANA prediction across multiple levels of abstraction. The three examples below illustrate the comparison levels. Note that for the comparison, we ignore the instance IDs of the entities.

### Example 1:

OANA prediction: Place[cup,on,table,None,None]

Target action: Place[cup,on,table,next\_to,plate]

### Example 2:

OANA prediction: Navigate[book]

Target action: Pick[book]

### Example 3:

OANA prediction: Explore[bedroom]

Target action: Explore[livingroom]

### Action comparison levels:

- Action Type  
Example 1 - match: Place  $\leftrightarrow$  Place  
Example 2 - no match: Navigate  $\leftrightarrow$  Pick  
Example 3 - match: Explore  $\leftrightarrow$  Explore
- Principal Parameter  
Example 1 - match: cup  $\leftrightarrow$  cup  
Example 2 - match: book  $\leftrightarrow$  book  
Example 3 - no match: bedroom  $\leftrightarrow$  livingroom
- All Parameters  
Example 1 - no match: (cup,on,table,None,None)  $\leftrightarrow$  (cup,on,table,next\_to,plate)  
Example 2 - match: (book)  $\leftrightarrow$  (book)  
Example 3 - no match: (bedroom)  $\leftrightarrow$  (livingroom)
- Action Type + Principal Parameter  
Example 1 - match: (Place,cup)  $\leftrightarrow$  (Place,cup)  
Example 2 - no match: (Navigate,book)  $\leftrightarrow$  (Pick,book)  
Example 3 - no match: (Explore,bedroom)  $\leftrightarrow$  (Explore,livingroom)
- Action Type + All Parameters  
Example 1 - no match:  
(Place,cup,on,table,None,None)  $\leftrightarrow$  (Place,cup,on,table,next\_to,plate)  
Example 2 - no match: (Navigate,book)  $\leftrightarrow$  (Pick,book)  
Example 3 - no match: (Explore,bedroom)  $\leftrightarrow$  (Explore,livingroom)
- Principal Parameter if Action Type is matched  
Example 1 - match: (cup)  $\leftrightarrow$  (cup)  
Example 2 - no count: no action type match  
Example 3 - no match: (bedroom)  $\leftrightarrow$  (livingroom)

Table 3: **Task performance and coordination metrics under Decoupled Partial Observability (full version of Table 1)** with different Other Agent’s State (OAS) observation, Knowledge Exchange (KE) in the same room, and Other Agent’s Next Action (OANA) prediction settings, using four different LLM planners for both agents. Results show that OAS observation mainly improves task quality while KE mainly improves execution efficiency, that the best coordination is reached with both enabled and without OANA prediction, and that OANA prediction raises partner mentioning but lowers Success Rate in all but one configuration. SR: Success Rate, PC: Percent Complete, OA Mention: Other-Agent Mention Ratio, R/H: Robot/Human agent. Subgroup bests are underlined; overall bests are bolded for each model.

(a) Llama-3.1-8B-Instruct

| OAS | KE | OANA | Sim<br>Steps ↓ | SR<br>%↑ | PC<br>%↑ | Plan.<br>(R / H) ↓ | Cycles<br>(R / H) ↓ | OA Mention<br>(R / H) %↑ | Exploration<br>Eff. (R/H) ↓ | Offload<br>%↑ | Extra<br>Effo. %↓ | Extra Plan.<br>(R / H) %↓ | Action Success<br>(R / H) %↑ |
|-----|----|------|----------------|----------|----------|--------------------|---------------------|--------------------------|-----------------------------|---------------|-------------------|---------------------------|------------------------------|
| ✗   | ✗  | ✗    | 4214           | 60.9     | 77.1     | 24.2 / 23.2        | 9.8 / 14.6          | 1348 / 1795              | 57.0                        | 62.3          | 23.2 / 37.7       | 70.2 / 69.4               |                              |
| ✗   | ✗  | ✓    | 4243           | 55.0     | 73.0     | 21.2 / 20.9        | 85.5 / 88.7         | 1590 / 2244              | 60.7                        | 62.0          | 17.7 / 39.8       | 70.9 / 66.0               |                              |
| ✗   | ✓  | ✗    | 4914           | 54.3     | 72.9     | 27.7 / 25.1        | 9.5 / 17.3          | 1300 / 1741              | 56.9                        | 69.6          | 26.5 / 32.3       | 64.5 / 62.4               |                              |
| ✗   | ✓  | ✓    | 4121           | 56.2     | 72.9     | 20.7 / 20.4        | 87.7 / 92.1         | 1487 / 2047              | 58.9                        | 62.7          | 16.1 / 35.5       | 71.6 / 66.0               |                              |
| ✓   | ✗  | ✗    | 4022           | 58.8     | 74.7     | 22.1 / 21.1        | 20.5 / 32.7         | 1793 / 2166              | 55.1                        | 58.5          | 20.5 / 33.8       | 72.0 / 66.4               |                              |
| ✓   | ✗  | ✓    | 4017           | 56.6     | 74.2     | 19.8 / 19.4        | 74.7 / 81.2         | 1670 / 2200              | 56.9                        | 58.5          | 20.0 / 39.3       | 73.6 / 69.6               |                              |
| ✓   | ✓  | ✗    | 3781           | 62.3     | 76.8     | 21.5 / 19.8        | 21.0 / 30.9         | 1779 / 2148              | 55.5                        | 58.5          | 20.6 / 32.0       | 72.0 / 68.3               |                              |
| ✓   | ✓  | ✓    | 3890           | 57.2     | 74.5     | 19.2 / 18.7        | 75.6 / 80.8         | 1653 / 2138              | 56.4                        | 59.1          | 18.3 / 35.6       | 73.4 / 70.1               |                              |

(b) 4-bit quantized Llama-3.1-70B-Instruct

| OAS | KE | OANA | Sim<br>Steps ↓ | SR<br>%↑ | PC<br>%↑ | Plan.<br>(R / H) ↓ | Cycles<br>(R / H) ↓ | OA Mention<br>(R / H) %↑ | Exploration<br>Eff. (R/H) ↓ | Offload<br>%↑ | Extra<br>Effo. %↓ | Extra Plan.<br>(R / H) %↓ | Action Success<br>(R / H) %↑ |
|-----|----|------|----------------|----------|----------|--------------------|---------------------|--------------------------|-----------------------------|---------------|-------------------|---------------------------|------------------------------|
| ✗   | ✗  | ✗    | 3056           | 76.0     | 86.9     | 15.7 / 15.1        | 13.3 / 17.2         | 909 / 1100               | 52.4                        | 39.4          | 17.3 / 25.4       | 79.9 / 80.0               |                              |
| ✗   | ✗  | ✓    | 3648           | 72.9     | 85.6     | 17.0 / 15.8        | 71.3 / 61.1         | 1312 / 1455              | 52.4                        | 43.5          | 23.0 / 26.4       | 77.1 / 75.8               |                              |
| ✗   | ✓  | ✗    | 3041           | 76.9     | 87.6     | 15.6 / 14.7        | 13.1 / 16.8         | 1239 / 1365              | 53.0                        | 43.0          | 15.5 / 22.2       | 78.4 / 79.3               |                              |
| ✗   | ✓  | ✓    | 3125           | 72.1     | 84.6     | 16.0 / 14.0        | 56.2 / 54.7         | 1215 / 1416              | 52.9                        | 42.1          | 18.2 / 17.7       | 78.9 / 78.1               |                              |
| ✓   | ✗  | ✗    | 3425           | 76.2     | 86.5     | 15.6 / 13.8        | 16.9 / 26.6         | 1220 / 1550              | 55.3                        | 36.5          | 13.9 / 21.2       | 81.7 / 81.7               |                              |
| ✓   | ✗  | ✓    | 3019           | 72.5     | 84.3     | 14.4 / 13.5        | 74.9 / 64.3         | 1239 / 1419              | 53.3                        | 38.0          | 15.7 / 19.8       | 80.5 / 79.9               |                              |
| ✓   | ✓  | ✗    | 3095           | 77.5     | 87.6     | 15.4 / 13.1        | 18.1 / 24.7         | 1180 / 1456              | 55.8                        | 36.5          | 13.9 / 17.7       | 81.5 / 81.4               |                              |
| ✓   | ✓  | ✓    | 2849           | 72.8     | 84.6     | 13.9 / 12.8        | 75.3 / 65.4         | 1178 / 1371              | 52.9                        | 36.8          | 12.9 / 18.9       | 82.1 / 80.7               |                              |

- All Parameters if Action Type is matched  
Example 1 - no match: (cup,on,table,None,None) ↔ (cup,on,table,next\_to,plate)  
Example 2 - no count: no action type match  
Example 3 - no match: (bedroom) ↔ (livingroom)

### A.3 Additional Results under Decoupled Partial Observability

This section gives the complete metrics behind Sections 4.2–4.4 and the analyses that the main text summarizes.

#### A.3.1 Task Outcomes and Coordination

**What the OANA prediction buys is planner-family dependent.** For the Llama models, OANA reduces Planning Cycles for both agents in six of eight settings (8B in all four, e.g. 24.2/23.2→21.2/20.9), yet Simulation Steps do not fall correspondingly for the 8B (4214→4243) and Percent Complete declines in seven of the eight settings (it is unchanged for the 8B with KE only), indicating fewer but longer plans and earlier termination rather than more efficient planning. For Gemini the opposite holds: Planning Cycles rise slightly in all four settings (+0.5 on average) while Exploration Efficiency improves consistently (2177/2262→1735/1653, 1744/1798→1649/1634, 1666/1772→1438/1424, 1617/1638→1368/1359), so prediction makes this planner commit to objects sooner at a small planning cost. DeepSeek pays the highest overhead (Planning Cycles rise

Table 3: (continued)

(c) Gemini 3.1 Flash Lite

| OAS | KE | OANA | Sim<br>Steps ↓     | SR<br>%↑           | PC<br>%↑           | Plan.<br>(R / H) ↓      | Cycles<br>(R / H) ↓ | OA Mention<br>(R / H) %↑  | Exploration<br>Eff. (R/H) ↓ | Offload<br>%↑      | Extra<br>Effo. %↓         | Extra Plan.<br>(R / H) %↓ | Action Success<br>(R / H) %↑ |
|-----|----|------|--------------------|--------------------|--------------------|-------------------------|---------------------|---------------------------|-----------------------------|--------------------|---------------------------|---------------------------|------------------------------|
| ✗   | ✗  | ✗    | <u>4118</u>        | <u>80.4</u>        | <u>89.8</u>        | <u>10.2 / 11.1</u>      | 30.4 / 37.8         | 2177 / 2262               | 54.7                        | 44.3               | <u>26.9 / 36.3</u>        | <u>81.1 / 80.1</u>        |                              |
| ✗   | ✗  | ✓    | 4304               | 76.0               | 88.6               | 10.8 / 12.4             | <u>91.7 / 91.1</u>  | <u>1735 / 1653</u>        | <u>55.1</u>                 | <u>43.5</u>        | 33.6 / 38.0               | 80.5 / 75.7               |                              |
| ✗   | ✓  | ✗    | <u>3081</u>        | <u>79.9</u>        | <u>89.4</u>        | <u>8.2 / 8.7</u>        | 24.3 / 31.6         | 1744 / 1798               | 54.6                        | 38.2               | <u>16.4 / 25.1</u>        | 85.1 / <u>84.6</u>        |                              |
| ✗   | ✓  | ✓    | 3416               | 75.8               | 88.3               | 8.7 / 8.9               | <u>92.3 / 90.2</u>  | <u>1649 / 1634</u>        | <u>55.3</u>                 | <u>35.1</u>        | <u>23.6 / 22.5</u>        | <u>85.2 / 83.5</u>        |                              |
| ✓   | ✗  | ✗    | <u>3193</u>        | <b><u>81.2</u></b> | <u>89.8</u>        | <u>8.3 / 8.9</u>        | 27.7 / 38.2         | 1666 / 1772               | <u>55.7</u>                 | 33.0               | <u>16.6 / 26.7</u>        | <b><u>85.6 / 85.5</u></b> |                              |
| ✓   | ✗  | ✓    | 3453               | 79.1               | 88.6               | 8.8 / 9.4               | <u>89.8 / 88.3</u>  | <u>1438 / 1424</u>        | 55.3                        | <u>32.2</u>        | <u>21.7 / 22.5</u>        | 85.0 / 83.4               |                              |
| ✓   | ✓  | ✗    | <b><u>2911</u></b> | <u>81.1</u>        | <b><u>90.3</u></b> | <b><u>8.0 / 8.5</u></b> | 27.5 / 36.5         | 1617 / 1638               | <b><u>56.4</u></b>          | 33.3               | <b><u>14.9 / 20.5</u></b> | <b><u>85.6 / 84.2</u></b> |                              |
| ✓   | ✓  | ✓    | 3107               | 78.6               | 89.1               | 8.5 / 8.7               | <u>90.6 / 88.5</u>  | <b><u>1368 / 1359</u></b> | 55.8                        | <b><u>32.1</u></b> | <u>20.0 / 16.5</u>        | 84.2 / 83.7               |                              |

(d) DeepSeek V3.2

| OAS      | KE       | OANA     | Sim<br>Steps ↓ | SR<br>%↑    | PC<br>%↑    | Plan.<br>(R / H) ↓        | Cycles<br>(R / H) ↓      | OA Mention<br>(R / H) %↑  | Exploration<br>Eff. (R/H) ↓ | Offload<br>%↑ | Extra<br>Effo. %↓         | Extra Plan.<br>(R / H) %↓ | Action Success<br>(R / H) %↑ |
|----------|----------|----------|----------------|-------------|-------------|---------------------------|--------------------------|---------------------------|-----------------------------|---------------|---------------------------|---------------------------|------------------------------|
| <i>✗</i> | <i>✗</i> | <i>✗</i> | <u>3857</u>    | <u>82.3</u> | <u>91.0</u> | <u>13.3</u> / <u>13.3</u> | 39.1 / 43.6              | <u>1458</u> / <u>1584</u> | 54.8                        | 38.2          | <u>20.3</u> / <u>28.0</u> | 89.5 / 88.4               |                              |
| <i>✗</i> | <i>✗</i> | ✓        | 5486           | 75.2        | 87.1        | 18.9 / 16.6               | <u>99.9</u> / <u>100</u> | 1641 / 1987               | <u>55.8</u>                 | <u>35.0</u>   | 39.0 / 38.4               | <u>92.1</u> / <u>89.7</u> |                              |
| <i>✗</i> | ✓        | <i>✗</i> | <u>2951</u>    | 82.2        | 90.6        | <b>12.2</b> / <u>11.4</u> | 35.9 / 40.9              | <u>1380</u> / <u>1471</u> | 53.8                        | 34.8          | <u>16.0</u> / <u>19.3</u> | 91.9 / 92.3               |                              |
| <i>✗</i> | ✓        | ✓        | 4249           | 75.6        | 87.4        | 16.4 / 13.4               | <u>100</u> / <u>100</u>  | 1521 / 1709               | <b>57.9</b>                 | <u>31.9</u>   | 36.4 / 29.4               | <u>93.7</u> / 91.5        |                              |
| ✓        | <i>✗</i> | <i>✗</i> | <u>3112</u>    | <b>83.6</b> | <b>91.5</b> | <u>12.6</u> / <u>11.5</u> | 49.5 / 52.6              | <u>1237</u> / <u>1396</u> | 54.6                        | 28.1          | <u>15.1</u> / <u>17.9</u> | 92.7 / <b>92.8</b>        |                              |
| ✓        | <i>✗</i> | ✓        | 3927           | 78.4        | 88.8        | 14.8 / 12.5               | <u>100</u> / <u>99.9</u> | 1341 / 1571               | <u>55.5</u>                 | <b>26.7</b>   | 30.9 / 23.6               | <u>93.2</u> / 90.3        |                              |
| ✓        | ✓        | <i>✗</i> | <b>2794</b>    | <u>82.4</u> | <u>90.5</u> | <b>12.2</b> / <b>11.0</b> | 48.6 / 51.4              | <b>1214</b> / <b>1381</b> | 52.5                        | 27.5          | <b>13.6</b> / <b>17.1</b> | 92.8 / <u>92.6</u>        |                              |
| ✓        | ✓        | ✓        | 3551           | 80.0        | 89.4        | 14.0 / 11.4               | <u>100</u> / <u>100</u>  | 1328 / 1462               | <u>53.6</u>                 | 27.0          | 30.5 / 19.8               | <b>93.9</b> / 91.3        |                              |

by up to 5.6, and Extra Planning nearly doubles, from 20.3% to 39.0% for the robot without shared context), but its robot Action Success improves under prediction in every setting (as does the Llama 8B’s), peaking at 93.9%; its human agent declines in three of the four. Prediction thus improves the quality of individual actions for some planners while degrading the joint outcome.

**Model capacity sets the baseline quality of coordination and determines how safely additional context can be added.** Extraneous Effort at each model’s best setting decreases monotonically with planner strength (Llama 8B 58.5%, Llama 70B 36.5%, Gemini 32.1%, DeepSeek 26.7%), while Task Offloading stays within a narrow 52.4–60.7% band across all 32 runs: the performance differences come from better temporal alignment and less duplicated work, not from shifting the workload between the agents. Exploration Efficiency separates the two families most clearly. For the Llama planners, the settings without OAS give the fastest first object pickup (70B 909/1100 at the baseline; 8B 1300/1741 with KE only, its worst Success Rate configuration), and richer context delays the first interaction; for Gemini and DeepSeek, the richest settings give the fastest pickup (1368/1359 and 1214/1381). Weaker planners, therefore, behave greedily and early, and additional interaction signals can hurt them, whereas stronger planners convert the same signals into faster search and better task outcomes alike.

**OANA turns successes into ordinary failures, not near-misses.** The Percent Complete–Success Rate gap widens under prediction in 15 of 16 model×context pairs, but this follows arithmetically from the lower Success Rate: more episodes fail, and each failed episode still contributes partial progress. The informative quantity is the average completion of a failed episode,  $(PC - SR)/(100 - SR)$ , which lies between 38% and 53% in all 32 runs and rises under prediction in only 8 of the 16 pairs. Prediction, therefore, converts some would-be successes into failures that, like failures without prediction, complete about half of the task’s propositions. Across the no-prediction settings, the same quantity is 38–41% for the Llama 8B and 43–49% for the three stronger planners, so weaker planners also leave more of a task undone when they fail, not merely fail more often.

---

**Object knowledge without partner state is a weak lever on redundant work and a counterproductive one for the Llama planners.** Adding KE to the object-only baseline raises Extraneous Effort for both Llama planners (8B 62.3%→69.6%, 70B 39.4%→43.0%), and although it lowers Extraneous Effort for Gemini (44.3%→38.2%) and DeepSeek (38.2%→34.8%), OAS alone lowers it substantially further in both cases (33.0% and 28.1%). OAS is thus the dominant lever on redundant rearrangement for every model, and KE without OAS is at best a weak substitute and at worst counterproductive. This generalizes the 8B failure discussed in Section 4.2: synchronizing object knowledge tells an agent that an object exists and where it is, but not that the partner is already acting on it, so the newly visible objects become candidate targets for both agents simultaneously. The 70B exhibits the same direction as the 8B in Extraneous Effort (though not in Extra Planning) without a Success Rate cost (76.0%→76.9%), which suggests the mechanism is general and its severity is governed by planner capacity.

**Prediction partially repairs the ungrounded-KE failure mode, which supports the attribution account.** Adding OANA to the Llama 8B’s pathological KE-only configuration reduces Simulation Steps from 4914 to 4121, Planning Cycles from 27.7/25.1 to 20.7/20.4, and Extraneous Effort from 69.6% to 62.7%, while raising Action Success from 64.5/62.4% to 71.6/66.0%. It is also the only setting in which prediction raises Success Rate (54.3%→56.2%, with Percent Complete unchanged at 72.9%), although by less than the standard error of a difference between two single runs (about 2.2 points). Under the attribution account, this follows naturally: KE supplies object knowledge that the 8B cannot assign to an owner, and OANA forces the missing assignment step by requiring an explicit hypothesis about what the partner will do with those objects. Prediction here substitutes, imperfectly, for the partner-state signal that OAS would otherwise provide.

**Spontaneous partner reasoning tracks task success across models; mandated partner reasoning does not.** In the Decoupled baseline, the Other-Agent Mention Ratio rises with planner strength (robot: 9.8%, 13.3%, 30.4%, 39.1% for Llama 8B, 70B, Gemini, DeepSeek; human: 14.6%, 17.2%, 37.8%, 43.6%) and orders the models identically to Success Rate (60.9%, 76.0%, 80.4%, 82.3%). Stronger planners attend to the partner more often without being asked to. OANA raises the same metric to 55–92% for the Llama and Gemini planners and saturates it at essentially 100% for DeepSeek, yet Success Rate falls in fifteen of the sixteen cases (the exception is the 8B with KE only, discussed above). The metric is therefore informative as an unprompted behavioral signal and uninformative once it is instructed, which we take as evidence that endogenous partner modeling and prompt-compliant partner mentioning are distinct phenomena that this metric cannot separate on its own.

**Planning volume is not a proxy for planning competence.** Gemini attains 81.1% Success Rate using 8.0/8.5 planning cycles per episode, while Llama 8B attains 62.3% using 21.5/19.8, and DeepSeek attains 82.4% using an intermediate 12.2/11.0. Expressed as simulator steps per high-level action at each model’s OAS + KE setting, Gemini issues roughly 176 steps per action (both agents’ planning cycles pooled) against 92 for the 8B, 109 for the 70B, and 120 for DeepSeek. Gemini therefore commits to substantially longer-horizon actions and revises them rarely, whereas the 8B replans frequently over short horizons and converts that additional deliberation into neither fewer simulator steps nor higher success. Frequent replanning in these settings is a symptom of unstable commitment or a poor grasp of the environment and its actions, not careful planning.

**KE is the cheaper intervention and OAS is the more accurate one.** For all three stronger planners, KE alone yields fewer Simulation Steps than OAS alone (Llama 70B 3041 vs. 3425, Gemini 3081 vs. 3193, DeepSeek 2951 vs. 3112), while OAS alone yields the higher Success Rate in three of four models (Gemini 81.2% vs. 79.9%, DeepSeek 83.6% vs. 82.2%, Llama 8B 58.8% vs. 54.3%; the Llama 70B reverses this at 76.2% vs. 76.9%). Since object knowledge synchronization requires only shared world-graph state while partner-state observation requires continuous monitoring of the collaborator, this matters practically: deployments constrained to one channel should choose based on whether latency or task correctness is the binding constraint, and the two channels are close enough to be redundant for strong planners that implementing both is justified mainly by the modest additional efficiency it buys.

---

### A.3.2 Role Asymmetry Between the Agents

**Division of labor is nearly invariant to every intervention; what changes is timing.** Task Offloading stays within 52.4–60.7% across all 32 runs, a range of eight points, while Success Rate varies from 54.3% to 83.6% and Simulation Steps from 2794 to 5486 over the same runs. No setting causes agents to renegotiate who does what to any meaningful degree. The small movements that do occur are mostly toward the robot: OANA raises its share in 10 of 16 pairs, most for the 8B baseline (57.0%→60.7%) and DeepSeek with KE only (53.8%→57.9%), leaves it unchanged for the 70B baseline, and lowers it slightly, by at most 2.9 points, for the 70B with any shared context and for Gemini with OAS enabled. The performance differences reported in Section 4.2 are consequently produced by better temporal alignment, fewer duplicated actions, and earlier object acquisition, not by a different allocation of work.

**The robot reaches its first object interaction earlier than the human in 28 of 32 runs.** The mean Exploration Efficiency gap (human minus robot) is 482 simulator steps for the Llama 8B, 205 for the Llama 70B, 180 for DeepSeek, and 18 for Gemini. The four exceptions are exactly Gemini’s four OANA configurations, where the human is marginally faster (1368/1359 at OAS + KE). The robot’s action repertoire consists entirely of primitives that begin with locating and picking an object, so every robot plan converges on a pickup, whereas a substantial share of the human’s available actions operate on objects already in place and do not require one. The gap is largest for the weakest planner and nearly vanishes for Gemini, indicating that it reflects how strongly a planner is steered by the shape of its action list rather than by a hard environmental constraint.

**The human is the trailing agent: it performs a higher rate of replanning than the robot after the other has stopped taking action in 25 of 32 runs.** Averaged over all configurations, the Extra Planning gap (human minus robot) is +15.4 points for the Llama 8B, +4.9 for the 70B, +4.3 for Gemini, and −1.0 for DeepSeek (the average gap in extra planning cycles shows the same pattern: +3.0 for the 8B, +0.5 for the 70B, +0.6 for Gemini, and −0.6 for DeepSeek). Restricting to the no-prediction settings, the richest shared context compresses this gap in every model: from +14.5 to +11.4 points for the 8B, +8.1 to +3.8 for the 70B, +9.4 to +5.6 for Gemini, and +7.7 to +3.5 for DeepSeek as the configuration moves from object-only observation to OAS + KE. This asymmetry coexists with Task Offloading above 50% in all 32 runs, meaning the robot completes more subtasks by count while nonetheless going idle first. Together with the prompted roles, a natural reading is that the human’s exclusive state-modification actions sit at the ends of dependency chains: an object usually must be transported to the correct receptacle before it needs to be cleaned, filled or switched on, so the human’s distinctive contributions are serialized behind the robot’s rearrangements and tend to be the last work remaining. Subtask-count offloading consequently overstates how much the robot relieves its partner, because the robot’s contribution comes early in the episode and the human completes the remaining work.

**Shared context helps the team primarily by shortening the trailing window.** Extra Planning falls for the human from 37.7% to 32.0% (8B), 25.4% to 17.7% (70B), 36.3% to 20.5% (Gemini) and 28.0% to 17.1% (DeepSeek) between the object-only baseline and OAS + KE, in every case a larger absolute reduction than for the robot over the same comparison. Because episode length is determined by the later-finishing agent, improvements concentrated on the trailing agent translate directly into the Simulation Step reductions reported in Section 4.2. This also explains why OAS and KE improve efficiency more reliably than they improve Success Rate: they compress idle and redundant time at the end of the episode without changing which agent is capable of finishing the task.

**Prediction reverses the direction of the lag for strong planners and amplifies it for the weakest one.** Under OANA, DeepSeek’s robot Extra Planning roughly doubles (20.3%→39.0% at baseline, 13.6%→30.5% at OAS + KE), turning the gap negative (human minus robot) in all four contexts (−0.6, −7.0, −7.3, −10.7 points); Gemini shows the same reversal in two of four contexts, and the Llama 70B a marginal one with KE only (−0.5 points). For the 8B, the gap instead widens from +14.5 to +22.1 points at baseline and from +11.4 to +17.3 at OAS + KE. Since the robot must predict from the human’s 14 action types while the human predicts from the robot’s 9, prediction imposes an asymmetric burden that strong planners absorb as additional robot-side deliberation after the partner has finished, while the weakest planner instead leaves its human agent with more remaining work.

**Without prediction, the human reasons about the partner more often; with prediction, the robot does so with the stronger planners.** The Other-Agent Mention Ratio is higher for the human in all 16 no-prediction runs, across every model and context (for example 21.0/30.9% for the Llama 8B and 27.5/36.5% for Gemini at OAS + KE). Enabling OANA flips the ordering for Gemini in all four contexts (90.6/88.5% at OAS + KE) and for the Llama 70B in all four (75.3/65.4% at OAS + KE), saturates it for DeepSeek near 100% on both sides, and leaves it unchanged for the 8B. The spontaneous pattern is consistent with dependence driving attention: the agent whose work is blocked by the partner has more reason to model the partner. The induced pattern is consistent with prediction difficulty driving reasoning length: the robot faces the larger prediction space and spends more of its reasoning trace on the partner once prediction is mandatory. That these two orderings point in opposite directions is a further reason to treat mention frequency as a behavioral signal rather than a quality measure.

**The robot executes actions more reliably than the human, and prediction widens that gap.** Robot Action Success is at least as high as human Action Success in 28 of 32 runs, with the largest margins for the 8B (up to 5.6 points, e.g., 72.0% vs. 66.4% with OAS only) and near parity for the 70B and DeepSeek. Under OANA, the change in Action Success is less favorable for the human than for the robot in 13 of 16 model×context pairs, for instance, −0.6 vs. −4.4 points for Gemini at baseline and +0.5 vs. −2.5 for DeepSeek with OAS only. The human’s larger action space contains the actions with the most restrictive preconditions, since state modification requires the object to be present, accessible, and in the appropriate configuration, so mispredicting the partner has more ways to invalidate a human action than a robot action. Anticipatory execution is thus riskier for whichever agent holds the precondition-heavy repertoire.

**Planning-cycle asymmetry is planner-specific rather than environment-imposed.** The robot issues more planning cycles than the human in all eight 8B runs, all eight 70B runs, and seven of eight DeepSeek runs, but fewer in all eight Gemini runs (8.0/8.5 at OAS + KE). Gemini is also the planner with the smallest exploration gap between the agents. The same asymmetric action spaces therefore produce opposite planning-cycle orderings depending on the planner, which indicates that role specialization here is a joint property of the task structure and the model, not a mechanical consequence of the environment. Reporting per-agent rather than averaged metrics is necessary to observe this; agent-averaged planning cycles are nearly identical for Gemini and would conceal the reversal entirely.

### A.3.3 Partner Prediction

**Combining OAS observation with object KE gives the best Action Type prediction accuracy for every model and both agents.** The OAS + KE configuration gives the top Action Type accuracy in all eight model×agent combinations, strictly in seven and tied in the eighth (Llama 8B 37.9/34.2%, Llama 70B 55.5/55.8%, Gemini 52.4/52.7%, DeepSeek 49.0/54.7%, R/H), and the top accuracy at every other unconditional abstraction level for the Llama 70B and DeepSeek planners. Relative to the object-only baseline, the gains are substantial for the three stronger planners: +17.0/+14.0 points for the Llama 70B, +14.4/+16.3 for Gemini, and +21.1/+23.5 for DeepSeek on Action Type. Access to the partner’s state and to a synchronized object world graph therefore both supply information that predicts the partner’s next action, which is not obvious a priori: neither channel transmits intentions, plans, or language, only observed states and object locations.

**OAS observation is the stronger single channel for three of the four planners, but the two channels are substantially redundant rather than additive.** Taking Action Type as the reference metric, the sum of the two single-channel gains exceeds the gain from enabling both in seven of the eight model×agent combinations, and often by a wide margin: for Gemini, the separate gains total +22.1/+27.8 points while the combination delivers only +14.4/+16.3, a realization of roughly 60%. DeepSeek realizes about 83% (+25.1/+28.6 separately vs. +21.1/+23.5 together), the Llama 70B 87% and 64% (+19.5/+21.8 vs. +17.0/+14.0), and only the 8B robot is marginally super-additive (+5.8 vs. +6.1). Partner state and shared object knowledge thus overlap in what they reveal about the partner’s next action, which is consistent with their mechanisms: observing that the partner is *Picking apple\_0* and learning that *apple\_0* has appeared in the shared world graph or its position has changed, despite the agent not interacting with the object, are two routes to the same inference. The practical consequence mirrors Section 4.2: implementing the second channel buys considerably less than implementing the first.

---

**Shared context tells an agent what its partner will do far more than which object it will act on.** Averaged over the four planners, moving from the object-only baseline to OAS + KE improves Action Type accuracy by +14.7 points for the robot and +14.6 for the human, but improves the conditional principal-parameter accuracy by only +4.2 and +1.0 points, respectively. The asymmetry is visible in the extreme cases: Gemini’s human agent improves its Action Type accuracy from 36.4% to 52.7% while its conditional principal-parameter accuracy moves from 52.6% to 51.5%, so essentially the entire joint improvement for that agent comes from the type factor. Even at the best-performing cell in the whole table (70B with OAS + KE), roughly a third of correctly typed predictions still carry the wrong principal parameter (67.2/64.1% conditional accuracy) and about 40% carry a wrong full parameter set (61.4/59.4%). The residual difficulty in partner modeling, therefore, lies in the arguments rather than the verbs, and it is the component least responsive to the interventions we study.

**For the smallest planner, richer context improves the type term and degrades the parameter term, leaving joint accuracy nearly unchanged.** The Llama 8B is the only model whose conditional accuracies fall clearly below baseline for both agents (elsewhere the largest drop is 2.0 points, Gemini’s human with OAS): enabling OAS drops conditional principal-parameter accuracy from 54.2% to 49.2% (robot) and conditional all-parameter accuracy from 49.4% to 45.5%, with OAS + KE no better. Because its Action Type accuracy simultaneously rises from 31.8% to 36.1% (37.9% with OAS+KE), the two factors nearly cancel, and the joint action-type-plus-all-parameters accuracy moves only from 15.7% to 16.4% (16.9%). The decomposition reveals what the joint metric hides: the Llama 8B does extract partner information from OAS and convert it into better coarse predictions, but the additional prompt content displaces the object-level reasoning needed to fill in arguments, so the newly correctly typed predictions are disproportionately ones it cannot complete. This is the prediction-side counterpart of the capacity sensitivity documented in Section 4.2, and it recurs with the opposite polarity: in task solving, KE without OAS was the 8B’s pathological configuration, whereas in prediction, KE is the only channel that helps it (+1.1/+3.7 points of conditional principal-parameter accuracy) and OAS is the one that hurts.

**Whether object knowledge alone suffices depends on the planner, not on the task.** For Gemini, KE alone recovers or exceeds the entire benefit of OAS alone on Action Type (+11.4 vs. +10.7 for the robot, +15.3 vs. +12.5 for the human, i.e. ratios above one), and for DeepSeek, it recovers roughly 60% of it. For the Llama planners, the same channel recovers less: about a third on the robot side and about half on the human side for the 70B (+5.1 vs. +14.4 and +7.8 vs. +14.0, ratios of 0.35 and 0.56), and about a third for the 8B (at most 0.37). A plausible account is that a synchronized object graph supports prediction by self-projection: once both agents hold the same beliefs about where objects are, an agent can predict the partner by asking what it would itself do next, and this route is used more by planners that re-derive the partner’s situation than by those that read off the partner’s state. The Llama planners appear to rely mainly on the direct partner-state signal, whereas the API-accessed planners, Gemini in particular, exploit common ground. This distinction is invisible in the joint metrics and only emerges once the KE-only condition is run.

**Parameters are the binding constraint on partner prediction, and the errors are asymmetric.** Among predictions that name every parameter correctly, the action type is also correct 84.8% of the time on average across the 32 model×context×agent cells (range 77.4–92.9%), whereas among predictions with the correct action type, only 52.2% carry a fully correct parameter set (range 44.5–61.4%). Getting the arguments right, therefore, almost implies getting the verb right, while the converse fails roughly half the time. The residual error mode is nonetheless informative: about one in seven parameter-correct predictions still attaches the wrong verb, so agents that have correctly identified the objects their partner is engaged with can still misjudge the manipulation being performed on them. We note that the direction of this asymmetry follows arithmetically from the action type being the easier marginal (Bayes relates the two conditionals by the ratio of the marginals), so it quantifies the size of the parameter bottleneck rather than establishing it independently.

**Task-solving strength does not predict partner-modeling strength, and for the strongest planner, the two rankings are close to inverted.** DeepSeek V3.2 achieves the best Success Rate, Percent Complete, and Action Success of all four planners in Table 3, yet at the object-only baseline, its robot is the weakest partner predictor in the table, below even the 8B’s (27.9% vs. 31.8% Action Type). Conversely, the 4-bit 70B is the third-best task solver but the best predictor under full shared context (55.5/55.8%). Prediction

accuracy and task competence are thus dissociable: a planner can be highly effective at deciding its own next action while modeling its partner poorly, as expected if strong task performance comes mainly from individual planning rather than from coordination. This also supplies independent confirmation of the argument in Section 4.2 that partner-mention frequency is not a quality measure. DeepSeek mentions the other agent in essentially 100% of replanning steps under OANA in every context, the highest rate of any planner, while producing some of the least accurate partner predictions.

**Within the API-accessed planners, prediction accuracy tracks the size of the success-rate penalty that OANA imposes.** For DeepSeek, the correspondence is monotone across all four contexts: as baseline Action Type accuracy rises through 27.9%, 37.1%, 43.8% and 49.0% for no context, KE, OAS and OAS + KE, the Success Rate cost of enabling prediction falls through  $-7.1$ ,  $-6.6$ ,  $-5.2$  and  $-2.4$  points. Gemini follows the ordering only loosely: its penalty tracks whether OAS is available ( $-4.4$  and  $-4.1$  points without OAS,  $-2.1$  and  $-2.5$  with it) more closely than accuracy, since KE-only reaches higher Action Type accuracy than OAS-only (49.4/51.7% vs. 48.7/48.9%) yet costs about twice as much. This quantifies the mechanism proposed in Section 4.2: agents act on their predictions, so inaccurate anticipation translates into premature commitment and lost task propositions, and improving the prediction reduces the damage. We note that the relationship does not extend to the Llama planners, whose penalties do not shrink as accuracy rises, so we present it as an account of the API-accessed models rather than a general law; disentangling it properly would require intervening on prediction quality directly, which our design does not do.

**Constrained decoding cannot explain the cross-family differences in action type, but may contribute to those in parameters.** The locally run Llama planners use PARTNR’s grammar-constrained decoding (Appendix A.1.2), which restricts every predicted call to a valid action and, for OANA predictions, restricts objects to the classes present in the scene from the start of the episode; the API-accessed planners have neither restriction. On action type the grammar confers little, and the data agree: with KE only, the unconstrained Gemini still predicts the partner’s action type better than the constrained Llama 70B (49.4/51.7% against 43.6/49.6%). On parameters, the picture differs: the Llama 70B’s conditional principal-parameter accuracy (62.2–67.6%) exceeds Gemini’s (50.6–58.3%) and DeepSeek’s (50.2–62.2%) in every context and for both agents, the direction expected if knowing the scene’s object classes helps fill in arguments, although the equally constrained Llama 8B (49.2–57.2%) shows that grammar alone is not sufficient. We therefore treat action-type comparisons across model families as reliable and parameter-level comparisons as favoring the Llama planners.

The two agents face structurally different prediction problems. The human predicts over the robot’s 9 action types, while the robot predicts over the human’s 14, which include the object state manipulation actions (*Clean*, *Fill*, *Pour*, *PowerOn*, and *PowerOff*) absent from the robot’s repertoire. Both run the same planner, so systematic differences between the R and H columns of Table 2 reflect this asymmetry and the prompted roles (Section 4.3) rather than model capability. Under the union target, we find a double dissociation; Appendix A.5 tests which parts of it survive the strict target and full observability.

**Under the union target, the human predicts the partner’s action type more accurately.** For the Llama 70B and DeepSeek, the human leads in all four contexts ( $+0.3$  to  $+6.0$  and  $+3.3$  to  $+5.7$  points, respectively), and Gemini leads on the human side in three of four. The effect is in the direction implied by the action-space sizes. For the Llama 70B, it also shrinks once OAS is available, from  $+3.3$  at baseline ( $+6.0$  with KE only) to  $+2.9$  with OAS and  $+0.3$  with OAS + KE, indicating that richer context partially compensates the robot for its larger hypothesis space rather than helping both agents equally. This lead is not stable across scoring and regime: under the strict next-action target, it disappears in the partial regime, where the robot matches or leads in 14 of 16 cells, and for the two Llama planners, it reappears under full observability (Appendix A.5). The pattern suggests that the smaller hypothesis space helps only once object uncertainty is removed, and that under partial observability, part of the union-target lead comes from recognizing the robot’s ongoing navigation.

**The robot predicts parameters more accurately once the action type is right, and this advantage appears only with shared context.** At the object-only baseline, the conditional principal-parameter gaps are negligible in every model ( $-0.7$ ,  $+0.8$ ,  $+0.9$  and  $-1.5$  points H–R for 8B, 70B, Gemini and DeepSeek). Enabling either channel opens a robot-side lead that persists in all three stronger planners:  $-3.1$  to  $-4.2$

Table 4: **Task solving performance and coordination metrics under full observability setting** with different Other Agent’s State (OAS) observation and Other Agent’s Next Action (OANA) prediction settings. The 4-bit quantized Llama-3.1-70B-Instruct and Llama-3.1-8B-Instruct serve as planners for both agents. MS: model size, SR: Success Rate, PC: Percent Complete, OA Mention: Other-Agent Mention Ratio, R/H: Robot/Human agent. Subgroup bests are underlined; overall bests are bolded.

| MS  | OAS      | OANA     | Sim<br>Steps ↓ | SR<br>%↑    | PC<br>%↑    | Plan.<br>(R / H) ↓        | Cycles<br>(R / H) ↓       | OA Mention<br>(R / H) %↑ | Exploration<br>Eff. (R/H) ↓ | Offload<br>%↑ | Extra<br>Effo. %↓  | Extra<br>(R / H) %↓       | Plan.<br>(R / H) %↓ | Action Success<br>(R / H) %↑ |
|-----|----------|----------|----------------|-------------|-------------|---------------------------|---------------------------|--------------------------|-----------------------------|---------------|--------------------|---------------------------|---------------------|------------------------------|
| 8B  | <b>X</b> | <b>X</b> | 2383           | <u>64.7</u> | <u>79.5</u> | 20.6 / 19.0               | 11.6 / 15.4               | 421 / 455                | 51.1                        | 60.0          | 22.7 / 20.9        | <u>73.0</u> / <u>68.9</u> |                     |                              |
|     | <b>X</b> | ✓        | 3113           | 58.5        | 72.9        | 17.2 / 17.6               | 67.5 / 74.9               | 1054 / 1306              | 54.6                        | 61.0          | 19.1 / 28.9        | 71.0 / 65.4               |                     |                              |
|     | ✓        | <b>X</b> | <u>2001</u>    | 62.6        | 78.5        | 18.1 / 18.3               | 23.5 / 32.4               | 498 / <u>452</u>         | 51.5                        | <u>54.8</u>   | 18.9 / <u>16.8</u> | 70.9 / 62.4               |                     |                              |
|     | ✓        | ✓        | 2890           | 60.3        | 75.3        | <u>17.1</u> / <u>15.7</u> | <u>72.1</u> / <b>77.9</b> | 1053 / 1296              | <b>56.7</b>                 | 57.0          | <u>13.8</u> / 24.3 | 72.4 / 68.6               |                     |                              |
| 70B | <b>X</b> | <b>X</b> | 1742           | <b>76.9</b> | <b>87.7</b> | 12.8 / 12.5               | 14.3 / 15.5               | 387 / 428                | 49.1                        | 35.6          | 11.5 / 15.4        | 84.0 / <b>83.3</b>        |                     |                              |
|     | <b>X</b> | ✓        | 1825           | 75.3        | 86.6        | 13.4 / 12.0               | 50.8 / 52.9               | 456 / 539                | <u>51.8</u>                 | 37.7          | 14.5 / 13.4        | 81.6 / 79.7               |                     |                              |
|     | ✓        | <b>X</b> | <b>1611</b>    | <b>76.9</b> | 87.3        | <b>12.1</b> / <b>11.6</b> | 19.8 / 21.8               | <b>385</b> / <b>411</b>  | 49.9                        | <b>31.5</b>   | 8.2 / <b>11.1</b>  | <b>84.2</b> / 82.9        |                     |                              |
|     | ✓        | ✓        | 1681           | 76.3        | 87.0        | <b>12.1</b> / <b>11.6</b> | <b>73.3</b> / 65.4        | 426 / 515                | 51.4                        | 35.5          | <b>7.7</b> / 11.4  | 81.2 / 79.5               |                     |                              |

points for the 70B,  $-5.7$  to  $-7.1$  for Gemini, and  $-2.0$  to  $-4.9$  for DeepSeek, with the same pattern at the all-parameter level. The likely source is the arity of the two action repertoires rather than the size of their parameter vocabularies, which the action grammar (Appendix A.1.2) makes comparable: *Navigate* ranges over furniture, rooms, and objects alike, and the human’s *Clean*, *PowerOn* and *PowerOff* range over furniture and objects alike. What differs is structure. The five action types available only to the human (*PowerOn*, *PowerOff*, *Clean*, *Fill*, *Pour*) all take a single argument, whereas *Place* and *Rearrange*, both in the robot’s nine-type repertoire, each take five slots comprising an object, a spatial relation, a furniture target, and an optional constraint with its reference entity. Conditional on a correct action type, the agent predicting the robot therefore draws from a pool in which the two compound actions carry proportionally more weight, and a single wrong slot is enough to fail the all-parameter criterion; the agent predicting the human additionally draws from five single-slot actions on which parameter prediction is a much simpler problem. This also offers a reading of why the gap is absent at baseline: without shared context neither agent can name the objects and receptacles involved, so both directions fail on the arguments regardless of arity, and the asymmetry only becomes expressible once OAS and KE make the single-argument cases nearly determined while leaving the compound cases underdetermined. Under the strict target, this robot-side lead persists clearly only for Gemini (Appendix A.5.1), so we offer the arity account as a hypothesis specific to union scoring.

**The smallest planner inverts the type asymmetry, which suggests it is not exploiting the smaller action space.** The 8B robot outpredicts the 8B human on Action Type in all four contexts ( $-2.2$  to  $-3.7$  points H–R), reversing the pattern seen in every stronger planner. A planner that reasons over the partner’s available actions should find the 9-action problem easier than the 14-action one. That the 8B does not suggest that its predictions are driven less by the structure of the partner’s action space than by surface features of the prompt, which is consistent with its failure to convert correct type predictions into correct arguments discussed above.

**The two asymmetries cancel, so the aggregate metrics look symmetric while the underlying competences are not.** On the strictest joint metric, action type plus all parameters, the H–R gap stays within  $\pm 3.3$  points in all 16 cells and within  $\pm 2.5$  for three of the four planners, even though its two factors reach  $+6.0$  points in one direction (Action Type, 70B with KE) and  $-7.0$  in the other (All Params if Type, Gemini with OAS). Reporting only the joint accuracies would therefore suggest that the two agents model each other about equally well, when in fact, under union scoring, the human is better at anticipating what its partner will do and the robot at anticipating what it will do it to. The decomposition into type and conditional parameter accuracy makes the asymmetry visible, and it is a reason to report conditional accuracies rather than joint accuracies alone in settings with heterogeneous action spaces.

Table 5: **Comparison of task performance and coordination with enabled OANA prediction** across different observability scenarios. OAS: Other Agent’s State, KE: Knowledge Exchange, SR: Success Rate, PC: Percent Complete, Plan. Cycles: Planning Cycles per agent (R: Robot / H: Human), OA Mention: Other-Agent Mention Ratio (R / H %). Subgroup bests are underlined, overall bests are bolded. Larger models and fuller observability generally lead to better task success and more efficient planning. Note: For full observability, object KE is not applicable.

| Model Size | Observability     | OAS | KE | OANA | Sim Steps ↓ | SR %↑       | PC %↑       | Plan. Cycles ↓ |             | OA Mention %↑ |             |
|------------|-------------------|-----|----|------|-------------|-------------|-------------|----------------|-------------|---------------|-------------|
|            |                   |     |    |      |             |             |             | R              | H           | R             | H           |
| 8B         | Decoupled Partial | ✗   | ✗  | ✓    | 4243        | 55.0        | 73.0        | 21.2           | 20.9        | 85.5          | 88.7        |
|            | Decoupled Partial | ✗   | ✓  | ✓    | 4121        | 56.2        | 72.9        | 20.7           | 20.4        | <b>87.7</b>   | <b>92.1</b> |
|            | Decoupled Partial | ✓   | ✗  | ✓    | 4017        | 56.6        | 74.2        | 19.8           | 19.4        | 74.7          | 81.2        |
|            | Decoupled Partial | ✓   | ✓  | ✓    | 3890        | 57.2        | 74.5        | 19.2           | 18.7        | 75.6          | 80.8        |
|            | Full              | ✗   | -  | ✓    | 3113        | 58.5        | 72.9        | 17.2           | 17.6        | 67.5          | 74.9        |
|            | Full              | ✓   | -  | ✓    | <u>2890</u> | <u>60.3</u> | <u>75.3</u> | <u>17.1</u>    | <u>15.7</u> | 72.1          | 77.9        |
| 70B        | Decoupled Partial | ✗   | ✗  | ✓    | 3648        | 72.9        | 85.6        | 17.0           | 15.8        | 71.3          | 61.1        |
|            | Decoupled Partial | ✗   | ✓  | ✓    | 3125        | 72.1        | 84.6        | 16.0           | 14.0        | 56.2          | 54.7        |
|            | Decoupled Partial | ✓   | ✗  | ✓    | 3019        | 72.5        | 84.3        | 14.4           | 13.5        | 74.9          | 64.3        |
|            | Decoupled Partial | ✓   | ✓  | ✓    | 2849        | 72.8        | 84.6        | 13.9           | 12.8        | <u>75.3</u>   | <u>65.4</u> |
|            | Full              | ✗   | -  | ✓    | 1825        | 75.3        | 86.6        | 13.4           | 12.0        | 50.8          | 52.9        |
|            | Full              | ✓   | -  | ✓    | <b>1681</b> | <b>76.3</b> | <b>87.0</b> | <b>12.1</b>    | <b>11.6</b> | 73.3          | <u>65.4</u> |

#### A.4 Evaluation under Full Observability

We additionally evaluate the effects of OAS observation and OANA prediction under PARTNR’s *Full* observability setting, where both agents use the complete global ground-truth world graph at all times. In this setting, at each replanning, beyond room and furniture information, agents observe all objects and their locations (textual description in the prompt), independent of their own positions. Consequently, explicit exploration is no longer required, although it remains available.

Since object information is globally accessible, the object KE is not applicable anymore, and the true OAS, if enabled, is also accessible continuously, irrespective of the agents’ relative locations (agents in different rooms no longer receive the “*Acting in another room*” observation). The OANA prediction follows the same protocol as in the partial observability setting (Section 3.3): at each replanning step, agents first reason about the partner’s and their own next actions and then predict the partner’s next action before selecting their own action.

##### A.4.1 Task Performance and Coordination under Full Observability

Table 4 reports the same metrics under PARTNR’s full observability setting, in which both agents hold complete and continuously updated object knowledge from the start of the episode. Object KE is not applicable here: full observability already gives every agent all object positions at all times, so it subsumes and strictly exceeds the co-location-gated synchronization that KE provides. The design therefore reduces to a  $2 \times 2$  grid of OAS observation and OANA prediction, run with the Llama-3.1-8B and 4-bit Llama-3.1-70B planners. Tables 5 and 6 place these runs alongside the corresponding Decoupled Partial Observability configurations with and without OANA prediction, respectively. Because the comparison spans only two planners from one family, we treat it as a controlled probe of what perceptual uncertainty contributes to the effects reported in Sections 4.2–4.3, not as an independent evaluation.

**Full observability roughly halves execution time while leaving task success essentially unchanged for the stronger planner.** At matched settings, Simulation Steps fall by 43% for both planners at the object-only baseline (Llama 8B 4214→2383, Llama 70B 3056→1742) and by 50% and 53% respectively with

Table 6: **Comparison of task performance and coordination with disabled OANA prediction** across different observability scenarios. OAS: Other Agent’s State, KE: Knowledge Exchange, SR: Success Rate, PC: Percent Complete, Plan. Cycles: Planning Cycles per agent (R: Robot / H: Human), OA Mention: Other-Agent Mention Ratio (R / H %). Subgroup bests are underlined, overall bests are bolded. Larger models and fuller observability generally lead to better task success and more efficient planning. Note: For full observability, object KE is not applicable.

| Model Size | Observability     | OAS | KE | OANA | Sim Steps ↓ | SR %↑       | PC %↑       | Plan. Cycles ↓ |             | OA Mention %↑ |             |
|------------|-------------------|-----|----|------|-------------|-------------|-------------|----------------|-------------|---------------|-------------|
|            |                   |     |    |      |             |             |             | R              | H           | R             | H           |
| 8B         | Decoupled Partial | ✗   | ✗  | ✗    | 4214        | 60.9        | 77.1        | 24.2           | 23.2        | 9.8           | 14.6        |
|            | Decoupled Partial | ✗   | ✓  | ✗    | 4914        | 54.3        | 72.9        | 27.7           | 25.1        | 9.5           | 17.3        |
|            | Decoupled Partial | ✓   | ✗  | ✗    | 4022        | 58.8        | 74.7        | 22.1           | 21.1        | 20.5          | <b>32.7</b> |
|            | Decoupled Partial | ✓   | ✓  | ✗    | 3781        | 62.3        | 76.8        | 21.5           | 19.8        | 21.0          | 30.9        |
|            | Full              | ✗   | -  | ✗    | 2383        | <u>64.7</u> | <u>79.5</u> | 20.6           | 19.0        | 11.6          | 15.4        |
|            | Full              | ✓   | -  | ✗    | <u>2001</u> | 62.6        | 78.5        | <u>18.1</u>    | <u>18.3</u> | <b>23.5</b>   | 32.4        |
| 70B        | Decoupled Partial | ✗   | ✗  | ✗    | 3056        | 76.0        | 86.9        | 15.7           | 15.1        | 13.3          | 17.2        |
|            | Decoupled Partial | ✗   | ✓  | ✗    | 3041        | 76.9        | 87.6        | 15.6           | 14.7        | 13.1          | 16.8        |
|            | Decoupled Partial | ✓   | ✗  | ✗    | 3425        | 76.2        | 86.5        | 15.6           | 13.8        | 16.9          | <u>26.6</u> |
|            | Decoupled Partial | ✓   | ✓  | ✗    | 3095        | <b>77.5</b> | 87.6        | 15.4           | 13.1        | 18.1          | 24.7        |
|            | Full              | ✗   | -  | ✗    | 1742        | 76.9        | <b>87.7</b> | 12.8           | 12.5        | 14.3          | 15.5        |
|            | Full              | ✓   | -  | ✗    | <b>1611</b> | 76.9        | 87.3        | <b>12.1</b>    | <b>11.6</b> | <u>19.8</u>   | 21.8        |

OAS enabled (4022→2001 and 3425→1611). Success Rate moves far less. The Llama 70B reaches 76.9% under full observability, which does not exceed the 77.5% it attains under Decoupled Partial Observability with OAS + KE, and its best Percent Complete is 87.7% against 87.6% in the partial regime, a difference well within the uncertainty of a single run. The Llama 8B does gain, from 62.3% to 64.7%, but 2.4 points of Success Rate is a modest return on the complete elimination of perceptual uncertainty. Task accuracy for these planners is therefore mostly not limited by perception: a perfect world model makes them much faster at the tasks they already solve but does not let them solve appreciably more.

**Continuous object knowledge improves speed but not accuracy.** The clearest comparison is the Llama 70B without OAS or prediction: KE-only under Decoupled Partial Observability yields 76.9% Success Rate, 87.6% Percent Complete and 3041 Simulation Steps, while full observability yields 76.9%, 87.7% and 1742 steps. Success Rate is identical, and Percent Complete differs by 0.1 points, while the episode is 43% shorter. Upgrading from intermittent, proximity-triggered object synchronization to permanent, complete knowledge thus converts entirely into execution efficiency. This supports the reading of KE in Section 4.2: KE mainly improves efficiency, and even complete object knowledge does not raise the accuracy ceiling.

**Exploration Efficiency collapses, which confirms what the metric measures in the partial regime.** First successful pickup arrives 57–79% earlier under full observability at every matched setting without prediction (Llama 70B baseline 909/1100→387/428, Llama 70B with OAS 1220/1550→385/411, Llama 8B baseline 1348/1795→421/455, Llama 8B with OAS 1793/2166→498/452). The magnitude is expected, since with all object positions known in advance, there is nothing to search for, and the residual 385–500 steps are navigation time rather than exploration. This confirms that the Exploration Efficiency differences reported in Section 4.3 and Appendix A.3.1 tracked search behavior and not, for example, differences in how quickly a planner commits to a first action.

**Redundant work is a coordination failure, not a perception failure: most extraneous effort survives perfect object knowledge.** Extraneous Effort at each planner’s best full-observability setting is 54.8% for the Llama 8B and 31.5% for the Llama 70B, against 58.5% and 36.5% at their best partial-observability settings. Removing all perceptual uncertainty therefore eliminates only about 4 and 5 points of redundant rearrangement, leaving roughly 94% and 86% of it in place. Agents with identical, complete, and

correct world models still perform a third to three-fifths of their rearrangements unnecessarily. Since neither agent can act on stale or wrong object information in this regime, the remaining redundancy must come from both agents independently selecting the same task-relevant object, a failure that partner-state access, not world knowledge, has to address.

**OAS observation keeps its coordination benefits under full observability but loses its task-quality benefits, and for the weakest planner it becomes net harmful.** With prediction disabled, enabling OAS still reduces Extraneous Effort for both planners (Llama 8B 60.0%→54.8%, Llama 70B 35.6%→31.5%) and still reduces Extra Planning for both agents (22.7/20.9%→18.9/16.8% and 11.5/15.4%→8.2/11.1%), with both reductions comparable to or larger than the corresponding ones in the partial regime. Success Rate, however, does not follow: it is unchanged for the Llama 70B (76.9% in both cases, with Percent Complete slipping 87.7%→87.3%) and falls for the Llama 8B (64.7%→62.6%), whose human agent also drops to 62.4% Action Success, the lowest value anywhere in Table 4. This happens even though OAS is more available here than in the partial regime: under full observability, the partner’s state is observed continuously rather than only on co-location (Appendix A.4), while the co-located percept sharing that OAS also provides (Section 3.2) becomes redundant, since every object is already known. Read together with Section 4.2, this separates two things OAS was doing at once. Its coordination contribution, telling an agent what its partner is occupied with, and hence what not to duplicate, is intrinsic and survives. Its contribution to task outcomes was substantially a matter of compensating for incomplete world knowledge, and once that gap is closed, the added prompt content is a cost that the 8B, in particular, cannot absorb.

**OANA prediction lowers Success Rate and Percent Complete in all four full-observability configurations, bringing the count to nineteen of twenty across both regimes (the exception being the partial-observability Llama 8B with KE only).** The penalties are −6.2 and −2.3 points of Success Rate for the Llama 8B without and with OAS, and −1.6 and −0.6 points for the Llama 70B, with Percent Complete moving in the same direction in every case and the Other-Agent Mention Ratio rising from 12–32% to 51–78%. That the effect holds without exception under perfect perception rules out one otherwise plausible account of it, namely that prediction hurts because agents anticipate partners whose situation they cannot see. Agents that know everything about the world, and that are asked to predict a partner acting in that same fully known world, still complete fewer task propositions than agents that simply act.

**The size of that penalty, however, depends on planner capacity.** The Llama 70B’s penalties under full observability (−1.6 and −0.6 points) are the smallest it incurs in any configuration in this paper, against −3.1 to −4.8 points across the four partial-observability settings, and its Simulation Step overhead falls to +83 and +70 steps (under 5%), against +592 at the partial-observability baseline. The Llama 8B shows no such relief: its penalties under full observability (−6.2, −2.3) match those in the corresponding partial-observability settings (−5.9, −2.2) almost exactly, and its Simulation Step overhead grows to +730 and +889 steps, a 31% and 44% increase on episodes that are already the shortest it produces. Applying the grounding argument of Section 4.2 to this setting, complete world knowledge makes prediction nearly free for the capable planner and no cheaper at all for the weak one.

**Under full observability, the exploration cost of prediction is pure deliberation overhead, and it is large for the smaller planner.** Enabling OANA delays the first pickup from 421/455 to 1054/1306 steps for the Llama 8B and from 387/428 to 456/539 for the Llama 70B. In the partial regime, such a delay is ambiguous between slower search and slower decision-making; here, there is nothing to search for, so the entire 2.5–2.9× increase for the 8B is time spent deliberating before acting. Prediction consumes a substantial share of this planner’s early decision-making without improving what it decides, which is consistent with its joint prediction accuracy barely moving with added context (Section 4.4).

**The human-trailing asymmetry is largely produced by partial observability rather than by the action spaces alone.** Defining the completion lag as human minus robot Extra Planning, as in Section 4.3, the no-prediction lag shrinks from +8.1 to +3.9 points at baseline and from +7.3 to +2.9 with OAS for the Llama 70B, and reverses outright for the Llama 8B, from +14.5 to −1.8 at baseline and from +13.3 to −2.1 with OAS. Task Offloading moves in the same direction, falling from 55.1–60.7% (8B) and 52.4–55.8% (70B) in the partial regime to 51.1–56.7% and 49.1–51.8%, respectively; the 70B’s share even falls below parity in both of its no-prediction settings (49.1% and 49.9%). The dependency-chain account of Section 4.3

Table 7: **Other Agent’s Next Action (OANA) prediction accuracy (%)** across action abstraction levels for robot (R) and human (H) agents under **Decoupled Partial** observability. The **target action is the next action** of the other agent. Results highlight that OAS observation, alone or combined with KE, improves prediction for the three stronger planners, while the Llama 8B gains almost nothing at the strictest level.

| Model                        | OAS | KE | Action Type (R/H) ↑       | Principal Par. (R/H) ↑    | A. Type+Princ. Par. (R/H) ↑ | All Params (R/H) ↑        | A. Type+All Params (R/H) ↑ | Principal Par. if Type (R/H) ↑ | All Params if Type (R/H) ↑ |
|------------------------------|-----|----|---------------------------|---------------------------|-----------------------------|---------------------------|----------------------------|--------------------------------|----------------------------|
| Llama-3.1-8B-Instruct        | ✗   | ✗  | 23.1 / 20.7               | 18.0 / 15.4               | 11.5 / 10.4                 | 11.4 / 9.6                | 10.1 / 8.8                 | 50.0 / 50.1                    | <u>43.6</u> / 42.4         |
|                              | ✗   | ✓  | 24.0 / 21.6               | <u>19.2</u> / <u>17.2</u> | <u>12.2</u> / 12.2          | <u>12.0</u> / <u>10.8</u> | <u>10.4</u> / 9.9          | <u>50.8</u> / <u>56.4</u>      | 43.3 / <u>45.5</u>         |
|                              | ✓   | ✗  | 25.8 / <u>23.6</u>        | 15.2 / 15.0               | 11.3 / <u>12.3</u>          | 10.9 / <u>10.8</u>        | 10.0 / <u>10.2</u>         | 43.8 / 51.8                    | 38.7 / 43.1                |
|                              | ✓   | ✓  | <u>26.2</u> / 23.4        | 15.8 / 14.8               | 11.9 / <u>11.4</u>          | 11.1 / 10.4               | 9.9 / 9.6                  | 45.5 / 48.5                    | 37.9 / 41.1                |
| Llama-3.1-70B-Instruct 4-bit | ✗   | ✗  | 27.5 / 26.9               | 26.2 / 26.2               | 17.3 / 18.5                 | 19.0 / 19.9               | 15.4 / 16.5                | 63.0 / 68.7                    | 56.1 / 61.4                |
|                              | ✗   | ✓  | 30.7 / 31.2               | 30.1 / 26.9               | 21.5 / 21.4                 | 22.1 / 20.7               | 18.6 / 18.4                | 69.8 / 68.5                    | 60.6 / 58.9                |
|                              | ✓   | ✗  | 34.7 / <b>33.3</b>        | 30.7 / 28.7               | 25.0 / <u>24.5</u>          | 24.8 / 24.2               | 21.7 / <b>22.2</b>         | 72.2 / 73.6                    | 62.6 / 66.8                |
|                              | ✓   | ✓  | <b>35.4</b> / 32.6        | <b>32.7</b> / <b>28.8</b> | <b>26.3</b> / 24.4          | <b>26.2</b> / <b>24.3</b> | <b>23.2</b> / 21.9         | <b>74.4</b> / <b>75.0</b>      | <b>65.5</b> / <b>67.3</b>  |
| Gemini 3.1 Flash Lite        | ✗   | ✗  | 23.1 / 21.0               | 16.8 / 13.4               | 11.3 / 9.7                  | 12.8 / 10.4               | 9.9 / 8.6                  | 48.8 / 46.0                    | 43.0 / 41.1                |
|                              | ✗   | ✓  | <u>30.4</u> / <u>29.4</u> | 23.5 / 18.8               | 17.6 / <u>14.6</u>          | 19.7 / <u>15.9</u>        | 16.3 / <u>13.5</u>         | 58.1 / 49.7                    | 53.7 / 46.0                |
|                              | ✓   | ✗  | 27.6 / 25.9               | 24.6 / 18.8               | 16.7 / 12.9                 | 20.0 / 14.3               | 15.8 / 11.9                | 60.6 / 49.8                    | 57.2 / 45.8                |
|                              | ✓   | ✓  | 30.3 / 27.3               | <u>27.3</u> / <u>20.1</u> | <u>18.7</u> / 13.7          | <u>22.1</u> / 15.4        | <u>17.4</u> / 12.7         | <u>61.8</u> / <u>50.3</u>      | <u>57.6</u> / <u>46.4</u>  |
| DeepSeek V3.2                | ✗   | ✗  | 18.7 / 18.5               | 15.3 / 15.0               | 10.0 / 9.6                  | 11.4 / 10.9               | 8.9 / 8.4                  | 53.6 / 51.8                    | 47.5 / 45.5                |
|                              | ✗   | ✓  | 24.4 / 26.4               | 20.8 / 21.8               | 15.1 / 15.3                 | 16.0 / 16.7               | 13.6 / 13.9                | 61.9 / 57.8                    | 55.5 / 52.6                |
|                              | ✓   | ✗  | 28.0 / 27.1               | 24.5 / 23.5               | 19.0 / 18.0                 | 20.7 / 19.4               | 17.8 / 16.7                | <u>67.9</u> / 66.2             | <u>63.6</u> / 61.7         |
|                              | ✓   | ✓  | <u>31.5</u> / <u>31.5</u> | <u>27.7</u> / <u>26.9</u> | <u>21.3</u> / <u>21.2</u>   | <u>22.9</u> / <u>22.0</u> | <u>19.8</u> / <u>19.5</u>  | 67.8 / <u>67.2</u>             | 63.1 / <u>61.8</u>         |

predicts a residual lag, since state-modification actions still come after transport, and for the 70B, a residual of +2.9 to +3.9 points is what remains. But the bulk of the gap was an exploration artifact: under partial observability, the human’s slower search pushed its work later into the episode and inflated the robot’s share of completed subtasks. This qualifies the assistive-robotics caution in Section 4.3: the finisher role is real but considerably milder once the partner is not also searching. Prediction restores the asymmetry in three of four cases, driving the 8B lag to +9.8 and +10.5 points, consistent with the robot’s larger prediction space imposing an asymmetric deliberation burden.

**The robot’s execution advantage, by contrast, is unaffected by observability, which supports attributing it to the action spaces.** Robot Action Success exceeds human Action Success in all eight full-observability runs, by +3.8 to +8.5 points for the Llama 8B and +0.7 to +1.9 for the Llama 70B, the same ordering and roughly the same magnitudes as the 28 of 32 runs in the partial regime. Full observability also lifts the 70B’s best Action Success by about 2 points (best 84.2/83.3% against 82.1/81.7%). An asymmetry that persists unchanged when perceptual uncertainty is removed is most naturally explained by the precondition structure of the human’s state-modification actions rather than by anything the agents can or cannot see.

Overall, full observability saves much time but adds little accuracy: it removes search almost entirely, leaves most redundant work untouched, and roughly halves the completion lag between the agents for the 70B while removing it for the 8B. What it does not do is substitute for knowing what one’s partner is doing, which is why OAS retains its coordination effects here while losing its task-quality ones, and why prediction remains costly even when nothing is hidden.

## A.5 Further Action Prediction Evaluations

### A.5.1 Next Action Target for Decoupled Partial Observability

Table 7 re-scores the same OANA-enabled runs against a stricter target: only the partner’s *subsequent* high-level action counts as correct, whereas Table 2 also credited a prediction that matched the action the

partner was executing at the moment of prediction (Section 3.3). The two tables therefore describe identical agent behavior scored under two readings of what it means to anticipate a partner. The union target measures whether an agent has a correct model of what its partner is doing *or* about to do; the strict target isolates genuine anticipation. Comparing them separates the part of our prediction results that reflects reading the partner’s present state from the part that reflects projecting its future.

**The strict target removes roughly a third of all prediction accuracy, and on average, the entire loss sits in the action-type term.** Averaged over the 32 model×context×agent cells, Action Type accuracy falls by 16.0 points, or 36.5% in relative terms, with the drop ranging from −8.7 points (Llama 8B, baseline, robot) to −25.4 (Gemini, OAS + KE, human). The conditional accuracies do not fall on average (all-parameter 52.2% → 52.4%), but they split by planner. For the Llama 70B the conditional principal-parameter accuracy rises in all eight cells, by as much as +11.4 points (OAS only, human, 62.2% → 73.6%), and for DeepSeek in all eight, by up to +9.8; for the Llama 8B it falls in all eight, and for Gemini in six of eight. Since the joint metrics factorize as  $\text{Acc}(\text{type}) \times \text{Acc}(\text{params} \mid \text{type})$ , an identity we again verified in all 64 cells of Table 7, the stricter target makes the coarse problem markedly harder for every planner, while making the fine problem somewhat easier for the 70B and DeepSeek and somewhat harder for the 8B and Gemini.

**That divergence identifies what the union target was crediting: ongoing actions that are easy to type and hard to parameterize.** When an agent observes its partner *Walking in hallway\_1*, answering *Navigate[·]* scores a correct action type under the union target without any anticipation, because the ongoing action is directly readable from the OAS observation, but the destination argument remains unconstrained. Such events inflate type accuracy and depress conditional parameter accuracy simultaneously. Excluding them, as the strict target does, therefore produces the pattern we observe for the Llama 70B and DeepSeek, the two planners whose union-target Action Type gains most from OAS (Section 4.4); for the 8B and Gemini, the conditional term falls as well, so this account covers only part of the data. This also means the union results should be read as a mixture of two capabilities: state recognition, which is comparatively easy and which the strict target removes, and anticipation, which is what remains here. The gap between the two tables is a quantitative estimate of how much of apparent partner modeling is the former.

**Shared context still helps, and for the three stronger planners it helps substantially, but the smallest planner gains nothing once the ongoing action is excluded.** Moving from the object-only baseline to OAS + KE improves the strictest joint metric (action type plus all parameters) by +7.8/+5.4 points for Llama 70B, +7.5/+4.1 for Gemini and +10.9/+11.1 for DeepSeek (R/H). For the Llama 8B, the same move yields −0.2/+0.8 points: its robot side is marginally *worse* with full shared context than without it (10.1%→9.9%). The union-target analysis already found the 8B’s gains nearly cancelling between the two factors (15.7%→16.9%); under the strict target, the cancellation is complete. Whatever the 8B extracts from partner state and shared object knowledge supports recognizing what its partner is currently doing and does not support predicting what it will do next.

**Gemini’s best strict-target prediction comes from object knowledge alone, not from partner state.** Under the union target, OAS + KE topped Action Type in all eight model×agent combinations (strictly in seven). Under the strict target, it tops only four. Gemini is best with KE only on both sides (30.4/29.4% against 30.3/27.3% with OAS + KE, and against 27.6/25.9% with OAS alone), and both Llama planners peak on the human side with OAS only. The KE-versus-OAS contrast reported in Section 4.4 consequently becomes stronger for Gemini, where KE alone now delivers a larger Action Type gain than OAS alone on both sides (+7.3/+8.4 against +4.5/+4.9), while for the Llama 70B it delivers about half of the OAS gain on the robot side and two-thirds on the human side (+3.2/+4.3 against +7.2/+6.4). The self-projection account we proposed, in which a synchronized object graph lets an agent predict its partner by asking what it would itself do next, is an account of *anticipation* specifically, and it is under the anticipation-only target that Gemini’s reliance on object knowledge is most visible.

**Under the strict target, the human’s action-type advantage disappears in the partial regime.** Under the union target, the human agent led the robot on Action Type in 11 of 16 cells, and we attributed this to the human predicting over the robot’s 9 action types while the robot predicts over the human’s 14. Under the strict target, the ordering inverts: the robot matches or leads in 14 of 16 cells. The Llama 70B’s

gap moves from +3.3 to -0.6 points at baseline and from +0.3 to -2.8 with OAS + KE; DeepSeek’s moves from +3.3 to -0.2 and from +5.7 to 0.0. The relative accuracy loss is also consistently larger on the human side for the three stronger planners (for example, -41.6% against -36.2% for the Llama 70B with OAS + KE), and only slightly larger (0.7–2.6 points) for the 8B. A target-space-size effect should be indifferent to whether the target is the current or the next action; a state-recognition effect should not. This suggests that the human’s apparent edge at coarse partner prediction came largely from the robot’s ongoing actions being easy to read, which follows from the robot’s role as the transport layer (Section 4.3) and hence its disproportionate time spent navigating, rather than from the human facing a smaller hypothesis space.

**The robot’s conditional parameter advantage is not robust either and survives clearly only for Gemini.** Under the union target, the robot led on conditional principal-parameter accuracy in the three stronger planners once either channel was enabled. Under the strict target, this holds only for Gemini, where it in fact widens considerably (from -5.7, -7.1, -6.6 to -8.4, -10.8, -11.5 points H-R for KE, OAS and OAS + KE). For the Llama 70B, the lead shrinks or reverses (+5.7, -1.3, +1.4, +0.6); for DeepSeek, it shrinks to at most -4.1; and for the 8B, the human leads in all four contexts. The double dissociation reported in Section 4.4 is thus a property of the union scoring rather than a stable feature of the two prediction problems, and the arity-based account we offered for it applies to that scoring only.

**One consequence of this is that the joint metric is no longer close to symmetric for every planner.** The H-R gap on action type plus all parameters stayed within  $\pm 3.3$  points in all 16 union cells because the two factor-level asymmetries pointed in opposite directions and cancelled. Under the strict target, the cancellation survives for the Llama 8B ( $\pm 1.3$ ), the Llama 70B ( $\pm 1.3$ ) and DeepSeek ( $\pm 1.1$ ), but fails for Gemini, whose robot leads by 2.8, 3.9 and 4.7 points once any shared context is enabled, because its type asymmetry and its conditional asymmetry now both favor the robot. Symmetry in an aggregate metric is therefore coincidental rather than structural, which is a further argument for reporting the two factors separately.

**The parameter bottleneck is the same under both targets.** Among predictions naming every parameter correctly, the action type is also correct 85.7% of the time on average (range 77.1–94.4%), against 84.8% under the union target; among correctly-typed predictions, 52.4% carry a fully correct parameter set (range 37.9–67.3%), against 52.2%. The two scorings differ by 0.9 and 0.2 points on these conditionals, despite differing by 16 points on unconditional type accuracy. Getting the arguments right almost implies getting the verb right regardless of how the target is defined, and the reverse fails about half the time in both cases. The asymmetry between the coarse and fine sub-problems is thus a property of partner modeling in this environment and not an artifact of the temporal flexibility in our matching protocol.

**Absolute anticipation accuracy is low enough to reframe the trade-off documented in Section 4.2.** The best strict-target performance anywhere in Table 7 is 35.4/33.3% on action type and 23.2/22.2% on action type plus all parameters, from the Llama 70B with OAS + KE on the robot side and with OAS alone on the human side. The 8B never exceeds 26.2% on action type or 10.4% on the strictest metric. Agents required to predict their partner are therefore acting alongside an anticipation that is wrong roughly two-thirds of the time at the coarsest granularity and three-quarters of the time at the finest, even in the best-resourced configuration. That OANA prediction lowers Success Rate in fifteen of sixteen model $\times$ context pairs is considerably less surprising in this light, and the monotone relationship we reported between prediction accuracy and the size of that penalty for DeepSeek becomes the more natural reading: prediction is not merely an overhead but an overhead built on a mostly incorrect signal.

In summary, the strict target leaves the mechanism-level conclusions of Section 4.4 intact (the factorization, the parameter bottleneck, the planner dependence of how KE is used, and the 8B’s inability to convert context into complete predictions), while substantially revising the agent-level ones. The asymmetries we attributed to the structure of the two action spaces turn out to depend on crediting ongoing actions, and next-action anticipation is both considerably weaker than the union numbers suggest and more evenly distributed between the two agents than those numbers imply.

Table 8: **Full observability OANA prediction accuracy (%)** across action abstraction levels for robot (R) and human (H) agents. The target action is defined as the **union of the current and next actions** of the other agent. Results show that allowing OAS observation mostly improves prediction accuracy in the case of the 8B. Using Llama-3.1-Instruct models as planners for the agents.

| Model Size | OAS | Action Type (R/H) ↑       | Principal Par. (R/H) ↑ | A. Type+Princ. Par. (R/H) ↑ | All Params (R/H) ↑        | A. Type+All Params (R/H) ↑ | Principal Par. if Type (R/H) ↑ | All Params if Type (R/H) ↑ |
|------------|-----|---------------------------|------------------------|-----------------------------|---------------------------|----------------------------|--------------------------------|----------------------------|
| 8B         | ✗   | 29.3 / 33.2               | 26.6 / <u>28.6</u>     | 17.0 / 20.9                 | 17.1 / 18.6               | 14.1 / 16.4                | 58.1 / <u>62.7</u>             | 48.1 / 49.3                |
|            | ✓   | <u>33.4</u> / <u>35.5</u> | <u>27.2</u> / 28.3     | <u>20.1</u> / <u>22.1</u>   | <u>19.1</u> / <u>20.0</u> | <u>16.9</u> / <u>18.4</u>  | <u>60.3</u> / 62.3             | <u>50.5</u> / <u>51.6</u>  |
| 70B        | ✗   | 57.0 / 64.2               | 49.7 / <b>53.3</b>     | 40.8 / <b>44.6</b>          | 40.8 / <b>44.4</b>        | 35.5 / <b>39.3</b>         | <b>71.5</b> / <b>69.5</b>      | <b>62.3</b> / <b>61.3</b>  |
|            | ✓   | <b>65.6</b> / <b>70.8</b> | <b>50.2</b> / 48.2     | <b>44.5</b> / 43.2          | <b>42.7</b> / 41.7        | <b>38.5</b> / 38.5         | 67.8 / 60.9                    | 58.7 / 54.3                |

Table 9: **Full observability OANA prediction accuracy (%)** across action abstraction levels for robot (R) and human (H) agents. The target action is defined as the **next action** of the other agent. Results show that allowing Other Agent’s State (OAS) observation mostly improves prediction accuracy in the case of the 8B model, except for the Principal Parameter, while for the 70B model, it hinders the performance. Using Llama-3.1-Instruct models as planners for the agents.

| Size | OAS | Action Type (R/H) ↑       | Principal Par. (R/H) ↑    | A. Type+Princ. Par. (R/H) ↑ | All Params (R/H) ↑        | A. Type+All Params (R/H) ↑ | Principal Par. if Type (R/H) ↑ | All Params if Type (R/H) ↑ |
|------|-----|---------------------------|---------------------------|-----------------------------|---------------------------|----------------------------|--------------------------------|----------------------------|
| 8B   | ✗   | 23.3 / 26.3               | <u>22.9</u> / <u>23.5</u> | 14.6 / 17.6                 | 13.5 / 14.1               | 11.7 / 13.3                | 62.5 / 67.2                    | 50.1 / 50.7                |
|      | ✓   | <u>23.4</u> / <u>26.7</u> | 21.7 / 23.0               | <u>15.5</u> / <u>18.6</u>   | <u>13.8</u> / <u>15.4</u> | <u>12.2</u> / <u>14.8</u>  | <u>66.1</u> / <u>69.5</u>      | <u>52.4</u> / <u>55.5</u>  |
| 70B  | ✗   | <b>43.8</b> / <b>45.9</b> | <b>41.8</b> / <b>41.4</b> | <b>35.3</b> / <b>36.5</b>   | <b>33.2</b> / <b>33.3</b> | <b>30.3</b> / <b>31.3</b>  | 80.6 / <b>79.5</b>             | <b>69.2</b> / <b>68.1</b>  |
|      | ✓   | 43.5 / 43.1               | 38.6 / 36.3               | <b>35.3</b> / 33.9          | 31.4 / 30.5               | 29.5 / 29.3                | <b>81.2</b> / 78.5             | 67.8 / 67.9                |

## A.5.2 Action Prediction Evaluations for Full Observability

Tables 8 and 9 report prediction accuracy under full observability for the union and the strict next-action targets, respectively, and Tables 10 and 11 place both alongside the corresponding Decoupled Partial Observability runs. Object KE is again inapplicable, since full observability already supplies every object position at all times, so the design is a  $2 \times 2$  of OAS observation and target definition, run with the two Llama planners. Crossing the observability regime with target definition is what makes these runs informative: the regime controls how much the predictor knows about the *world*, and the target controls whether credit requires recognizing the partner’s *present* action or anticipating its *future* one. The joint metrics again factorize, to within rounding, as  $\text{Acc}(\text{type}) \times \text{Acc}(\text{params} \mid \text{type})$  in all 32 cells of the two tables, so we continue to attribute every movement to the type term or the conditional term.

**Full observability produces the strongest partner prediction anywhere in this work, but only for the larger planner.** The Llama 70B reaches 65.6/70.8% Action Type accuracy under the union target with OAS, against a best of 55.5/55.8% across all Decoupled Partial configurations, and 43.8/45.9% under the strict target without OAS, against 35.4/33.3% in the partial regime. The gains are +10.1/+15.0 and +8.4/+12.6 points, respectively, and they extend to the strictest joint metric (+4.4/+6.1 under union, +7.1/+9.1 under strict). Notably, full observability *without* OAS already exceeds the best Decoupled Partial configuration with both OAS and KE on all eight R/H comparisons we can make, so complete object knowledge substitutes for, and then surpasses, the co-location-gated channels, at least for a planner able to use it.

**For the smaller planner, the same intervention improves parameters and not verbs, and on the robot side, it makes coarse prediction worse.** The Llama 8B’s best Action Type accuracy falls from 37.9% to 33.4% (union) and from 26.2% to 23.4% (strict) on the robot side when moving from its best partial-observability configuration to full observability, while rising modestly on the human side (+1.3 and +3.1). Its conditional accuracies, by contrast, improve substantially: conditional principal-parameter

Table 10: **OANA prediction accuracy (%)** comparison across Decoupled Partial and Full observability settings for robot (R) and human (H) agents. The target action is the **union of the current and next actions** of the other agent. Results show that full observability improves prediction for the Llama 70B and the 8B human agent, while the 8B robot’s Action Type accuracy falls below its best partial-observability configuration. Subgroup bests (per model scale) are underlined, and overall bests are bolded.

| Model Size | Observability     | OAS KE OANA |   |   | Action Type ↑ |             | A. Type + Princ. Param. ↑ |             | A. Type + All Param. ↑ |             |
|------------|-------------------|-------------|---|---|---------------|-------------|---------------------------|-------------|------------------------|-------------|
|            |                   |             |   |   | R             | H           | R                         | H           | R                      | H           |
| 8B         | Decoupled Partial | ✗           | ✗ | ✓ | 31.8          | 29.5        | 17.2                      | 15.8        | 15.7                   | 14.1        |
|            | Decoupled Partial | ✗           | ✓ | ✓ | 33.3          | 31.1        | 18.4                      | 17.8        | 16.6                   | 15.4        |
|            | Decoupled Partial | ✓           | ✗ | ✓ | 36.1          | 33.8        | 17.8                      | 17.9        | 16.4                   | 15.8        |
|            | Decoupled Partial | ✓           | ✓ | ✓ | <u>37.9</u>   | 34.2        | 18.9                      | 17.2        | <u>16.9</u>            | 15.5        |
|            | Full              | ✗           | - | ✓ | 29.3          | 33.2        | 17.0                      | 20.9        | 14.1                   | 16.4        |
|            | Full              | ✓           | - | ✓ | 33.4          | <u>35.5</u> | <u>20.1</u>               | <u>22.1</u> | <u>16.9</u>            | <u>18.4</u> |
| 70B        | Decoupled Partial | ✗           | ✗ | ✓ | 38.5          | 41.8        | 24.1                      | 26.5        | 22.1                   | 24.3        |
|            | Decoupled Partial | ✗           | ✓ | ✓ | 43.6          | 49.6        | 29.5                      | 32.0        | 26.5                   | 28.7        |
|            | Decoupled Partial | ✓           | ✗ | ✓ | 52.9          | 55.8        | 35.2                      | 34.7        | 31.7                   | 32.3        |
|            | Decoupled Partial | ✓           | ✓ | ✓ | 55.5          | 55.8        | 37.3                      | 35.8        | 34.1                   | 33.2        |
|            | Full              | ✗           | - | ✓ | 57.0          | 64.2        | 40.8                      | <b>44.6</b> | 35.5                   | <b>39.3</b> |
|            | Full              | ✓           | - | ✓ | <b>65.6</b>   | <b>70.8</b> | <b>44.5</b>               | 43.2        | <b>38.5</b>            | 38.5        |

accuracy rises by +5.0/+5.5 points under the union target and by +15.3/+13.1 under the strict target (50.8/56.4%→66.1/69.5%). The decomposition localizes this cleanly. Perfect object knowledge resolves the ambiguity that the conditional term measures (which objects and receptacles are involved) and does nothing for the question of which verb the partner will choose, which for this planner is the binding constraint. The small robot-side regression on Action Type is consistent with the prompt-length effect noted in Appendix A.3.3: full observability enumerates every object in the scene, and the 8B pays for that context rather than exploiting it.

**Crossing the two targets isolates what OAS observation contributes, and under full observability, it contributes essentially nothing but present-state recognition.** For the Llama 70B, enabling OAS raises union-target Action Type by +8.6/+6.6 points but changes strict-target Action Type by −0.3/−2.8. The same intervention that appears to improve partner prediction by 6.6 to 8.6 points does not improve anticipation at all. This directly tests the mechanism proposed in Appendix A.5.1: OAS observations describe what the partner is doing right now (*Walking in hallway\_1, Picking apple\_0*), so they let an agent name the partner’s ongoing action, which the union target credits and the strict target does not. Under Decoupled Partial Observability, roughly half of the OAS benefit survived the strict target (70B: +14.4/+14.0 under union against +7.2/+6.4 under strict), because there, OAS also carried object information that was otherwise unavailable. Once full observability supplies that information independently, OAS only reports the partner’s current state.

**The union-to-strict accuracy loss is accordingly concentrated in the configurations where OAS is enabled.** Under full observability, the relative drop from union to strict target is 20.5–28.5% without OAS and 24.8–39.1% with it, the two largest values in the table being the 70B’s OAS-enabled cells (−33.7% and −39.1%). Averaged over the two Llama planners, the drop is 27.6% under full observability against 32.5% under Decoupled Partial Observability, so complete world knowledge makes a larger share of measured prediction accuracy reflect anticipation rather than state recognition, while adding OAS on top pushes it back the other way. Any evaluation of partner modeling that credits the ongoing action should therefore report the strict variant alongside it, because the two scorings respond in opposite directions to the same intervention.

**Under full observability, OAS degrades the conditional term for the larger planner, which mirrors its behavior in task solving.** With the union target, enabling OAS costs the 70B −3.7/−8.6

Table 11: **OANA prediction accuracy (%)** comparison across Decoupled Partial and Full observability settings for robot (R) and human (H) agents. The target action is the **next action** of the other agent. Results highlight that full observability improves prediction for the Llama 70B and the 8B human agent, while the 8B robot’s Action Type accuracy falls below its best partial-observability configuration; under full observability, adding OAS does not improve the 70B’s next-action prediction. Subgroup bests (per model scale) are underlined, and overall bests are bolded.

| Model Size | Observability     | OAS KE OANA  |              |              | Action Type $\uparrow$ |             | A. Type + Princ. Param. $\uparrow$ |             | A. Type + All Param. $\uparrow$ |             |
|------------|-------------------|--------------|--------------|--------------|------------------------|-------------|------------------------------------|-------------|---------------------------------|-------------|
|            |                   |              |              |              | R                      | H           | R                                  | H           | R                               | H           |
| 8B         | Decoupled Partial | $\times$     | $\times$     | $\checkmark$ | 23.1                   | 20.7        | 11.5                               | 10.4        | 10.1                            | 8.8         |
|            | Decoupled Partial | $\times$     | $\checkmark$ | $\checkmark$ | 24.0                   | 21.6        | 12.2                               | 12.2        | 10.4                            | 9.9         |
|            | Decoupled Partial | $\checkmark$ | $\times$     | $\checkmark$ | 25.8                   | 23.6        | 11.3                               | 12.3        | 10.0                            | 10.2        |
|            | Decoupled Partial | $\checkmark$ | $\checkmark$ | $\checkmark$ | <u>26.2</u>            | 23.4        | 11.9                               | 11.4        | 9.9                             | 9.6         |
|            | Full              | $\times$     | -            | $\checkmark$ | 23.3                   | 26.3        | 14.6                               | 17.6        | 11.7                            | 13.3        |
|            | Full              | $\checkmark$ | -            | $\checkmark$ | 23.4                   | <u>26.7</u> | <u>15.5</u>                        | <u>18.6</u> | <u>12.2</u>                     | <u>14.8</u> |
| 70B        | Decoupled Partial | $\times$     | $\times$     | $\checkmark$ | 27.5                   | 26.9        | 17.3                               | 18.5        | 15.4                            | 16.5        |
|            | Decoupled Partial | $\times$     | $\checkmark$ | $\checkmark$ | 30.7                   | 31.2        | 21.5                               | 21.4        | 18.6                            | 18.4        |
|            | Decoupled Partial | $\checkmark$ | $\times$     | $\checkmark$ | 34.7                   | 33.3        | 25.0                               | 24.5        | 21.7                            | 22.2        |
|            | Decoupled Partial | $\checkmark$ | $\checkmark$ | $\checkmark$ | 35.4                   | 32.6        | 26.3                               | 24.4        | 23.2                            | 21.9        |
|            | Full              | $\times$     | -            | $\checkmark$ | <b>43.8</b>            | <b>45.9</b> | <b>35.3</b>                        | <b>36.5</b> | <b>30.3</b>                     | <b>31.3</b> |
|            | Full              | $\checkmark$ | -            | $\checkmark$ | 43.5                   | 43.1        | <b>35.3</b>                        | 33.9        | 29.5                            | 29.3        |

points of conditional principal-parameter accuracy and  $-3.6/-7.0$  of conditional all-parameter accuracy, and it lowers the human agent’s unconditional Principal Par., A. Type+Princ. Par., All Params, and A. Type+All Params scores as well. The pattern matches what we report for task performance in Appendix A.4.1: once perceptual uncertainty is removed, OAS retains the functions that are intrinsic to partner state and loses the ones that were compensating for missing world knowledge, while its prompt cost remains. For the 8B, whose baseline competence is lower, OAS still helps on both targets, so the cost only dominates where the remaining headroom is small.

**Full observability largely dissolves the parameter bottleneck while leaving the type bottleneck intact.** The highest conditional accuracies in this paper occur under full observability with the strict target: 80.6/79.5% and 81.2/78.5% conditional principal-parameter accuracy for the 70B, against 74.4/75.0% at its best in the partial regime and 67.6/64.5% under the union target there. Roughly one correctly typed prediction in five now carries a wrong principal parameter, down from about one in three. Unconditional Action Type under the strict target is 43.8/45.9% without OAS and 43.5/43.1% with it. An agent with a complete, correct, and continuously updated model of the world, predicting a partner acting in that same fully known world, still names the wrong next action type more than half the time. The residual difficulty in partner modeling is therefore not informational in the sense our interventions address: it concerns which of several individually plausible actions a partner will select, and neither partner-state observation, nor object knowledge exchange, nor complete observability resolves it.

**The human’s coarse-prediction advantage reappears under full observability and survives the strict target, consistent with the action-space-size account holding once object uncertainty is removed.** The human agent leads on Action Type in all four union-target cells (+3.9, +2.1, +7.2, +5.2 points H–R) and in three of four strict-target cells (+3.0, +3.3, +2.1,  $-0.4$ ). This contrasts with Decoupled Partial Observability, where the robot matched or led in 14 of 16 strict-target cells. The resolution the two regimes together suggest is that the type-space advantage is real but only expressible once object uncertainty is removed: under partial observability, not knowing where things are dominates the prediction problem for both agents and masks the difference in how many verbs each must choose among; under full observability,

that source of difficulty is gone, and the smaller hypothesis space shows through. This account rests on two planners of one family.

**The conditional asymmetry, by contrast, is stable in direction within each planner and opposite between them.** Under full observability, the 8B human leads on conditional principal-parameter accuracy in all four cells (+4.6, +2.0, +4.7, +3.4 points H–R across both targets), while the 70B robot leads in all four (−2.0, −6.9, −1.1, −2.7). The 8B ordering matches what it showed under the strict target in the partial regime, whereas the 70B robot led in only one of four contexts there (Appendix A.5.1), so its lead is stable under full observability but not across regimes. The split is thus a planner property rather than a target-definition artifact, which is a further reason to treat the arity-based account offered in Section 4.4 as unconfirmed: an explanation grounded in the structure of the two action spaces should predict the same direction for both planners, and it does not.

Overall, full observability helps a capable planner much more than a weak one: it largely solves the parameter part of the prediction problem, leaves the action-type part largely unsolved, and reduces OAS observation to information about the partner’s present state, which improves scores that credit present-state recognition but not anticipation. The best strict-target accuracy we measure anywhere (43.8/45.9% at the action-type level, 30.3/31.3% with all parameters) is attained by agents that know everything about the world and are told nothing about their partner. Even in this best case, more than half of the predicted action types are wrong, so the OANA cost reported in Section 4.2 comes from acting on mostly incorrect predictions.

## A.6 Prompts for the agents

See the prompts below, which were used for the agents in various experiments. We are not using the *{rag\_examples}* tag, since we are using zero-shot planners. The *{input}* tag would be substituted with the task description, e.g., *Place the candle and candle holder on the white table in the living room. Move the plant to the same table.*

The instruction prompt for the Decentralized ReAct agents **without** OANA prediction (the PARTNR baseline prompt; Chang et al., 2025):

### Decentralized | Zero-Shot | ReAct | Without OANA prediction

```
{system_tag}You are an agent that solves multi-agent planning problems. The task assigned to you will be situated in a house and will generally involve navigating to objects, picking and placing them on different receptacles to achieve rearrangement. You strictly follow any format specifications and pay attention to the previous actions taken in order to avoid repeating mistakes. Rooms do not need to be explored more than once.
```

```
There will be another agent trying to solve the same task that you are at the same time. You may find that that agent has picked up relevant objects or is in the process of completing parts of the task. If that is the case you may want to move on to a different part of the task.
```

```
Rooms do not need to be explored more than once. This means if you have explored the living room and have not found the object, then you should explore the kitchen, if a relevant object is still not found, you should explore the hallway etc...
```

```
{agent_role_description}
```

```
Many calls to the same action in a row are a sign that something has gone wrong and you should try a different action.{eot_tag}{rag_examples}{user_tag}Task: {input}
```

```
{world_description}
```

```

Possible Actions:
{tool_descriptions}
- Done: Used to indicate that the agent has finished the task. Example (Done[])

What is the next action to make progress towards completing the task?
Return your response in the following format

Thought: <reasoning for why you are taking the next action>
<next action call>
Assigned!

Here is an example:
Thought: Since there are no objects found I should explore a room I have not
explored yet
Explore[<room name>]
Assigned!
{eot_tag}{assistant_tag}

```

Our instruction prompt for the Decentralized ReAct agents **with** OANA prediction is shown below. Since the human uses extra object state modifier actions as well, the *{your\_tool\_descriptions}* and the *{other\_agent\_tool\_descriptions}* would be different based on the given roles (see the tool description later).

### Decentralized | Zero-Shot | ReAct | With OANA prediction

```

{system_tag}You are an agent (Agent A) that solves multi-agent planning problems.
The task assigned to you will be situated in a house and will generally involve
navigating to objects, picking and placing them on different receptacles to achieve
rearrangement. You strictly follow any format specifications and pay attention to
the previous actions taken in order to avoid repeating mistakes. Rooms do not need
to be explored more than once.

There will be another agent (Agent B) trying to solve the same task that you are at
the same time. You may find that that agent has picked up relevant objects or is in
the process of completing parts of the task. If that is the case, you may want to
move on to a different part of the task.
The other agent (Agent B) is not controlled by you, but you are working together to
solve the common task, so you should predict what the other agent (Agent B) will do
next and take that into account when deciding your next action.

Rooms do not need to be explored more than once. This means if you have explored
the living room and have not found the object, then you should explore the kitchen,
if a relevant object is still not found, you should explore the hallway etc...

{agent_role_description}

Many calls to the same action in a row are a sign that something has gone wrong and
you should try a different action.{eot_tag}{rag_examples}{user_tag}Task: {input}

{world_description}

Your (Agent A) possible actions:
{your_tool_descriptions}

```

- Done: Used to indicate that the agent has finished the task. Example (Done[])

The other agent's (Agent B) possible actions:

{other\_agent\_tool\_descriptions}

- Done: Used to indicate that the agent has finished the task. Example (Done[])

What is your prediction of the other agent's (Agent B) next action to make progress towards completing the task?

What is your (Agent A) next action to make progress towards completing the task?

Return your response in the following format

Thought: <reasoning for what is the other agent's (Agent B) next action and why you (Agent A) are taking your next action>

The other agent's predicted next action: <other agent's (Agent B) predicted next action call>

My next action: <your (Agent A) next action call>

Assigned!

Here is an example:

Thought: The other agent is probably exploring a room for the objects, but since no objects were found, I should explore a room too, which I have not explored yet

The other agent's predicted next action: Explore[<room name>]

My next action: Explore[<room name>]

Assigned!

{eot\_tag}{assistant\_tag}

The agent's role description depends on whether it is assigned the role of the human or the robot, and would be specified accordingly (Chang et al., 2025):

### Agent Role Description

#### Human's Role Description:

You are playing the role of the task giver. This means if the instruction says something like "You should move the object and I will wash it", then the other agent should be moving the object, and you should washing the it.

#### Robot's Role Description:

You are playing the role of the task receiver. This means if the instruction says something like "You should move the object and I will wash it", then you should move the object and the other agent should wash it.

The action tool descriptions are shown below (from Chang et al., 2025).

### Tool Description

- Clean: Used for cleaning an object. You need to provide the name of the object to clean.

- Close: Used for closing an articulated entity. You must provide the name of the furniture you want to close. Example (Close[chest\_of\_drawers\_1])

- Explore: Search a specific room by visiting various receptacles or furnitures in that room. The input to the skill is the EXACT name of the room to be visited. Use

the room names provided in the house description. This tool exhaustively explores the specified room. Example (Explore[kitchen\_1])

- Fill: Used for filling an object. You need to provide the name of the object to fill.
- Navigate: Used for navigating to an entity. You must provide the name of the entity you want to navigate to. Example (Navigate[counter\_22])
- Open: Used for opening an articulated entity. You must provide the name of the furniture you want to open. Example (Open[chest\_of\_drawers\_1])
- Pick: Used for picking up an object. You must provide the name of the object to be picked. The agent cannot hold more than one object at a time. Example (Pick[cup\_1])
- Place: Used for placing an object on a target location. You need to provide the name of the object to be placed, the name of the furniture where it should be placed, spatial relation ("on" or "within") describing the relation between the object and furniture. The object to be placed must already be held by the agent (i.e. picked previously). In addition to these, you can request to place the object near another object. For that you can optionally provide a spatial constraints ("next\_to") and the name of the reference object. To place next to an object, the reference object must already be on the target furniture. API template  
Place[<object\_to\_be\_placed>, <spatial\_relation>, <furniture to be placed on>, <spatial\_constraint>, <reference\_object>]. spatial\_constraint and reference\_object should be set to "None" when necessary.
- Pour: Used for pouring from one container to another. This skill will pour into the specified container from whichever container is currently held by the agent.
- PowerOff: Used for turning off a powered object. You need to provide the name of the object to be turned off.
- PowerOn: Used for turning on a powered object. You need to provide the name of the object to be turned on.
- Rearrange: Used for moving an object from its current location to the target location. You need to provide the name of the object to be moved, the name of the furniture where it should be moved, spatial relation ("on" or "within") describing the relation between the object and furniture. This will automatically pick the specified object and move to the target furniture and attempt to place it. In addition to these, you can request to place the object near another object. For that you can optionally provide a spatial constraint ("next\_to") and the name of the reference object. To place next to an object, the reference object must already be on the target furniture. API template  
Rearrange[<object\_to\_be\_moved>, <spatial\_relation>, <furniture to be placed on>, <spatial\_constraint>, <reference\_object>]. spatial\_constraint and reference\_object should be set to "None" when necessary.
- Wait: Used to make agent stay idle for some time. Example (Wait[])
- Done: Used to indicate that the agent has finished the task. Example (Done[])

Only the human has access to these tools, which modify the object state on top of the base tools. The tool descriptions for the agents differ on this (from Chang et al., 2025).

### Human Only Tool Descriptions

- Clean: Used for cleaning an object. You need to provide the name of the object to clean.
- Fill: Used for filling an object. You need to provide the name of the object to fill.
- Pour: Used for pouring from one container to another. This skill will pour into the specified container from whichever container is currently held by the agent.

- PowerOff: Used for turning off a powered object. You need to provide the name of the object to be turned off.
- PowerOn: Used for turning on a powered object. You need to provide the name of the object to be turned on.

The world description provides a textual overview of all rooms, their furniture, and the objects currently observed. In (Decoupled) Partial observability, the objects are not known in advance, and exploration is required. An example of a scene is shown below:

### World Description Example

```
Furniture: living_room_1: floor_living_room_1, table_10, couch_18, shelves_26,
chair_28, chair_29, table_32, chair_33, chair_34, chair_35, chair_36, table_46,
table_47, stand_55, chest_of_drawers_56
other_room_1: floor_other_room_1
bathroom_1: floor_bathroom_1, chair_23, table_37, unknown_38, bench_48
bathroom_2: floor_bathroom_2, unknown_14, table_50
bedroom_1: floor_bedroom_1, chair_16, table_22, bed_45, bench_49,
chest_of_drawers_52, chest_of_drawers_54
laundryroom_1: floor_laundryroom_1, washer_dryer_11, washer_dryer_12, shelves_15
entryway_1: floor_entryway_1, bench_51
bedroom_2: floor_bedroom_2, table_13, chair_17, bed_21, table_24, table_25, chair_27,
stool_31, bench_44, table_59, chest_of_drawers_61
hallway_1: floor_hallway_1, table_30
kitchen_1: floor_kitchen_1, counter_19, chair_20, cabinet_39, cabinet_40, chair_41,
chair_42, chair_43, counter_53, cabinet_57, fridge_58, unknown_60

The following furnitures have a faucet: cabinet_57
Objects:
No objects found yet
```

For the experiments, we use a summarized representation of the scene; after each action call, we append the textual description of the objects observed by the agent. Note that the agents could move through different rooms in the house in order to get to another room and could discover and observe objects meanwhile.

### Scene Summary Example Object Only Baseline

```
<|start_header_id|>assistant<|end_header_id|>

Thought: Since there are no objects found yet, I should explore a room I have not
explored yet. The task requires the candle, candle holder, and plant, so I'll start
by exploring the living room to see if I can find any of these objects.
Explore[living_room_1]
Assigned!<|eot_id|><|start_header_id|>user<|end_header_id|>

Result: Successful execution!
Objects: candle_holder_1: table_30 in hallway_1
candle_0: table_30 in hallway_1
plant_container_2: table_30 in hallway_1
box_5: table_46 in living_room_1
jug_4: chair_27 in bedroom_2
box_3: washer_dryer_12 in laundryroom_1<|eot_id|>
```

<|start\_header\_id|>assistant<|end\_header\_id|>

Thought: I found the candle, candle holder, and plant container in the hallway, which is one of the rooms I needed to explore. Now that I have the objects, I can start working on the task. I'll navigate to the living room to pick up the candle holder, candle, and plant container.

Navigate[table\_30]

Assigned!<|eot\_id|><|start\_header\_id|>user<|end\_header\_id|>

Result: Successful execution!

Objects: candle\_holder\_1: table\_30 in hallway\_1

candle\_0: table\_30 in hallway\_1

plant\_container\_2: table\_30 in hallway\_1

box\_5: table\_46 in living\_room\_1

jug\_4: chair\_27 in bedroom\_2

box\_3: washer\_dryer\_12 in laundryroom\_1<|eot\_id|>

In the case of Other Agent's State Observation, the observed OAS is added to the textual observation description. If the Other Agent's Next Action Prediction is also enabled, then the prediction is made between the reasoning (Thought) and the action tool call at each replanning.

### Scene Summary Example With OAS Observation and OANA Prediction

<|start\_header\_id|>assistant<|end\_header\_id|>

Thought: Since the task requires placing the candle and candle holder on the white table in the living room, and moving the plant to the same table, I predict that Agent B will start by exploring the living room to find the objects. As Agent A, I will also explore a room to find the objects, but I will choose a different room to avoid duplication of effort. I will start by exploring the kitchen, as it is a common place to find objects like candles and plants.

The other agent's predicted next action: Explore[living\_room\_1]

My next action: Explore[kitchen\_1]

Assigned!<|eot\_id|><|start\_header\_id|>user<|end\_header\_id|>

Result of your action: Successful execution!

The other agent's actions since the previous observation: Acting in an other room, Walking in living\_room\_1, Acting in an other room, Walking in living\_room\_1, Acting in an other room, Walking in living\_room\_1, Acting in an other room

Your observations:

Objects: jug\_4: chair\_27 in bedroom\_2

vase\_7: table\_13 in bedroom\_2

clock\_6: chair\_17 in bedroom\_2

candle\_holder\_1: table\_30 in hallway\_1

candle\_0: table\_30 in hallway\_1

plant\_container\_2: table\_30 in hallway\_1

box\_5: table\_46 in living\_room\_1

box\_3: washer\_dryer\_12 in laundryroom\_1<|eot\_id|>

<|start\_header\_id|>assistant<|end\_header\_id|>

Thought: Since I found some objects in the kitchen, but not the ones required for the task, I will continue exploring other rooms to find the remaining objects. I noticed that Agent B was walking in the living room, but didn't find any objects

---

there. I also found some objects in the hallway, including the candle holder and candle, which are required for the task. I will navigate to the hallway to pick up these objects.

The other agent's predicted next action: Explore[bedroom\_2]

My next action: Navigate[table\_30]

Assigned!<|eot\_id|><|start\_header\_id|>user<|end\_header\_id|>

Result of your action: Successful execution!

The other agent's actions since the previous observation: Walking in living\_room\_1,

Acting in an other room

Your observations:

Objects: jug\_4: chair\_27 in bedroom\_2

vase\_7: table\_13 in bedroom\_2

clock\_6: chair\_17 in bedroom\_2

candle\_holder\_1: table\_30 in hallway\_1

candle\_0: table\_30 in hallway\_1

plant\_container\_2: table\_30 in hallway\_1

box\_5: table\_46 in living\_room\_1

box\_3: washer\_dryer\_12 in laundryroom\_1<|eot\_id|>