A Task and Motion Planning

In this work, we utilize PDDLStream (Garrett, Lozano-Pérez, and Kaelbling 2020) as the TAMP planner—a sampling-based TAMP framework that integrates conditional samplers, referred to as `streams`, into the planning domain definition language (PDDL (Fox and Long 2003)). A heuristic-based PDDL solver, such as FastDownward (Helmert 2006), is typically employed to generate high-level symbolic action sequences, while the streams resolve low-level geometric and kinematic feasibility.

Discrete task planning problems in PDDL are defined by a domain file and a problem file. The domain file includes object types, (derived) predicates $\mathcal{P}$, and actions $\mathcal{A}$, which remain fixed and independent of any specific problem instance. The problem file specifies a particular problem instance by providing objects $\mathcal{O}$, initial literals $\mathcal{I}$, and goal literals $\mathcal{G}$.

PDDLStream introduces the concept of `streams`, continuous-valued functions that, given a set of inputs (*i.e.*, objects and typed variables), generate both output values and certified predicates—literals that evaluate to true and thereby serve as valid preconditions for actions. For instance, streams can sample feasible poses for object placement or leverage external motion planners (*e.g.*, RRT-Connect (Kuffner and LaValle 2000)) to produce collision-free trajectories. Streams are specified in a stream file. PDDLStream still uses a domain file, as in PDDL, but does not use a problem file. Instead, initial and goal literals are expressed programmatically.

We provide a simple example to demonstrate the specifications of an action and the corresponding stream. The `move` action, responsible for the base movement of a robot, requires that the robot is at a known configuration `q1` (expressed by `(AtBConf ?q1)`) and that a trajectory from `q1` to `q2` is validated by a motion planner (expressed by `(BaseMotion ?q1 ?t ?q2)`) before it can be executed. The action's effects, defined in the **`eff`** field, state that the robot should no longer be at configuration `q1`, but should be at configuration `q2`.

```
(:action move
 :param (?q1 ?q2 ?t)
 :pre (and (BaseMotion ?q1 ?t ?q2) (AtBConf ?q1))
 :eff (and (AtBConf ?q2) (not (AtBConf ?q1))
           (increase (total-cost) (MoveCost))))
```

The `motion` stream takes an initial configuration `q1` and a goal configuration `q2` as input, returns a trajectory `t`, and certifies the precondition of the `move` action via the `(BaseMotion ?q1 ?t ?q2)` predicate if trajectory generation by external motion planners is successful.

```
(:stream motion
 :inp (?q1 ?q2)
 :dom (and (BConf ?q1) (BConf ?q2))
 :out (?t)
 :cert (and (BTraj ?t) (BaseMotion ?q1 ?t ?q2)))
```

To address partial observability, we introduce an observation action called the `detect` action as part of the action set $\mathcal{A}$, alongside conventional actions commonly used in PDDLStream, such as `pick`, `place`, `move`, and their corresponding streams.

The `detect` action is parameterized by the object `?o`, surface `?s`, room `?r`, sampled object pose from bel($x_{p,0}^k$) `?pb`, base configuration `?bq`, head configuration `?hq`, and head trajectory `?ht` of the robot.

```
(:action detect
 :param (?o ?s ?r ?pb ?bq ?hq ?ht)
 :pre (and (IsItem ?o) (IsSurf ?s) (IsRoom ?r)
           (AtBConf ?bq) (PoseB ?o ?pb ?s)
           (Vis ?o ?pb ?bq ?hq ?ht))
 :eff (and (Supported ?o ?pb ?s) (AtPoseB ?o ?pb)
           (increase (total-cost)
           (DetectCost ?o ?s ?r ?pb))))
```

Among the preconditions listed above, `(PoseB ?o ?pb ?s)` and `(Vis ?o ?pb ?pb ?hq ?ht)` are predicates that must be verified by streams due to the presence of continuous typed variables.

The `sample-PoseB` stream samples a pose `?pb` for an object `?o` from a specified surface `?s`. The stream then certifies that `?pb` is a valid sampled pose of object `?o` on surface `?s` by generating the predicate `(PoseB ?o ?pb ?s)`, which is one of the preconditions for the `detect` action.

```
(:stream sample-PoseB
 :inp (?o ?s)
 :dom (and (IsItem ?o) (IsSurf ?s))
 :out (?pb)
 :cert (and (PoseB ?o ?pb ?s)))
```

The `inverse-visibility` stream produces a robot configuration ensuring that the sampled object pose `?pb` remains within the robot's field of view. Specifically, `?pb` is considered visible if it lies within the robot's predefined visibility cone. This stream guarantees that the robot's orientation and proximity are appropriate for observing the location where the object is believed to be, by generating the predicate `(Vis ?o ?pb ?bq ?hq ?ht).`

```
(:stream inverse-visibility
:inp (?o ?pb ?s)
```

```
:dom (and (IsItem ?o) (IsSurf ?s) (PoseB ?o ?pb ?s))
:out (?hq ?ht ?bq)
:cert (and (Vis ?o ?pb ?bq ?hq ?ht)))
```

The preconditions `(IsItem ?o)`, `(IsSurf ?s)`, and `(IsRoom ?r)` state that the parameters `?o`, `?s`, and `?r` correspond to a specific object, surface, and room, respectively. `(AtBConf ?bq)` asserts that the robot's base configuration is `?bq`.

The effects of the `detect` action are defined as follows: `(AtPoseB ?o ?pb)` asserts that the object `?o` is at the sampled pose `?pb`, and `(Supported ?o ?pb ?s)` asserts that the sampled pose `?pb` of object `?o` is on the surface `?s`.

Similar to SS-Replan (Garrett et al. 2020), we employ the heuristic approach of a self-loop MDP (Keyder and Geffner 2008) to compute `DetectCost` as $cost(\texttt{detect}) = \frac{c_a}{p(s'|s,\texttt{detect})}$, where $p(s'|s,\texttt{detect})$ represents the probability of successfully detecting an object. This probability is directly influenced by the belief that the object is at the predicted semantic location and pose. Consequently, we define `DetectCost` as $\frac{c_a}{\mathrm{bel}(x^k_{r,t}, x^k_{s,t}, x^k_{p,0})}$.

B MCQA Prompt

The following is an example MCQA prompt used to estimate an object's (toaster) initial belief over surfaces.

```
Given a toaster is in the kitchen, predict the location of a toaster.
(A) Dishwasher
(B) Cabinet
(C) Counter
(D) Table
Return the letter that represents the location:
```

C Derivation for Belief Factorization

$$\mathrm{bel}(x^k_{r,t}, x^k_{s,t}, x^k_{p,t}) = P(x^k_{r,t}, x^k_{s,t}, x^k_{p,t} | \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, z^k_{p,1:t}, v_{r,1:t}, v_{s,1:t}) \tag{17}$$

$$= P(x^k_{p,t} | x^k_{r,t}, x^k_{r,t}, \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, z^k_{p,1:t}, v_{r,1:t}, v_{s,1:t}) P(x^k_{r,t}, x^k_{s,t} | \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, z^k_{p,1:t}, v_{r,1:t}, v_{s,1:t}) \tag{18}$$

$$= P(x^k_{p,t} | x^k_{r,t}, x^k_{r,t}, z^k_{p,1:t}) P(x^k_{r,t}, x^k_{s,t} | \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, v_{r,1:t}, v_{s,1:t}) \tag{19}$$

$$= P(x^k_{p,t} | x^k_{r,t}, x^k_{r,t}, z^k_{p,1:t}) P(x^k_{s,t} | x^k_{r,t}, \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, v_{r,1:t}, v_{s,1:t}) P(x^k_{r,t} | \mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}, v_{r,1:t}, v_{s,1:t}) \tag{20}$$

$$= P(x^k_{p,t} | x^k_{r,t}, x^k_{r,t}, z^k_{p,1:t}) P(x^k_{s,t} | x^k_{r,t}, \mathcal{Z}_{s,1:t}, v_{s,1:t}) P(x^k_{r,t} | \mathcal{Z}_{r,1:t}, v_{r,1:t}) \tag{21}$$

$$= P(x^k_{p,t} | x^k_{r,t}, x^k_{r,t}, z^k_{p,t}) P(x^k_{s,t} | x^k_{r,t}, \mathcal{Z}_{s,t}, v_{s,t}) P(x^k_{r,t} | \mathcal{Z}_{r,t}, v_{r,t}) \tag{22}$$

- (1) - (2): Chain rule.
- (2) - (3): Conditional independence assumption. Once the room and surface where the object is located($x^k_{r,t}, x^k_{s,t}$) are known, further observations over rooms and surfaces($\mathcal{Z}_{r,1:t}, \mathcal{Z}_{s,1:t}$) do not contribute additional information. Similarly, the visibility of the room and the surface ($v_{r,1:t} s_{r,1:t}$) does not contribute to estimating the pose of the object of interest.
- (3) - (4): Chain rule.
- (4) - (5): Conditional independence assumption. Once which room the object is located ($x^k_{r,t}$) is known, further observations over rooms ($\mathcal{Z}_{r,1:t}$) and the visibility of that room ($v_{r,t}$) does not provide additional information. Furthermore, the estimation of the object's room location depends solely on room observations($\mathcal{Z}_{r,1:t}$) and the room's visibility($v^k_{r,t}$).
- (5) - (6): Markov assumption.

D Surface Belief

Recursive Bayes filter equation for surface category estimation:

$$\mathrm{bel}(x^k_{s,t}) = P(x^k_{s,t} | x^k_{r,t}, \mathcal{Z}_{s,t}, v_{s,t}), \tag{23}$$

$$= \eta P(\mathcal{Z}_{s,t} | x^k_{s,t}, x^k_{r,t}, v_{s,t}) \overline{\mathrm{bel}}(x^k_{s,t}). \tag{24}$$

Observation model for updating $\mathrm{bel}(x^k_{s,t})$:

$$P(\mathcal{Z}_{s,t} \mid x^k_{s,t}, x^k_{r,t}, v_{s,t}) = P(z^k_{s,t} \mid x^k_{s,t}, x^k_{r,t}, v_{s,t}) \times \prod_{\substack{j=1 \\ j \neq k}}^{K} P\big(z^j_{s,t} \mid x^k_{s,t}, x^k_{r,t}\big). \tag{25}$$

Visibility-aware observation model on surfaces:

$$P(z^k_{s,t}|x^k_{s,t}, x^k_{r,t} v_{s,t}) = \begin{cases} (1 - p_{\text{fn}}) v_{s,t}, & \text{Detected,} \\ p_{\text{fn}} v_{s,t}, & \text{Not detected.} \end{cases} \tag{26}$$

$$P(z^k_{s,t}|x^k_{s,t}, x^k_{r,t}, v_{s,t}) = \begin{cases} p_{\text{fp}} v_{s,t}, & \text{Detected,} \\ 1 - p_{\text{fp}} v_{s,t}, & \text{Not detected.} \end{cases} \tag{27}$$

Correlation model derivation:

$$P(z^j_{s,t}|x^k_{s,t}, x^k_{r,t}) = \sum_{\substack{x^j_{r,t} \in \mathcal{R} \\ x^j_s(t) \in \mathcal{S}}} P(z^j_{s,t}|x^j_s(t), x^j_{r,t}) \times P(x^j_s(t), x^j_{r,t}|x^k_{s,t}, x^k_{r,t}). \tag{28}$$

Where $P(x^j_s(t), x^j_{r,t}|x^k_{s,t}, x^k_{r,t})$ is the surface level correlation model.

E Correlation Model Prompt

We use the following example prompt to generate a three-sentence description of an object.
Prompt: `Explain the common use of a toaster in three sentences.`
LLM Response (GPT4o): `A toaster is commonly used to toast slices of bread, making them crispy and golden brown. It's also useful for heating up and adding a slight crunch to bagels, English muffins, and some pastries. Additionally, many people use a toaster to quickly prepare breakfast items like Pop-Tarts or frozen waffles.`

F Correlation Model

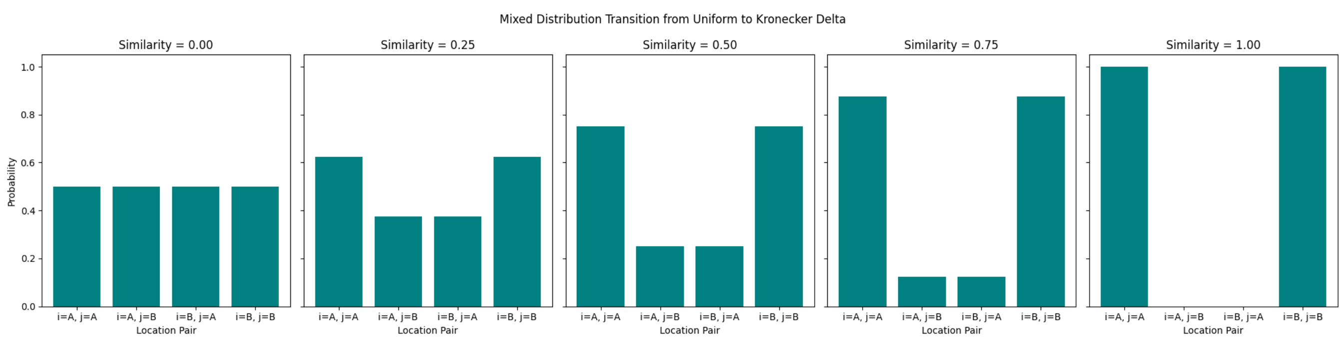


Figure 6: Visualization of $P(x^i_{r,t}|x^j_{r,t})$ when similarity between two objects $\texttt{sim}(i, j) \geq 0$.

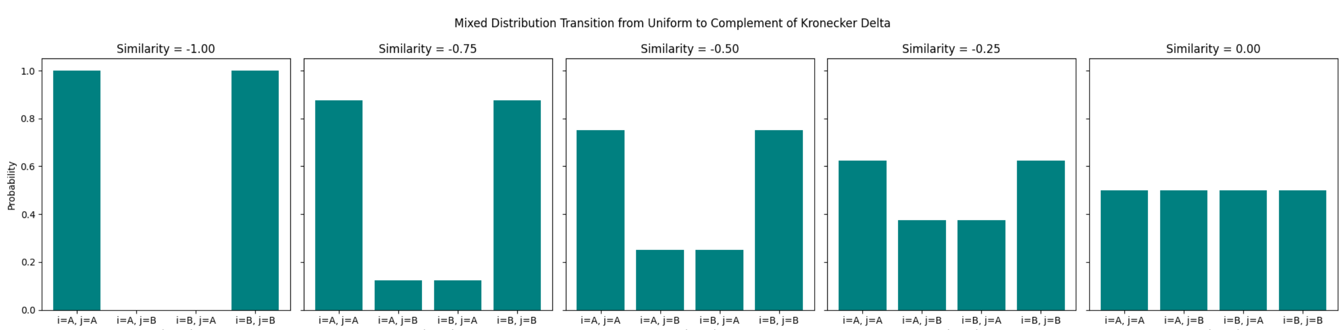


Figure 7: Visualization of $P(x^i_{r,t}|x^j_{r,t})$ when similarity between two objects $\texttt{sim}(i, j) \leq 0$.

G Prompt for Correlation Toggler

System Prompt: `You will be given an object that commonly appears in a typical household environment. Using common sense, determine if the object tends to be distributed throughout a typical household, such as doorknobs and light switches.`
User Prompt: `The following is the object: [object]`
LLM Response (GPT4o): `True / False`

H Occlusion Aware Particle Filter Algorithm

Algorithm 1: Particle filter for object pose estimation

0: **procedure** PARTICLE FILTER($X^k_{p,t-1}, z^k_{p,t}, \overline{z^k_{p,t}}$)
0: $\quad x^k_{p,t} \leftarrow \emptyset, \overline{X}^k_{p,t} \leftarrow \emptyset$
0: $\quad$ **for** $i = 1$ to N **do**
0: $\quad\quad$ sample $x_p(t)^{k,[i]} \sim p(x^k_{p,t} | x^{k,[i]}_{p,t-1})$
0: $\quad\quad$ **if** $z^k_{p,t}$ **then**
0: $\quad\quad\quad w^{k,[i]}_t = p(z^k_{p,t} | x_p(t)^{k,[i]}, x^k_{s,t}, x^k_{r,t})$
0: $\quad\quad$ **else**
0: $\quad\quad\quad w^{k,[i]}_t = p(\overline{z^k_{p,t}} | x_p(t)^{k,[i]}, x^k_{s,t}, x^k_{r,t})$
0: $\quad\quad$ **end if**
0: $\quad\quad \overline{X}^k_{p,t} = \overline{X}^k_{p,t} + \langle x_p(t)^{k,[i]}, w^{k,[i]}_t \rangle$
0: $\quad$ **end for**
0: $\quad$ **for** $i = 1$ to N **do**
0: $\quad\quad$ draw i with probability $\propto w^{k,[i]}_t$
0: $\quad\quad x^k_{p,t} = x^k_{p,t} \cup x_p(t)^{k,[i]}$
0: $\quad$ **end for**
0: **end procedure**=0

Visualization of semantics-aware weight of the particles.

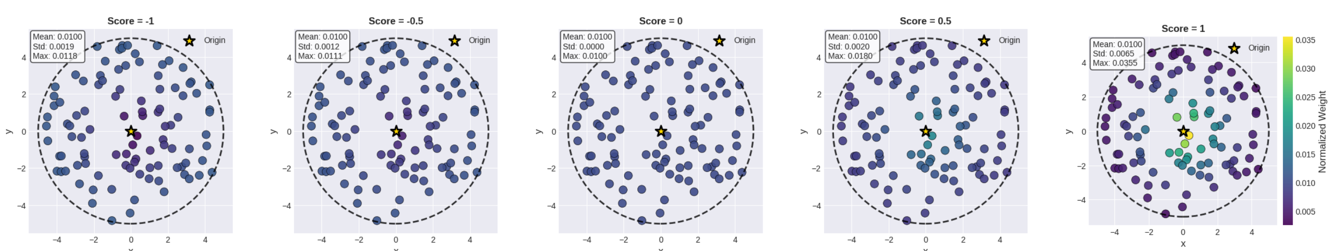


Figure 8: Visualization of w^k_t when object k was not visible but object j was observed.

I Simulation Environment Generation

All experiments are conducted in PyBullet (Coumans and Bai 2016–2021) using a PR2 robot with a constant base velocity of 0.25 m/s on a server equipped with an AMD Ryzen 9 5900 12-core processor.

The Housekeep dataset includes 1,799 object types, 585 surface types, and 15 room types. For each object in a room, 10 human annotators rank the surfaces where a human would most likely place the item, resulting in 45,556 entries. An example of such a data point is as follows:

`Object:` `multiport hub |` **`Room:`** `kitchen`
`Correct` `(ranked): carpet, fridge, table, counter, sink`
`Incorrect` `(ranked): chest, cooktop, microwave, dishwasher, stove`
`Implausible:` `oven, shelf, top cabinet, chair, bottom cabinet`

The *correct* label designates surfaces deemed preferable, the *incorrect* label identifies surfaces where an object is unlikely to be placed, and the *implausible* label refers to surfaces where object placement is considered nearly impossible by humans.

Surfaces with the *correct* label are ranked from most to least likely, while those with the *incorrect* label are ranked from least to most likely.

Since the 10 annotators may have different opinions about object placement (*e.g.*, the most likely to least likely surface placements of a `multiport hub` in a `kitchen`), we combine the annotations by assigning scores to surfaces in the correct and incorrect labels and summing the scores from all 10 annotators. If more than half of the annotators considered a surface implausible, we removed it from the dataset.

For a data point comprising N surfaces, we define a scoring scheme where the maximum attainable score is N and the minimum attainable score is $-N$. Surfaces labeled as *correct* begin with a score of N for the most likely surface, decreasing by 1 for each subsequently less likely surface. In contrast, surfaces labeled as *incorrect* start with a score of $-N$ for the least likely surface, increasing by 1 for each surface deemed more likely. The following example demonstrates this assignment when $N = 15$:

`Object:` `multiport hub |` **`Room:`** `kitchen`
`Correct:` `carpet=15, fridge=14, table=13, counter=12, sink=11`
`Incorrect:` `chest=-15, cooktop=-14, microwave=-13, dishwasher=-12, stove=-11`

To integrate the perspectives of the 10 annotators, we aggregate the scores to produce a cumulative value for each object placement. We then organize the data according to a *Room* $\rightarrow$ *Surface* $\rightarrow$ *Object* hierarchy. Finally, we normalize these aggregated scores to the interval $[0, 1]$, yielding a probabilistic representation of the likelihood that an object will be placed on a specific surface within a particular room.

To construct a ground-truth state of a typical household environment, we randomly sample a predefined number of rooms and the surfaces within each room. We then sample a predefined number of objects for each surface, using the probabilities derived from the aforementioned scoring scheme. An example of a sampled household environment in simulation is shown in Figure 2, where the environment is partitioned into rooms by walls, blue boxes denote surfaces, yellow boxes indicate obstacles, and gray boxes represent objects.

J LGBU Prompt

The following is the prompt used for LLM generated belief update. Note that the variables inside the brackets will change depending on the observation.

System Prompt: `You will receive:`

- `Current belief about where an object might be located.`
- `observation_location: the location that was just inspected.`
- `visibility: how much of the location was visible. 0 means not visible at all, 1 means fully visible.`
- `result: whether the object was found there.`
- `co_detected: other objects found at the same location.`

`Based on this information and common sense, predict where the object is most likely to be now. Choose the most likely location from the given options.`

User Prompt:

- `Current belief about [object_name]: [belief_text]`
- `observation_location: [obs_loc]`
- `visibility: [visibility]`
- `result: [result]`
- `co_detected: [co_detected_text]`

`Given this information, where is object_name most likely to be? [options_text]`

Similar to the MCQA method, the belief is updated using next token prediction, where `[option_text]` is a multiple choice with a character paired with a surface or room.

K Real World Experiment

The HSR robot used in this experiment is a mobile manipulator with 11 degrees of freedom. Object detection is performed using YOLOv8(Reis et al. 2023) based on RGB data from the onboard Xtion PRO LIVE RGB-D camera sensor. To ensure reliable grasps, we employ AnyGrasp(Fang et al. 2023) to resample the grasp before executing the pick action. Computations for YOLOv8, AnyGrasp, and PDDLStream are carried out on a separate laptop equipped with an AMD Ryzen 9 5900HS 8-core processor, 8GB of RAM, and an NVIDIA 3080 GPU.

Both the Baseline and Co-Model methods first invoke the `detect` action on the coffee table, as they begin with a uniform prior distribution across surfaces and rooms. Conversely, MCQA with Co-Model initiates `detect` on the table, informed by the LLM's common sense that the apple is likely to be on the kitchen table. While the Baseline method cannot derive information from other objects, both Co-Model and MCQA with Co-Model detect a similar object (a banana) and adjust the beliefs accordingly, prompting the robot to `detect` the table from a different viewpoint.

We report the initial and goal literals used in the real-world experiment, enumerate the `detect` actions executed by each method, and explain the reasons for any `detect` failures.

```
Initial literal:
(IsItem ?apple),(IsItem ?banana),(IsItem ?cracker_box),
(IsItem ?cereal_box),(IsItem ?screwdriver),
(IsSurf ?coffee_table),(IsSurf ?bench),(IsSurf ?table),
(IsRoom ?living_room),(IsRoom ?kitchen),
(HandEmpty ?arm), (AtBConf ?start)
Goal literal:
(At ?apple ?coffee_table)
```

```
Baseline:
detect ?apple ?coffee_table ?living_room (1)
detect ?apple ?table ?kitchen            (2)
detect ?apple ?bench ?living_room        (3)
detect ?apple ?coffee_table ?living_room (4)
detect ?apple ?table ?kitchen            (5)
Num of Replan: 4
```

- (1): Failed due to the apple not being placed on the coffee table.
- (2): Failed due to occlusion caused by the cracker and cereal boxes.
- (3): Failed due to the apple not being placed on the bench.
- (4): Failed due to apple not being placed on the coffee table.
- (5): Successfully detected the apple.

```
Co-Model:
detect ?apple ?coffee_table ?living_room (1)
detect ?apple ?table ?kitchen            (2)
detect ?apple ?table ?kitchen            (3)
Num of Replan: 2
```

- (1): Failed due to the apple not being placed on the coffee table.
- (2): Failed due to occlusion caused by the cracker and cereal boxes. A banana, similar to an apple, was detected instead, and the correlation model updated the belief accordingly.
- (3): Successfully detected the apple.

```
MCQA with Co-Model:
detect ?apple ?table ?kitchen (1)
detect ?apple ?table ?kitchen (2)
Num of Replan: 1
```

- (1):Failed due to occlusion caused by the cracker and cereal boxes. A banana, similar to an apple, was detected instead, and the correlation model updated the belief accordingly.
- (2): Successfully detected the apple.

L Ablation Studies

	4 rooms 8 surfs	4 rooms 16 surfs	6 rooms 12 surfs	6 rooms 24 surfs	8 rooms 16 surfs	8 rooms 32 surfs
Baseline	1021.5 ± 377.7	1468.8 ± 804.5	1771.6 ± 810.8	2986 ± 1754	2367.3 ± 961.7	4783 ± 3571.1
Co-Model	488.8 ± 201	823.9 ± 524.9	751.3 ± 340.6	1347.2 ± 1069.2	1201.2 ± 545.9	1765.5 ± 943.3
MCQA	466.2 ± 380.69	923.7 ± 1001.7	814.5 ± 942.5	2204.5 ± 2309.9	936.4 ± 1024.4	3690.8 ± 4342.5
MCQA with Co-Model	$\mathbf{362.6 \pm 178.6}$	$\mathbf{753.8 \pm 513.5}$	$\mathbf{566.3 \pm 488.7}$	$\mathbf{992 \pm 975.1}$	$\mathbf{802.4 \pm 899.6}$	$\mathbf{1816 \pm 3015.6}$
LGBU	533.8 ± 299.3	1055 ± 663.5	1461 ± 1852.2	1167.4 ± 758	1056.5 ± 709.6	2162.8 ± 3092.1
MCQA with LGBU	362.8 ± 220.1	926.2 ± 882.7	667.3 ± 795.7	1116.2 ± 1007.9	$547.5^* \pm 370.4$	2005.3 ± 2991.1

Table 2: Cumulative planning and execution time (in seconds). Numbers represent the mean $\pm$ standard deviation calculated over 50 problems.

*Note: LGBU and MCQA+LGBU experiments had an upper cap of total 100 replans, which were exceeded many times. Hence the actual values would be more than the ones reflected in this table

	4 rooms 8 surfs	4 rooms 16 surfs	6 rooms 12 surfs	6 rooms 24 surfs	8 rooms 16 surfs	8 rooms 32 surfs
Baseline	632.5 ± 124.2	715.1 ± 240.5	1205.3 ± 272	1994 ± 602.7	1564.9 ± 362	3017.5 ± 1066.1
Co-Model	121.2 ± 58.4	70.2 ± 198.5	185.1 ± 145.6	355.3 ± 398.7	398.8 ± 272	0
MCQA	99.4 ± 95.3	169.9 ± 306.4	248.2 ± 262.2	1212.6 ± 735.1	134 ± 231	1925.3 ± 1320.8
MCQA with Co-Model	0	0	0	0	0	50.5 ± 834.8
LGBU	164.4 ± 92.9	605.8 ± 288.8	894.8 ± 551.9	509.2 ± 405	787.2 ± 321.3	1066.9 ± 1054.3
MCQA with LGBU	0.2 ± 48.8	339.2 ± 286.5	101.0 ± 210.7	351.1 ± 361.2	288.8 ± 308.3	408.5 ± 871.7

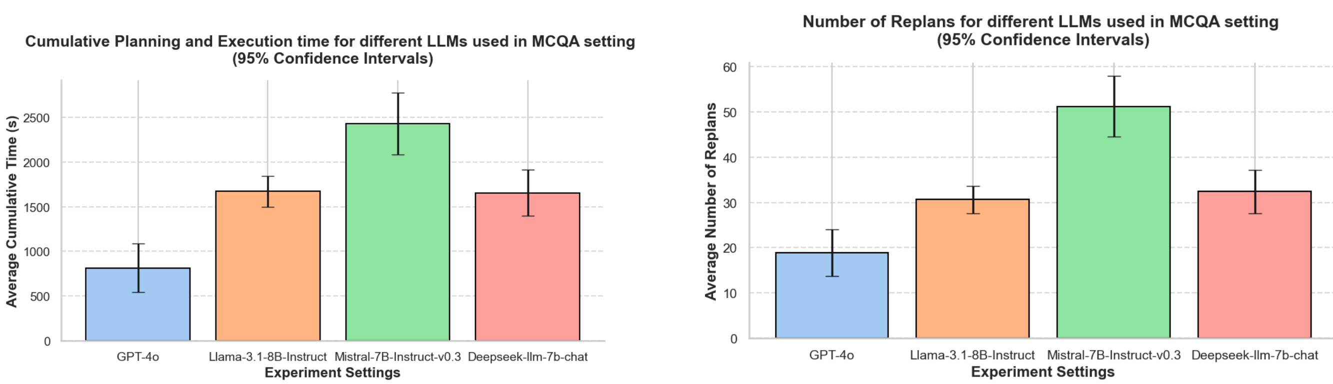
Table 3: Pairwise comparison of cumulative planning and execution time (in seconds) relative to the best-performing method. Numbers represent the mean $\pm$ 95% confidence interval calculated over 50 problems, and 0 indicates the best-performing method.

	4 rooms 8 surfs	4 rooms 16 surfs	6 rooms 12 surfs	6 rooms 24 surfs	8 rooms 16 surfs	8 rooms 32 surfs
Baseline	20.4 ± 8.7	31.7 ± 17.1	33.1 ± 14.8	53.5 ± 26.9	38.9 ± 15.1	70.5 ± 40.6
Co-Model	13.9 ± 6.5	28.0 ± 16.2	21.4 ± 9.3	35.5 ± 16.6	29.7 ± 12.6	45.7 ± 21.8
MCQA	13.3 ± 10.6	$\mathbf{24.6 \pm 21.3}$	18.9 ± 18.1	40.4 ± 32.6	$\mathbf{19.1 \pm 16.5}$	49.1 ± 44.3
MCQA with Co-Model	11.5 ± 6.3	24.9 ± 13.7	$\mathbf{15.9 \pm 10.4}$	$\mathbf{27.6 \pm 13.9}$	20.6 ± 19.8	$\mathbf{35.2 \pm 25.0}$
LGBU	15.0 ± 9.3	33.7 ± 21.7	39.8 ± 49.5	36.1 ± 22.5	27.1 ± 18.5	41.2 ± 26.3
MCQA with LGBU	$\mathbf{10.9 \pm 7.0}$	30.17 ± 25.7	19.1 ± 21.1	30.8 ± 23.4	15.4 ± 11.1	36.2 ± 25.4

Table 4: Number of replans. Numbers represent the mean $\pm$ standard deviation calculated over 50 problems.

	4 rooms 8 surfs	4 rooms 16 surfs	6 rooms 12 surfs	6 rooms 24 surfs	8 rooms 16 surfs	8 rooms 32 surfs
Baseline	9.5 ± 3.0	7.1 ± 6.8	17.2 ± 5.4	25.9 ± 8.8	19.8 ± 6.4	35.3 ± 13.7
Co-Model	3.0 ± 1.9	3.4 ± 5.8	5.5 ± 3.1	7.9 ± 5.5	10.6 ± 6.1	10.5 ± 7.4
MCQA	2.4 ± 2.7	0	2.9 ± 4.9	12.8 ± 8.7	0	13.9 ± 14.1
MCQA with Co-Model	0.6 ± 1.4	0.3 ± 6.6	0	0	1.4 ± 4.1	0
LGBU	4.1 ± 2.9	18.5 ± 10.7	23.9 ± 14.6	17.9 ± 10.1	21.3 ± 8.3	15.5 ± 8.8
MCQA with LGBU	11.2 ± 9.4	0	3.1 ± 5.5	10.2 ± 7.8	6.6 ± 7.0	6.1 ± 8.2

Table 5: Pairwise comparison of number of replans relative to the best-performing method. Numbers represent the mean $\pm$ 95% confidence interval calculated over 50 problems, and 0 indicates the best-performing method.



(a) Planning & execution time.

(b) Number of replans.

Figure 9: Performance metrics over 50 environments in a household layout (6 rooms, 12 surfaces), comparing different LLMs for the MCQA setting. The plots show the average cumulative time (a) and the number of replans (b), both with 95% confidence intervals. We see GPT-4o outperforms the other smaller models, and hence we have used GPT-4o in all other experiments in this paper.

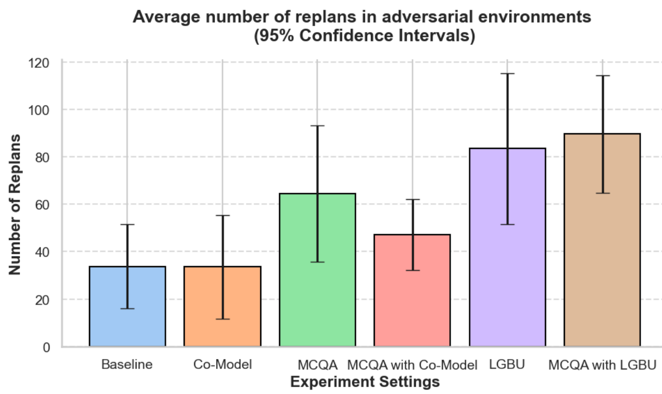


Figure 10: Average number of replans over 5 environments in household layout with 6 rooms and 12 surfaces with 95% confidence interval. Here, we randomize the placements of objects in the home, so they no longer represent the commonsense placements preferred by humans. In this case, we can see that Baseline performs the best, and the use of LLMs worsens performance.
Note: The LGBU and MCQA with LGBU experiments had a total limit of 100 replans, which was exceeded 3 out of 5 times this experiment was run.