Housekeep: Appendix

A Comparison to other existing benchmarks

In table 1, we summarize and compare our work against several works in robotics which model human preferences for assistive robots.

B Data Statistics

In this section we provide details about category level breakdown of objects and receptacles.

B.1 High-level Object and Receptacle Categories

Table 2 details the high-level categorization and frequencies of object and receptacles. We also provide one example of every high-level category, and the original source of the data. We gather 2194 object and receptacle models from multiple sources after filtering objects that are not useful for the task.

Object Filtering Details. We used category-based filtering for ReplicaCAD, and AB datasets (*e.g.* sofa, bikes, etc) to remove unhelpful objects. Then, we removed objects if any of their dimensions exceeded 50 meters. We also used some manual filtering in order to remove very small objects (*e.g.* keychains).

B.2 Low-level Object Categories

Table 3 lists the object categories in each of the train, val-unseen and test-unseen splits. The train split has 8 high-level categories, val-unseen has 2 high-level categories and test-unseen split has 9 high-level categories.

Table 1: Comparison of Housekeep to other rearrangement benchmarks

#	Benchmark	Goal	Object categories	Object models	Scenes	Rooms	Annotators
1	Transport Challenge [22]	Geometric	50	112	**15**	90-120	-
2	Habitat 2.0 [65]	Geometric	41	92	1	111	-
3	Behavior [64]	Predicate	**391**	1217	**15**	100	-
4	VRR [71]	Episodic	118	118	-	**120**	-
5	Taniguchi et al. [66]	Episodic	55	55	1	4	-
6	Jiang et al. [32]	Human Preferences	19	47	-	20	3-5
7	My House, My Rules [33]	Human Preferences	12	12	2	-	75
8	**Housekeep**	Human Preferences	268	**1799**	14	105	**372**

Table 2: **High-level categories.**: This table lists the high-level categories of objects and receptacles and the number of object/receptacle models from each data source for each high-level category

High-level category	**No. of object categories**	**Example**	**No. of models**					
			YCB [83]	R-CAD [65]	iGibson [62]	AB [30]	GSO [54]	Total
Objects								
packaged food	37	condiment	10	3	0	0	48	61
fruit	8	peach	8	0	0	0	0	8
cooking utensil	14	dispensing closure	3	3	0	4	14	24
sanitary	19	bath sheet	2	2	0	1	34	39
crockery	8	tumbler	8	10	0	8	22	48
cutlery	6	plate	4	3	0	0	9	16
tool	14	scissors	11	0	0	0	12	23
stationery	11	invitation card	1	6	0	5	22	34
sporting	8	dumbbell	6	0	0	27	0	33
toy	36	video game	13	0	0	0	282	295
electronic accessory	24	hard drive	0	1	0	45	95	141
storage	18	waste basket	0	2	0	22	33	57
furnishing	3	cushion	0	2	2	222	1	227
decoration	9	string lights	0	2	21	59	51	133
apparel	8	shoe	0	10	0	2	266	278
appliance	23	thermal laminator	0	7	23	215	23	268
kitchen accessory	8	lime squeezer	0	2	0	0	8	10
medical	5	antidepressant	0	0	0	0	66	66
cosmetic	9	face moisturizer	0	0	0	0	38	38
Receptacles								
furniture	17	sofa	0	0	320	0	0	320
appliance	13	fridge	0	0	64	0	0	64
storage	2	basket	0	0	11	0	0	11
Total	268 + 32	-	66	53	441	610	1024	2194

Table 3: Object categories in train, val-unseen and test-unseen splits

	High-level category	Object categories
train	**apparel**	cloth, gloves, handbag, hat, heavy duty gloves, helmet, shoe, umbrella
	appliance	camera, clock, coffeemaker, electric heater, fitness tracker wristband, flashlight, hair dryer, hair straightener, instant camera, lamp, laptop, light bulb, milk frother, portable speaker, router, set-top box, shredder, stand mixer, table lamp, tablet, thermal laminator, toaster, virtual reality viewer
	cooking utensil	blender jar, bundt pan, casserole dish, dispensing closure, dutch oven, pan, pitcher base, pressure cooker, ramekin, saute pan, skillet, skillet lid, spatula, teapot
	cutlery	fork, knife, knife block, plate, saucer, spoon
	decoration	candle holder, lantern, picture frame, plant, plant container, plant saucer, string lights, surface saver ring, vase
	medical	antidepressant, dietary supplement, laxative, medicine, weight loss guide
	packaged food	butter dish, cake mix, cake pan, candy, candy bar, cereal, chocolate, chocolate box, chocolate milk pods, chocolate powder, coffee beans, coffee pods, condiment, cracker box, donut, fondant, fruit snack, gelatin box, heavy master chef can, herring fillets, master chef can, mustard bottle, peppermint, pepsi can pack, pet food supplement, potted meat can, pudding box, salt shaker, snack cake, sparkling water, sugar box, sugar sprinkles, tea can pack, tea pods, tomato soup can, water bottle, xylitol sweetener
	sporting	baseball, dumbbell, dumbbell rack, golf ball, mini soccer ball, racquetball, softball, tennis ball
val-unseen	**kitchen accessory**	can opener, chopping board, dish drainer, honey dipper, lime squeezer, spoon rest, sushi mat, utensil holder
	sanitary	bath sheet, bleach cleanser, diaper pack, dishtowel, dustpan and brush, electric toothbrush, incontinence pads, parchment sheet, sanitary pads, soap dish, soap dispenser, sponge, sponge dish, tampons, toothbrush holder, toothbrush pack, towel, washcloth, wipe warmer
test-unseen	**cosmetic**	beard color gel, beauty pack, face moisturizer, hair color, hair conditioner, lipstick, mascara, skin care product, skin moisturizer
	crockery	bowl, cup, dog bowl, drink coaster, mug, stacking cups, tray, tumbler
	electronic accessory	battery, electronic adapter, electronic cable, graphics card, hard drive, hard drive case, headphones, ink cartridge, keyboard, laptop cover, laptop stand, motherboard, mouse, mouse pad, movie dvd, multiport hub, phone armband case, phone stand, remote control, software cd, tablet holder, tablet stand, usb drive, wireless accessory
	fruit	apple, banana, lemon, orange, peach, pear, plum, strawberry
	furnishing	cushion, neck rest, pillow
	stationery	book, crayon, file sorter, folder, invitation card, labeling tape, large marker, letter holder, paint bottle set, paint maker, pencil case
	storage	backpack, bookend, box, canister, carrying case, cube storage box, desk caddy, easter basket, jar, jewelry box, laundry box, lunch bag, lunch box, paper bag, shoe box, snack dispenser, storage bin, waste basket
	tool	adjustable wrench, anti slip tape, chain, clamp, duct tape, flat screwdriver, hammer, magnifying glass, measuring tape, padlock, phillips screwdriver, power drill, scissors, vinyl tape
	toy	action figure, android figure, balancing cactus, board game, card game, clay, colored wood blocks, dog chew toy, dollhouse toy, fingerpaint, foam brick, hand bell, jenga, lego duplo, nine hole peg test, nintendo switch, peg and hammer toy, puzzle game, rubiks cube, sidewalk chalk, sorting toy, stuffed toy, toy airplane, toy animal, toy basketball, toy bowling set, toy construction set, toy fishing, toy food, toy furniture set, toy instrument, toy kitchen set, toy tool kit, toy vehicle, video game, whale whistle

C AMT Human Preferences Dataset

In this section, we provide more details on our AMT study interface and perform some analysis on the collected data. Our interface consists of an instructions section and is followed by the main task section. After completing the task, the participants are allowed to submit feedback on the interface and the task. The video at https://www.youtube.com/watch?v=BcHmSzoNBYw walks through our AMT data collection interface.

C.1 Participant Instructions

Before beginning the study, each participant is required to read the instructions section. We show the full set of instructions we used during data collection in Figure 1. In our instructions, we describe the tasks that need to be performed to successfully complete a HIT (Human Intelligence Task; an AMT term for a unique task instance). As part of a single HIT, the participants are required to complete 10 sub-tasks. For each sub-task, the participant is given an object, a room and a list of receptacles within the given room. The participant is required to classify these receptacles as `correct`, `misplaced` and `implausible` locations. For the receptacles put into the `correct` and `misplaced` bins, the participant is also required to provide a relative ordering between receptacles.

The instructions section includes an interactive example that the participants can use to practice before they work on the actual tasks. As a part of our instructions, we provide multiple examples of valid responses. We ask the participants to assume the object is in its "base" state (*e.g.* utensils being clean, packaged food being unopened) before making their placement decisions.

C.2 Task Interface

We now describe the task interface in detail. We use the same examples that were used to train the participants.

Task Start: For each sub-task we display an object, a room name and four columns. We show all receptacles to be categorized in the first column, with empty correct and misplaced columns (ranked), and an empty implausible column. The object and receptacles are displayed as rotating animated GIFs. Figure 2 shows a screenshot of our task interface at the start of the task. In this example, the receptacles within the kitchen are to be classified as being the correct, misplaced and implausible locations for the alt shaker.

You need to complete 10 tasks for your hit. For each task you are given:

1. An object (e.g., *salt shaker*)
2. A room (e.g. *kitchen*),
3. A list of receptacles to place objects on in the given room (e.g. *counter*, *table*, *cabinet*).

As part of the task, you will answer questions about where objects might be found before and after cleaning a house:

1. **Before**: What are common locations in this room for the object to be left in a messy house? For example, food is left on the table in the dining room, toys can be scattered on the carpet in the living room.
2. **After**: Where is the object likely to be placed/stored in this room in a clean house? For example, a bowl should be placed in the cabinet of a kitchen.
3. **Implausible**: What are unlikely or impossible locations for the object to be found in any house? For example, you won't find an apple in the bathtub.

To answer these questions:

1. **Drag** the items (receptacles) from the left-most column and place them under one of the following three columns (**Before**, **After**, **Implausible**) depending on where you feel they best fit.
2. Also **rank** the selected receptacles in the **Before** and **After** columns in best at the top, to worst at the bottom order.
3. For completing the task, all items need to be moved to one of the last three columns (**Before**, **After**, **Implausible**).

Now, please feel free to play around with the above interactable example (Example 1) and try placing the items under appropriate columns.

A sample response: Now we discuss how a possible response to the task in Example 1 could look like.

- The correct placements for the *salt shaker* could be the kitchen *counter* and *top cabinet*.
- Also, it makes more sense to place the *salt shaker* on the kitchen *counter* compared to placing it in/on the *bottom cabinet*. So we will place counter higher than bottom cabinet under the **After** column.
- Also, it is likely that the *salt shaker* is misplaced on the kitchen *table*.
- Finally the *salt shaker* will never be placed/misplaced in the *sink*.

We request you to rank the items to the best of your ability, but we do understand that your preferences might vary. We are collecting data to capture the diversity in preferences.

It is ok to have empty columns. In a few cases, it may be possible that it doesn't make sense to move any items in a particular column. Consider the following example:

Example 2: You are given *fork* as the object and *bathroom* as the room.

It is highly unlikely that the *fork* will be placed/misplaced in any of the receptacles within the *bathroom*. So, in this case, you would place all the items under the **Implausible** column. So, **Before** and **After** columns would remain empty.

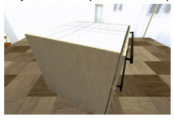
You need to make the following assumptions:

- The object is in a clean/fresh state. For example, if the object given to you is a plate.

object: clean plate, room: kitchen

You must assume that the plate is clean and so, it is less likely to go in a kitchen sink. Similarly, assume that spoons are clean and fruits are fresh.

- Objects can be placed both in/on a receptacle. For example, given a cabinet as the receptacle, objects can be placed both inside it and above it.



Other instructions:

- After completing the task, click on 'Next' to move to the next task. You will be shown the next task with a different object, room and list of receptacles.
- **You WILL NOT BE able to change your responses for a previous task after you hit 'Next'.** So, please ensure that the receptacles are correctly assigned and ranked before moving on to the next task.
- At the end, you will be asked to share your **feedback**. Enter your feedback and hit 'Submit' to complete your hit!

Fig. 1: AMT Instructions page describing the task with illustrative examples.

Fig. 2: AMT starting interface for categorizing and ranking receptacles in the kitchen for a salt shaker.

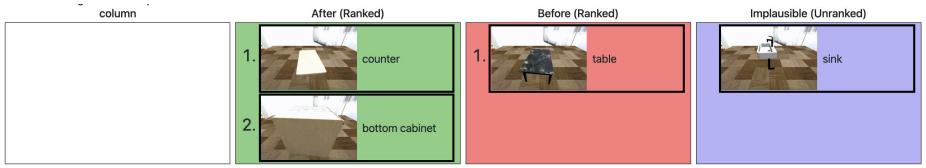


Fig. 3: AMT Example 1: A sample response for salt shaker on receptacles in the kitchen provided as an example to the users.

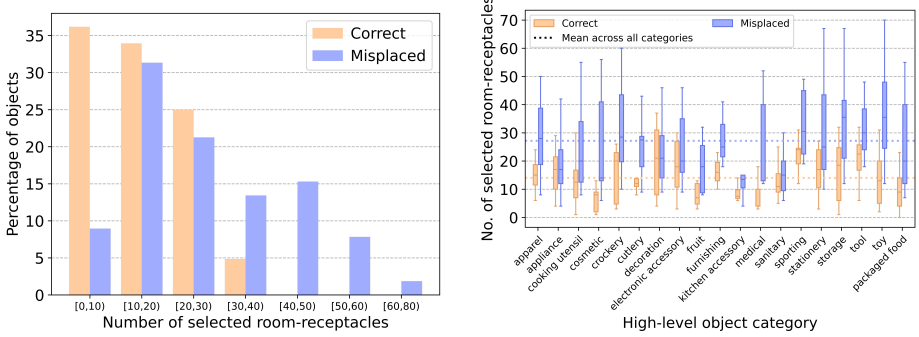
Fig. 4: AMT Example 2: A sample response for clean fork on receptacles in the bathroom.

Sample Response #1: Figure 3 shows a sample response for the task in Figure 2.

Sample Response #2: Now consider the example in Figure 4. Here the given object is fork and the given room is bathroom. Since any receptacle within the bathroom is unlikely to be a correct/misplaced location for fork, all receptacles are placed under the Implausible column.

C.3 Dataset statistics

We collect 10 annotations for each object-room pair. We consider that a room-receptacle (*e.g.* kitchen-sink) is *selected* as being a correct/misplaced location for a given object (*e.g.* sponge) if at least 6 annotators place the receptacle (*e.g.* sink) under the correct/misplaced column when shown the given object-room pair (*e.g.* sponge-kitchen). Figure 5a shows a histogram of objects across different numbers of room-receptacles selected as correct or misplaced. We see that fewer room-receptacles are selected as correct placement of objects while most receptacles are selected as incorrect. Additionally, for most objects (~70%), annotators selected fewer than 20 receptacles across all rooms as correct. On the other hand, annotators tend to select 10-50 receptacles across all rooms as incorrect placements for most objects. This is also confirmed by Figure 5b. It shows the distribution of the number of room-receptacles *selected* as being the correct and misplaced locations. More receptacles are selected as locations where objects are misplaced compared to receptacles where objects are correctly placed.



(a) Histogram of objects across different number of room-receptacles selected as correct or misplaced.

(b) Distribution per high-level category

Fig. 5: Number of room-receptacles selected as Correct and Misplaced.

D Housekeep

D.1 Episode Generation

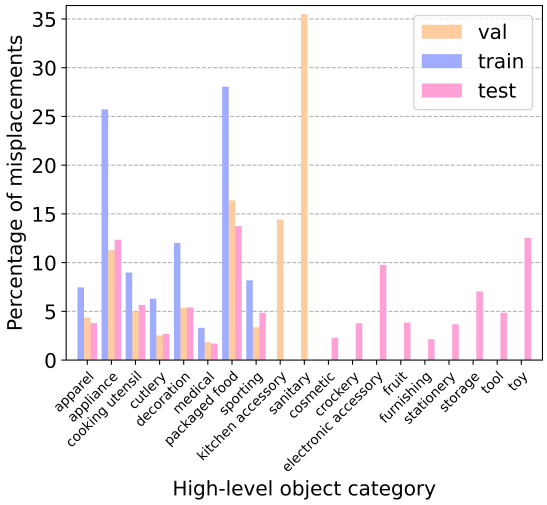
Algorithm 1 provides the logic used to generate an episode in Housekeep. We start with an empty scene `S` furnished with receptacles, AMT data `D`, objects repository `O`. Next, we filter objects by keeping only the ones that have at least one *correct* receptacle in the scene, and remove the others. After initializing an incorrectly placed object, we ensure that the agent is able to rearrange and place it on at least one of the *correct* receptacles. On the other hand, after initializing a correctly placed object, we just ensure that the agent is able to navigate to within grasping distance of it.

Algorithm 1: Dataset Generation

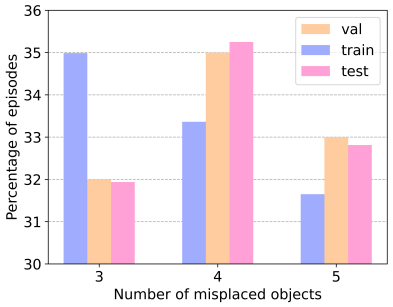
```
import modules: episode E; human-data D; objects O, scene S
input variables: misplaced objects n_m; correct objects n_c
def build_episode(E, D, O, S, n_m, n_c):
    # initialize and load modules
    E.init_empty(), D.load(), S.load(), O.load()
    # keep only objects that have at least one correct receptacle in the scene
    objs = S.filter_objects(O,D)
    # insert misplaced objects
    while len(E.objs) < n_m:
        # sample object to misplace
        obj = S.sample_misplaced_object()
        # get corresponding correct and misplace receptacles
        correct_recs, misplace_recs = S.get_recs(obj)
        # place object for rearrangement, ensure it is solvable
        if E.place(obj, misplace_recs) and E.check_solvable(obj):
            E.register(obj)
    # insert correctly placed objects
    while len(E.objs) < n_m+n_c:
        # sample object to place correctly
        obj = S.sample_placed_object()
        # get correct receptacles only
        correct_recs, _ = S.get_recs(obj)
        # place object on correct receptacle, ensure it is graspable
        if E.place(obj, correct_recs) and E.check_graspable(obj):
            E.register(obj)
```

D.2 Episode statistics

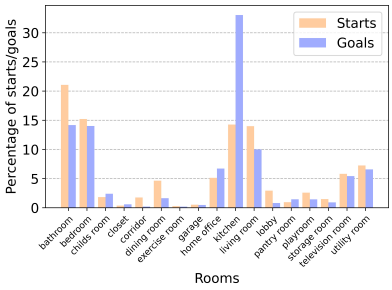
We analyze the generated train, val and test episodes. The val and test episodes include high-level categories already seen in train episodes as well as a few novel high-level categories (Figure 6a). Each episode in the train, val and test splits has $3-5$ misplaced objects. Our val and test episodes have slightly higher percentages of episodes with 4 or 5 misplaced objects compared to train episodes (Figure 6b). A large fraction of the misplaced objects in our episodes start in a bathroom, bedroom, kitchen or living room. A large number of goal receptacles for the misplaced objects are located in the kitchen 6c. This is expected since a large number of misplaced objects in a household



(a) Histogram of misplaced objects in episodes across different high-level object categories



(b) Histogram showing percentage of train, val and test episodes with given number of misplaced objects

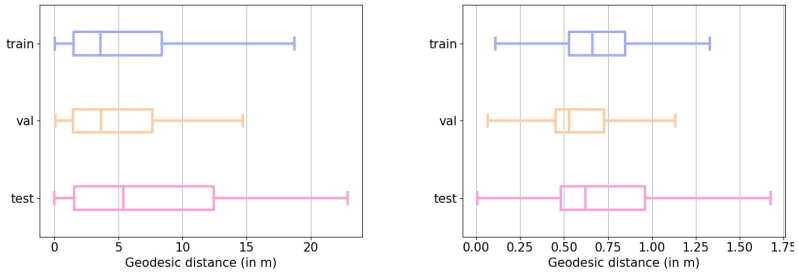


(c) Histogram showing percentage of start and goal positions in each room

Fig. 6: **Episode Statistics.** Analysis on misplaced objects in episodes and their start and goal positions

usually are food or cooking-related (see Figure 6a), and kitchens usually have a large number of receptacles.

Object-Receptacle Distances: Next, we visualize the distribution of geodesic distances from object to correct receptacles across all misplaced objects in all episodes. The median distance in our test episodes is 5.36m (Figure 7a) and the median distance to the closest correct receptacle (out of the 3-5 mispalced) in the test episodes is 0.62m (Figure 7b).



(a) Start to every goal (b) Start to closest goal

Fig. 7: Distribution of geodesic distance from start receptacle to (a) every goal (b) closest goal.

D.3 Formal definitions of metrics

In Section 3.4, we informally described our evaluation metrics for Housekeep. Here, we formally define the metrics for which more rigorous explanations are required.

For a given scene, $\mathcal{R}$ and $\mathcal{O}$ are the set of all receptacles and objects respectively. Given an object $o \in \mathcal{O}$, let c_{or}, m_{or} respectively be the ratio of annotators who placed receptacle $r \in \mathcal{R}$ in `correct` and `misplaced` bins respectively. We call an object *correctly placed* if $c_{or} > 0.5$, and *misplaced* if $m_{or} > 0.5$, where both cannot be simultaneously true. We use:

- $\mathcal{O}_m$ for the set of objects which were *initially misplaced* in the episode.
- $\mathcal{O}_i$ for the set of objects which were *interacted* with by the agent during the episode.
- $\mathcal{O}_{mi}$ ($\mathcal{O}_i \cup \mathcal{O}_m$) for the set of objects *initially misplaced* or *interacted* with by the agent during the episode.

Finally, we define the final placement of the object o at the end of the episode via a mapping function $\Phi : \mathcal{O} \to \mathcal{R}$. The receptacle on which an object $o \in \mathcal{O}$ is placed at the end of the episode is given by $\Phi(o)$

Given the relative change in placement of objects between the start and end states of the episode ($\mathcal{S}_1$ vs $\mathcal{S}_T$), we can formally write the rearrangement metrics as:

1. **Episode Success (`ES`)**: Strict binary (*all* or *none*) metric that is one if and only if all objects are correctly placed, $\texttt{ES}= \prod_{o \in \mathcal{O}} \mathbb{1}[c_{o,\Phi(o)} > 0.5]$.
2. **Object Success (`OS`)**: Fraction of the objects which were *initially misplaced* or *interacted* with by the agent placed correctly at end of the episode, $\texttt{OS}= \sum_{o \in \mathcal{O}_{mi}} \mathbb{1}[c_{o,\Phi(o)} > 0.5]/|\mathcal{O}_{mi}|$.
3. **Soft Object Success (`SOS`)**: The ratio of reviewers that agree that every object *interacted* with or *initially misplaced* is placed correctly averaged across all rearranged objects, $\texttt{SOS}= \sum_{o \in \mathcal{O}_{mi}} c_{o,\Phi(o)}/|\mathcal{O}_{mi}|$. This metric is more lenient because it will be a non-zero number even if just one annotator thought the mapping $(o, \phi(o))$ is correct.
4. **Rearrange Quality (`RQ`)**: The normalized ranking in $(0, 1]$ (via mean reciprocal rank [15]) of the receptacle on which an object is placed, ranked among all correct receptacles of an object, if the object was correctly placed, 0 otherwise, averaged across all *initially misplaced* or *interacted* objects. $\texttt{RQ}= \sum_{o \in \mathcal{O}_{mi}} \mathbb{1}[c_{o,\Phi(o)} > 0.5] mrr_{c_{o,\Phi(o)}}$. Intuitively, RQ will score higher those rearrangements that have a high overall rank in the human preferences dataset.

To formally define Pick and Place Efficiency (PPE), one of our exploration metrics, we need a few extra definitions.

We define $N : \mathcal{O}_i \rightarrow \{1, 2, \cdots\}$ to be a function mapping an object $o \in \mathcal{O}_i$ to the number of times it was *picked* or *placed* by the agent. We similarly define $N_{min} : \mathcal{O}_i \rightarrow \{0, 2\}$ to be the minimum number of picks and places to place an object $o \in \mathcal{O}_i$ in a correct receptacle: it is 2 when $o \in \mathcal{O}_m$ and 0 otherwise.

Pick and Place Efficiency (PPE): The minimum number of interactions needed to rearrange an object divided by the number of interactions the agent actually took to rearrange it if the object was placed in the correct receptacle by the agent at the end of the episode, and 0 if the object was in the incorrect receptacle at the end of the episode, averaged across all objects the agent *interacted* with. $\texttt{PPE} = \sum_{o \in \mathcal{O}_i} \mathbb{1}[c_{o,\phi(o)} > 0.5] \frac{N(o)}{N_{min}(o))} / |\mathcal{O}_i|$

E Agent

We expand on low-level modules used in the agent for navigation and pick-place. We also summarize the planning algorithm in Algorithm 2

Algorithm 2: Planner

```
import modules: rank L; explore E; map M; navigate N; rearrange R; pick-place P
variables: exploration steps n_e; max steps n
def plan(t=0):
    while t < n:  # stop when t=n at any line
        # nothing to rearrange
        if not R.rearrangements():
            # explore for n_e steps
            for i in range(n_e):
                # take an exploration step
                obs = E.act(M, N)
                # update map and rearrange modules
                M.update(obs); R.update(obs)
            t = t + n_e
            R.rescore(L) # update scores using L
        else :
            # rearrange until finished
            for r in R.rearrangements():
                # object and correct receptacle
                obj,rec = r.obj,r.rec
                # nav & pick obj, then nav & place on rec
                if N.nav(obj) & P.pick(obj) & N.nav(rec) & P.place(obj, rec):
                    M.update(obs); R.update(obs)
                t = t+n_r # update steps
```

E.1 Navigation

Navigation (N): Indoor navigation between two points (aka PointNav) is a well-studied problem both in embodied AI [73,77,81] and classical robotics [14,74,76]. Our navigation module takes as input the allocentric map and a goal position (object, receptacle, or frontier), and executes a sequence of low-level base control actions to reach the goal.

E.2 Pick-Place

Raycast for Pick-Place. When invoked, this action casts a ray 1.5m in front of the agent. If the agent is not currently holding an object and this ray intersects with a graspable object, then the object is now "held" by the agent. If the agent is already holding an object and the ray intersects with a receptacle, then the object is placed on that receptacle. Rather than place the object at the point selected on the receptacle, the object is automatically placed on the receptacle.

Pick-Place (P): Our hierarchical baseline picks and places objects via the instance ID of an object or receptacle currently in the view of the agent. The agent then orients itself to face the desired instance ID via look up/down and turn left/right actions. Once the desired instance ID is within the agent's view, the agent calls the ray-cast interaction action. The Pick-Place module fails if the agent is unable to view the object/receptacle of interest or navigate to a place within interaction distance. However, we ensure all episodes are solvable by an oracle agent, so this does not occur in the episodes on which we run our hierarchical baseline. The Pick-Place module can also fail to place an object on a receptacle if sufficient space is not available on the receptacle.

F Approach

F.1 ZS-MLM Prompts

We provide the prompts we try for the ZS-MLM baseline (described in 4.3) for ranking receptacles and rooms in ORR and OR tasks respectively. The prompts are evaluated based on their ability to assign higher scores for the correct placements of objects in the populated iGibson scenes. The best performing prompts are shown in bold.

For ORR task:

- "In <room>, store <object> <spatial-preposition> [mask]"
- "in <room>, put <object> <spatial-preposition> [mask]"
- **"In <room>, usually you put <object> <spatial-preposition> [mask]"**

For OR task:

- "The room where you find <object> is called [mask]"
- "The room where <object> is found is called [mask]"
- "In a house, the room where <object> is found is called [mask]"
- "In a house, the room where you find <object> is called [mask]"

- "In a household, the room where you find <object> is called [mask]"
- "In a household, usually, the room where you find <object> is called [mask]"
- "In a household, usually, you can find <object> in the room called [mask]"
- "In a household, usually, you can find in a room called [mask]"
- "In a household, often, you can find <object> in the room called [mask]"
- "In a household, likely, you can find in the room called [mask]"
- **"In a household, it is likely that you can find <object> in the room called [mask]"**
- "Within a household, the room where you find <object> is called [mask]"
- "Within a household, often times you can find <object> in the room called [mask]"
- "In a house, the room where <object> is kept is called [mask]"
- "In a house, the room where you keep <object> is called [mask]"
- "In a house, the room where <object> is stored is called [mask]"
- "In a house, the room where you store <object> is called [mask]"
- "In a house, the room where <object> is placed is called [mask]"
- "In a house, the room where you place <object> is called [mask]"

F.2 LLM Ranking Module Hyperparameters

In Table 4, we provide the hyperparameters that we use to train the OR and ORR modules using the contrastive matching (CM) strategy. Each method trained using CM is trained on a single GPU for 1000 epochs and we choose the training checkpoint that gives the best mAP score (evaluated as in Section 5.1) on the validation set. In the case of RoBERTa+CM, we use the pretrained roberta-base model and average the last-layer hidden state at all positions (including the CLS token) to obtain the text embeddings.

Table 4: Hyperparameter choices for training the CM modules

#	Hyperparameter	Value
1	Embedding size	768 (RoBERTa) / 300 (GloVe)
2	MLP hidden dimension	512
3	MLP out dimension	512
4	MLP hidden layers	2
5	Batch size	64
6	Optimizer	Adam
7	Learning rate	0.01
8	Weight decay	0.2

Table 5: Evaluation of exploration strategy on `val` split. `RND`: Random, `FWR`: Forward-Right, `FRT`: Frontier

#	Strategy	OS ↑	MC ↑	OC ↑	PDE ↑
1	RND	0.12 ± 0.01	43 ± 1	0.40 ± 0.02	0.22 ± 0.02
2	FWR	0.11 ± 0.01	38 ± 1	0.34 ± 0.02	0.20 ± 0.02
3	FRT	0.26 ± 0.01	86 ± 2	0.76 ± 0.02	0.33 ± 0.02

Table 6: Evaluation of rearrangement ordering on `val` split. `DIS`: DIScovery order, `SCG`: Score Gain, `A-O`: Agent-Object distance, `O-R`: Object-Receptacle distance

#	Order	OS ↑	PDE ↑
1	DIS	0.27 ± 0.01	0.35 ± 0.02
2	SCG	0.26 ± 0.01	0.34 ± 0.02
3	A-O	0.25 ± 0.01	0.32 ± 0.02
4	O-R	0.25 ± 0.01	0.32 ± 0.02

G Additional Experiments

G.1 Exploration Strategies

In Section 4, we discussed the `Explore` module that used frontier exploration (`FRT`). We evaluate 2 additional simple exploration strategies for a total of the following 3 strategies:

- `frontier`: Using the egocentric map we iteratively visit unexplored frontiers, frontiers are defined as the edges between known and unknown space. We keep our implementation details same as those used in [53].
- `random`: Executes a random action in the navigator.
- `forward-right`: Executes the forward action until a collision occurs, then turns right.

As we expect, from Table 5 we see that `FRT` outperforms `RND` and `FWD` in `OS`, exploration and efficiency metrics.

G.2 Planner Ablations

Rearrangement Ordering: In Section 4, when discussing the `Rearrange` submodule, we mentioned 3 key decisions in the submodule. One of them was the order in which misplaced objects are rearranged. In this section, we evaluate the following 4 ordering schemes:

- `score-diff`: We sort rearrangements in decreasing order of score difference between the current receptacle and best one.
- `obj-dist`: We sort rearrangements by the geodesic distance from agent to the object.

Table 7: Evaluation of oracle ranking for ORR and OR tasks on test-unseen split. ORC: Oracle, LM-OR: LLM for OR task, LM-ORR: LLM for ORR task, LM: LLM for both OR and ORR tasks

#	Strategy	OS ↑	MC ↑	OC ↑	PDE ↑
1	ORC	0.65 ± 0.01	74 ± 1	0.74 ± 0.01	0.89 ± 0.01
2	LM-OR	0.63 ± 0.01	75 ± 1	0.76 ± 0.01	0.85 ± 0.01
3	LM-ORR	0.32 ± 0.01	74 ± 1	0.74 ± 0.01	0.45 ± 0.01
4	LM	0.23 ± 0.01	73 ± 1	0.74 ± 0.01	0.35 ± 0.01

- rearrange-dist: We sort rearrangements by the geodesic distance required to execute the rearrangment.
- disc-time: We sort rearrangements by the time of discovery object.

In Table 6, we see that the DIS rearrangement ordering performs slightly better than the other orderings. We choose this ordering to run our main experiments.

Exploration Steps: One of the challenges in Housekeep is balancing the exploration-exploitation trade-off; the agent must explore to find misplaced objects or suitable receptacles, but also must exploit its existing knowledge of where objects belong. The exploration module in our hierarchical baseline has an adjustable parameter n_e that controls the number of steps at the beginning of the episode used for exploration. This parameter thus controls how long the agent spends exploring versus rearranging objects according to a plan.

We find that fewer exploration steps is more effective. If the agent spends too long exploring, then it will not have enough time to rearrange objects before the end of the episode. *e.g.* when $\mathsf{n}_e = 512$, our Object Coverage (OC) is 80%, which is 4 points ahead of the next best n_e. However, its Object Success (OS) is the worst among the variants of n_e we evaluated. We found the best number of exploration steps to be $\mathsf{n}_e = 16$, achieving higher performance in terms of object success (OS) than all $\mathsf{n}_e < 16$ and $\mathsf{n}_e > 16$.

G.3 Ranker Ablations

In Section 4.3, we discuss the OR and ORR tasks that are components of the ranking task. To study the importance of each task in the embodied setting, we decompose the Oracle Ranker in Table 7 which has an episode success (ES) of 35%, and find that using the language model only for object room matching (LM-OR) drops ES only by 7%, whereas using language model only for object-room receptacle matching (LM-ORR) drops ES by 32%. This shows that ORR matching is more important for overall success.

H More Qualitative Analysis

LLM-based Ranker, Compounding Errors. Compared to oracle ranker (Table 3, Row 1) language model (Table 3, Row 3) impacts object success (OS) by -56%, and

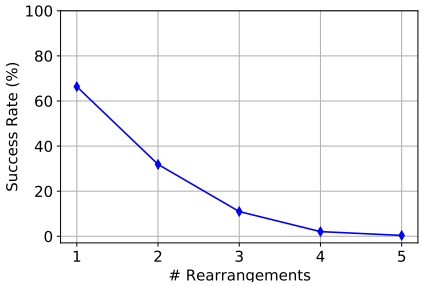


Fig. 8: Episode Success (ES@K) vs. number of rearrangements (K) using non-oracle baseline. As K increases, errors compound, and ES drops.

episode success (ES) by -96%. The dramatic drop in ES is expected as Housekeep is a multi-step problem with compounding errors between rearrangements. That means, with average 4 rearrangements necessary per episode and with OS at 46%, ES will be $0.46^4 \approx 0.045$ as seen. We analyze this in Figure 8 showing that ES@K drops with each successive rearrangement attempt made.

H.1 Agent states and scene layouts

Figure 9 and Figure 10 contain similar plots to the ones in Figure 3 that were discussed in Section 5.3. In particular, we notice that the layout of scene Beechwood_1 is significantly more complex than that of Benevolence_1, which is the cause of the difference between their object discovery plots as discussed in Section 5.3.

I Egocentric rearrangement video

We attach an egocentric video (https://www.youtube.com/watch?v=XccBpQNGN1Q) of the agent successfully rearranging all misplaced objects in an episode. The 3 overlays on the left are, from top to bottom: the depth sensor, instance ID mask with semantic information, and the allocentric top-down occupancy map used by the Mapping module (see Section 4). We also include text logs at the bottom left, showing the object the agent is currently holding, the position and name of the object/receptacle it is navigating towards, the action taken at each step, and whether it is exploring, navigating (rearranging) or picking/placing.

The scene contains 4 misplaced objects: an Easter basket in the utility room table, an electronic adapter and a padlock on the dryer, and a toy vehicle on the sofa. The agent explores until 0:15. It then rearranges the Easter basket, the adapter and the padlock by moving them to a shelf. It completes this rearrangement phase at 1:41, after which it goes back to exploring until 2:07. It then moves the toy vehicle object to a nearby shelf, after which it explores for the remainder of the episode.

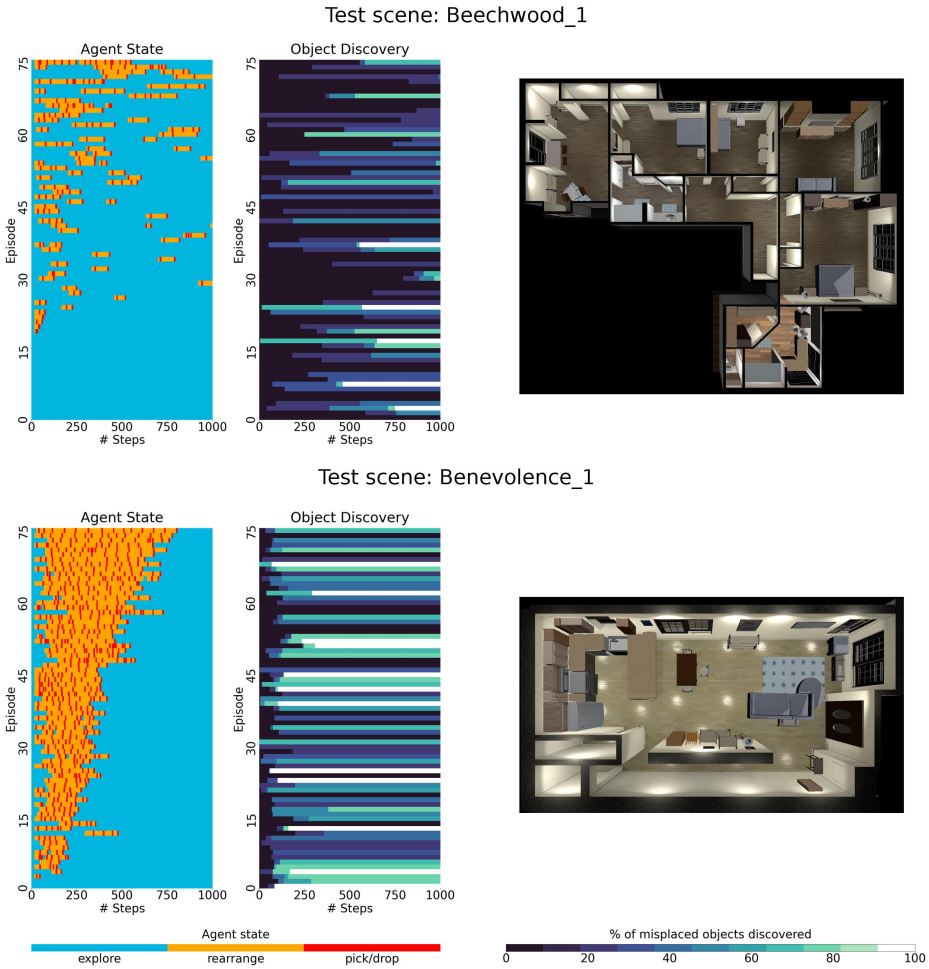


Fig. 9: Left column: visually depicting agent's progress on 75 randomly-sampled episodes from two test scenes, `beechwood_1` and `benevolence_1`. Right column: corresponding test scene layouts.

J Ranking module analysis

For the main results in the paper (Table 1 and Table 2), we used `RoBERTa+CM` as the scoring function. In this section, we analyze the design choices and the performance of our current ranking module.

J.1 Ablations

In Table 8, we analyze the effect of using different features as the language model text embedding. Our results in the paper use features that are globally averaged over all token positions of the language model (`Avg-all`). We perform experiments using the

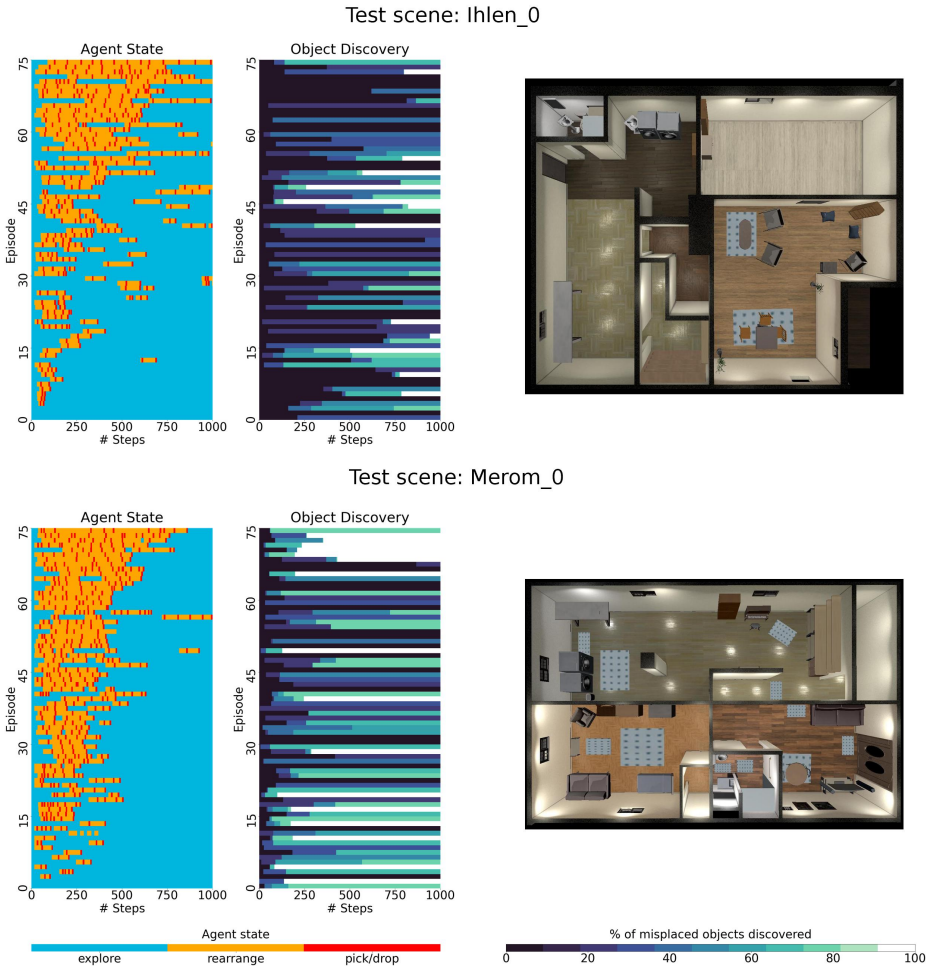


Fig. 10: Left column: visually depicting agent's progress on 75 randomly-sampled episodes from two test scenes, `ihlen_0` and `merom_0`. Right column: corresponding test scene layouts.

Table 8: **Comparison of features.** `ORR` and `OR` results on using different features as text embeddings

	ORR			OR		
# **Features**	train	val-u	test-u	train	val-u	test-u
1 CLS	0.80	0.79	0.79	0.72	0.61	0.66
2 Avg-all-exclude-CLS	0.82	0.79	0.80	1.0	0.61	0.66
3 **Avg-all**	0.81	0.79	0.81	1.0	0.65	0.65

features at CLS token (CLS) and using features averaged at all positions except CLS token (Avg-all-exclude-CLS). While the Avg-all-exclude-CLS features perform close to Avg-all features, using CLS features results in poor performance on seen categories for OR task.

Table 9: **Comparison of language models.** ORR and OR results with different language models

			ORR			OR		
#	Method	# LLM params.	train	val-u	test-u	train	val-u	test-u
1	**RoBERTa-base+CM**	125M	0.81	0.79	0.81	1.0	0.65	0.65
2	GPT2+CM	117M	0.84	0.79	0.83	0.92	0.62	0.64
3	T5-base+CM	220M	0.85	0.82	0.84	0.95	0.69	0.68

Next, we replace the embeddings from RoBERTa-base model with embeddings from GPT-2 and T5-base language models. Note that we use Avg-all features for all language models. We find that using T5-base model results in superior performance on both OR and ORR tasks (Table 9). The T5-base model has nearly double the number of parameters in RoBERTa-base model. We compare to T5-base model because the next smaller model, T5-small has 60 million parameters (half the number of parameters in RoBERTa-base).

J.2 High-level category-wise performance

We now analyze the performance of our RoBERTa+CM scoring function across different high-level categories. We compute mAP scores for OR and ORR tasks (as in Section 5.1) and average them per high-level object category. While the scoring function performs perfectly (mAP=1) on seen categories for the OR task, the OR task performance drops for unseen high-level categories categories (Figure 11). In contrast, the mAP score is close to 0.8 for most seen and unseen high-level categories (Figure 12). The test-unseen high-level categories of fruit, furnishing and cosmetic have low mAP scores for both OR and ORR tasks.

J.3 Generalization to unseen categories

In Table 2, we observed that the Object Success on unseen categories when using the language model-based ranking function is comparable Object Success on seen categories. We now provide qualitative examples showing the performance of our OR and ORR scoring functions on unseen categories.

Figure 13 shows the ranked list of rooms obtained for each object category using our OR ranking function. We also indicate if the room is a valid room for the given

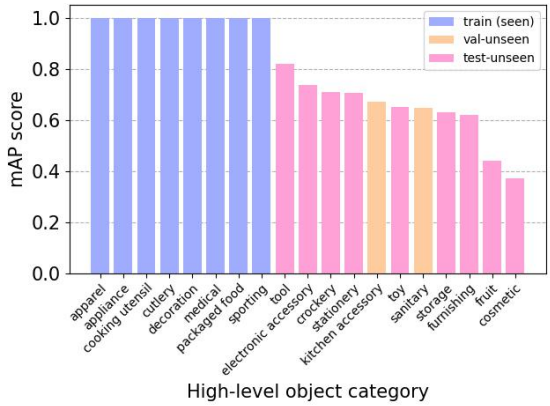


Fig. 11: `OR` performance of `RoBERTa + CM` across different high-level categories

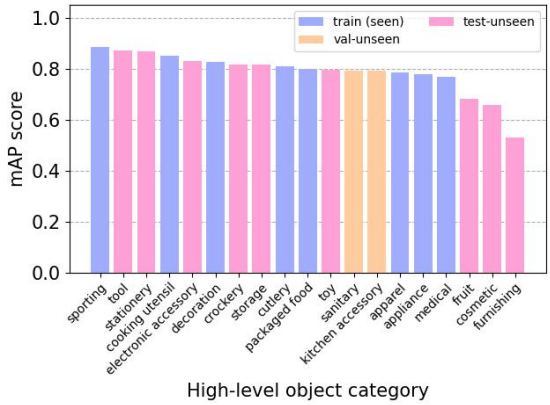


Fig. 12: `ORR` performance of `RoBERTa + CM` across different high-level categories

(a) **Category:** scissors

#	**Ranked list**	**Valid?**
1	kitchen	✓
2	closet	✓
3	playroom	✗
4	utility room	✓
5	dining room	✓
6	bedroom	✓
7	home office	✓
8	garage	✓
9	childs room	✓
10	pantry room	✓
11	bathroom	✓
12	living room	✓
13	television room	✓
14	lobby	✓
15	corridor	✓
16	storage room	✓
17	exercise room	✗

(b) **Category:** large marker

#	**Ranked list**	**Valid?**
1	closet	✓
2	kitchen	✓
3	garage	✓
4	utility room	✓
5	corridor	✓
6	bedroom	✓
7	dining room	✓
8	childs room	✓
9	playroom	✓
10	television room	✓
11	storage room	✓
12	home office	✓
13	living room	✓
14	pantry room	✗
15	bathroom	✓
16	lobby	✓
17	exercise room	✗

(c) **Category:** banana

#	**Ranked list**	**Valid?**
1	kitchen	✓
2	garage	✗
3	utility room	✗
4	closet	✗
5	dining room	✗
6	bedroom	✗
7	childs room	✓
8	pantry room	✗
9	home office	✓
10	storage room	✗
11	living room	✗
12	bathroom	✗
13	television room	✗
14	corridor	✗
15	playroom	✗
16	lobby	✗
17	exercise room	✗

Fig. 13: OR performance for unseen categories

(a) **Category:** scissors **Room:** living room

#	Ranked list	Valid?
1	bottom cabinet	✓
2	shelf	✗
3	chest	✓
4	console table	✗
5	table	✗
6	coffee table	✗
7	stool	✗
8	loudspeaker	✗
9	office chair	✗
10	sofa	✗
11	chair	✗
12	speaker system	✗
13	sofa chair	✗
14	carpet	✗

(b) **Category:** large marker **Room:** corridor

#	Ranked list	Valid?
1	shelf	✓
2	chest	✓
3	washer	✗
4	console table	✗
5	table	✗
6	dryer	✗
7	chair	✗
8	carpet	✗

(c) **Category:** banana **Room:** kitchen

#	Ranked list	Valid?
1	shelf	✓
2	top cabinet	✗
3	bottom cabinet	✗
4	chest	✗
5	counter	✓
6	fridge	✗
7	oven	✗
8	coffee machine	✗
9	sink	✗
10	stove	✗
11	table	✗
12	cooktop	✗
13	carpet	✗
14	dishwasher	✗
15	chair	✗
16	microwave	✗

Fig. 14: ORR performance for unseen categories

object. Recall that a room is considered valid if it contains at least one receptacle that is deemed `correct` by at least 6/10 annotators. While the ranked lists for scissors (a tool) and large marker (stationery) have the valid rooms on top, a few valid rooms are further down in the list for banana (fruit category).

Figure 14 shows the ranked list of receptacles with the room for the given object-room pair. These ranked lists are obtained using the `ORR` ranking function. We indicate if the receptacle is a valid receptacle next to the receptacle's name. For the shown examples, most of the valid receptacles are on top of the ranked lists.

References

1. Abdo, N., Stachniss, C., Spinello, L., Burgard, W.: Robot, organize my shelves! tidying up objects by predicting user preferences. 2015 IEEE International Conference on Robotics and Automation (ICRA) (2015)
2. Agrawal, H., Chandrasekaran, A., Batra, D., Parikh, D., Bansal, M.: Sort story: Sorting jumbled images and captions into stories. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (2016)
3. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al.: On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757 (2018)
4. Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I.D., Gould, S., van den Hengel, A.: Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018 (2018)
5. Armeni, I., He, Z., Zamir, A.R., Gwak, J., Malik, J., Fischer, M., Savarese, S.: 3d scene graph: A structure for unified semantics, 3d space, and camera. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019 (2019)
6. Batra, D., Chang, A.X., Chernova, S., Davison, A.J., Deng, J., Koltun, V., Levine, S., Malik, J., Mordatch, I., Mottaghi, R., Savva, M., Su, H.: Rearrangement: A challenge for embodied ai (2020)
7. Batra, D., Gokaslan, A., Kembhavi, A., Maksymets, O., Mottaghi, R., Savva, M., Toshev, A., Wijmans, E.: Objectnav revisited: On evaluation of embodied agents navigating to objects. arXiv preprint arXiv:2006.13171 (2020)
8. Bhagavatula, C., Bras, R.L., Malaviya, C., Sakaguchi, K., Holtzman, A., Rashkin, H., Downey, D., Yih, W., Choi, Y.: Abductive commonsense reasoning. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020 (2020)
9. Bisk, Y., Zellers, R., LeBras, R., Gao, J., Choi, Y.: PIQA: reasoning about physical commonsense in natural language. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020 (2020)
10. Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., Choi, Y.: COMET: Commonsense transformers for automatic knowledge graph construction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
11. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (2020)
12. Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., Dollar, A.M.: The ycb object and model set: Towards common benchmarks for manipulation research. In: 2015 international conference on advanced robotics (ICAR). IEEE (2015)

13. Cartillier, V., Ren, Z., Jain, N., Lee, S., Essa, I., Batra, D.: Semantic mapnet: Building allocentric semanticmaps and representations from egocentric views. arXiv preprint arXiv:2010.01191 (2020)
14. Chan, S.H., Wu, P.T., Fu, L.C.: Robust 2d indoor localization through laser slam and visual slam fusion. 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC) pp. 1263–1268 (2018)
15. Craswell, N.: Mean reciprocal rank. In: Encyclopedia of Database Systems (2009)
16. Crowston, K.: Amazon mechanical turk: A research tool for organizations and information systems scholars. In: Shaping the future of ict research. methods and approaches (2012)
17. Daruna, A., Liu, W., Kira, Z., Chernova, S.: Robocse: Robot common sense embedding (2019)
18. Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., Batra, D.: Embodied question answering. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018 (2018)
19. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (2019)
20. Ehsani, K., Han, W., Herrasti, A., VanderBilt, E., Weihs, L., Kolve, E., Kembhavi, A., Mottaghi, R.: ManipulaTHOR: A framework for visual object manipulation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2021)
21. Fleiss, J., et al.: Measuring nominal scale agreement among many raters. Psychological Bulletin **76**(5) (1971)
22. Gan, C., Schwartz, J.I., Alter, S., Schrimpf, M., Traer, J., de Freitas, J.L., Kubilius, J., Bhandwaldar, A., Haber, N., Sano, M., Kim, K., Wang, E., Mrowca, D., Lingelbach, M., Curtis, A., Feigelis, K.T., Bear, D.M., Gutfreund, D., Cox, D., DiCarlo, J.J., McDermott, J., Tenenbaum, J., Yamins, D.L.K.: Threedworld: A platform for interactive multi-modal physical simulation. NeurIPS **abs/2007.04954** (2020)
23. Gordon, D., Kembhavi, A., Rastegari, M., Redmon, J., Fox, D., Farhadi, A.: IQA: visual question answering in interactive environments. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018 (2018)
24. Granroth-Wilding, M., Clark, S.: What happens next? event prediction using a compositional neural network model. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA (2016)
25. Habitat: Habitat Challenge (2021), `https://aihabitat.org/challenge/2021/`
26. Hill, F., Mokra, S., Wong, N., Harley, T.: Human instruction-following with deep reinforcement learning via transfer-learning from text. ArXiv **abs/2005.09382** (2020)
27. Hong, Y., Wu, Q., Qi, Y., Rodriguez-Opazo, C., Gould, S.: A recurrent vision-and-language BERT for navigation. In: ECCV (2021)
28. Hu, X., Yin, X., Lin, K., Wang, L., Zhang, L., Gao, J., Liu, Z.: Vivo: Surpassing human performance in novel object captioning with visual vocabulary pre-training. ArXiv **abs/2009.13682** (2020)
29. Huang, W., Abbeel, P., Pathak, D., Mordatch, I.: Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. ArXiv **abs/2201.07207** (2022)

30. Jasmine, C., Shubham, G., Achleshwar, L., Leon, X., Kenan, D., Xi, Z., F, Y.V.T., Himanshu, A., Thomas, D., Matthieu, G., Jitendra, M.: Abo: Dataset and benchmarks for real-world 3d object understanding. arXiv preprint arXiv:2110.06199 (2021)
31. Jiang, J., Zheng, L., Luo, F., Zhang, Z.: Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation. arXiv preprint arXiv:1806.01054 (2018)
32. Jiang, Y., Lim, M., Saxena, A.: Learning object arrangements in 3d scenes using human context. In: Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012 (2012)
33. Kapelyukh, I., Johns, E.: My house, my rules: Learning tidying preferences with graph neural networks. In: CoRL (2021)
34. Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Gordon, D., Zhu, Y., Gupta, A., Farhadi, A.: AI2-THOR: An Interactive 3D Environment for Visual AI. arXiv (2017)
35. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. biometrics (1977)
36. Levesque, H.J., Davis, E., Morgenstern, L.: The winograd schema challenge. In: KR (2011)
37. Li, S., Puig, X., Du, Y., Wang, C., Akyürek, E., Torralba, A., Andreas, J., Mordatch, I.: Pre-trained language models for interactive decision-making. ArXiv **abs/2202.01771** (2022)
38. Li, X., Yin, X., Li, C., Hu, X., Zhang, P., Zhang, L., Wang, L., Hu, H., Dong, L., Wei, F., Choi, Y., Gao, J.: Oscar: Object-semantics aligned pre-training for vision-language tasks. In: ECCV (2020)
39. Liu, W., Bansal, D., Daruna, A., Chernova, S.: Learning Instance-Level N-Ary Semantic Knowledge At Scale For Robots Operating in Everyday Environments. In: Proceedings of Robotics: Science and Systems (2021)
40. Liu, W., Paxton, C., Hermans, T., Fox, D.: Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. arXiv preprint arXiv:2110.10189 (2021)
41. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 (2019)
42. Lu, J., Batra, D., Parikh, D., Lee, S.: Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada (2019)
43. Lu, K., Grover, A., Abbeel, P., Mordatch, I.: Pretrained transformers as universal computation engines. ArXiv **abs/2103.05247** (2021)
44. Majumdar, A., Shrivastava, A., Lee, S., Anderson, P., Parikh, D., Batra, D.: Improving vision-and-language navigation with image-text pairs from the web. ArXiv **abs/2004.14973** (2020)
45. Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P., Allen, J.: A corpus and cloze evaluation for deeper understanding of commonsense stories. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2016)
46. Moudgil, A., Majumdar, A., Agrawal, H., Lee, S., Batra, D.: Soat: A scene-and object-aware transformer for vision-and-language navigation. Advances in Neural Information Processing Systems **34** (2021)

47. Narasimhan, M., Wijmans, E., Chen, X., Darrell, T., Batra, D., Parikh, D., Singh, A.: Seeing the un-scene: Learning amodal semantic maps for room navigation. CoRR **abs/2007.09841** (2020)
48. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
49. Padmakumar, A., Thomason, J., Shrivastava, A., Lange, P., Narayan-Chen, A., Gella, S., Piramithu, R., Tur, G., Hakkani-Tür, D.Z.: Teach: Task-driven embodied agents that chat. ArXiv **abs/2110.00534** (2021)
50. Pennington, J., Socher, R., Manning, C.: GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014)
51. Petroni, F., Lewis, P., Piktus, A., Rocktäschel, T., Wu, Y., Miller, A.H., Riedel, S.: How context affects language models' factual predictions. In: Automated Knowledge Base Construction (2020)
52. Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., Miller, A.: Language models as knowledge bases? In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (2019)
53. Ramakrishnan, S.K., Jayaraman, D., Grauman, K.: An exploration of embodied visual exploration (2020)
54. Research, G.: Google Scanned Objects. `https://app.ignitionrobotics.org/GoogleResearch/fuel/collections/Google%20Scanned%20Objects` (2020), [Online; accessed Feb-2022]
55. Roberts, A., Raffel, C., Shazeer, N.: How much knowledge can you pack into the parameters of a language model? In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2020)
56. robotics, F.: Fetch. `http://fetchrobotics.com/` (2020)
57. Sakaguchi, K., Le Bras, R., Bhagavatula, C., Choi, Y.: Winogrande: An adversarial winograd schema challenge at scale. In: AAAI (2020)
58. Salganik, M.J.: Bit by Bit: Social Research in the Digital Age. Open review edition edn. (2017)
59. Sap, M., Bras, R.L., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N.A., Choi, Y.: ATOMIC: an atlas of machine commonsense for if-then reasoning. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019 (2019)
60. Sap, M., Rashkin, H., Chen, D., Le Bras, R., Choi, Y.: Social IQa: Commonsense reasoning about social interactions. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (2019)
61. Savva, M., Malik, J., Parikh, D., Batra, D., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V.: Habitat: A platform for embodied AI research. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019 (2019)
62. Shen, B., Xia, F., Li, C., Martín-Martín, R., Fan, L., Wang, G., Buch, S., D'Arpino, C., Srivastava, S., Tchapmi, L.P., et al.: igibson, a simulation environment for interactive tasks in large realistic scenes. arXiv preprint arXiv:2012.02924 (2020)

63. Shridhar, M., Thomason, J., Gordon, D., Bisk, Y., Han, W., Mottaghi, R., Zettlemoyer, L., Fox, D.: ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020 (2020)
64. Srivastava, S., Li, C., Lingelbach, M., Mart'in-Mart'in, R., Xia, F., Vainio, K., Lian, Z., Gokmen, C., Buch, S., Liu, C.K., Savarese, S., Gweon, H., Wu, J., Fei-Fei, L.: Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In: CoRL (2021)
65. Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D.S., Maksymets, O., et al.: Habitat 2.0: Training home assistants to rearrange their habitat. Advances in Neural Information Processing Systems **34** (2021)
66. Taniguchi, A., Isobe, S., Hafi, L.E., Hagiwara, Y., Taniguchi, T.: Autonomous planning based on spatial concepts to tidy up home environments with service robots. Advanced Robotics **35** (2021)
67. Thomason, J., Murray, M., Cakmak, M., Zettlemoyer, L.: Vision-and-dialog navigation. CoRL (2019)
68. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA (2017)
69. Wang, W., Bao, H., Dong, L., Wei, F.: Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. ArXiv **abs/2111.02358** (2021)
70. Wani, S., Patel, S., Jain, U., Chang, A.X., Savva, M.: Multion: Benchmarking semantic map memory using multi-object navigation. In: NeurIPS (2020)
71. Weihs, L., Deitke, M., Kembhavi, A., Mottaghi, R.: Visual room rearrangement. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2021)
72. Wijmans, E., Datta, S., Maksymets, O., Das, A., Gkioxari, G., Lee, S., Essa, I., Parikh, D., Batra, D.: Embodied question answering in photorealistic environments with point cloud perception. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019 (2019)
73. Wijmans, E., Kadian, A., Morcos, A., Lee, S., Essa, I., Parikh, D., Savva, M., Batra, D.: DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020 (2020)
74. Wu, P.T., Yu, C.A., Chan, S.H., Chiang, M.L., Fu, L.C.: Multi-layer environmental affordance map for robust indoor localization, event detection and social friendly navigation. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 2945–2950 (2019). https://doi.org/10.1109/IROS40897.2019.8968455
75. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: cira. vol. 97 (1997)
76. Yan, W., Weber, C., Wermter, S.: Learning indoor robot navigation using visual and sensorimotor map information. Frontiers in Neurorobotics **7** (2013). https://doi.org/10.3389/fnbot.2013.00015, https://www.frontiersin.org/article/10.3389/fnbot.2013.00015
77. Ye, J., Batra, D., Wijmans, E., Das, A.: Auxiliary tasks speed up learning pointgoal navigation. ArXiv **abs/2007.04561** (2020)

78. Zellers, R., Bisk, Y., Farhadi, A., Choi, Y.: From recognition to cognition: Visual commonsense reasoning. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019 (2019)
79. Zellers, R., Bisk, Y., Schwartz, R., Choi, Y.: SWAG: A large-scale adversarial dataset for grounded commonsense inference. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (2018)
80. Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., Choi, Y.: HellaSwag: Can a machine really finish your sentence? In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
81. Zhao, X., Agrawal, H., Batra, D., Schwing, A.: The Surprising Effectiveness of Visual Odometry Techniques for Embodied PointGoal Navigation. In: ICCV (2021)
82. Zhou, B., Khashabi, D., Ning, Q., Roth, D.: "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (2019)
83. Çalli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., Dollar, A.: The ycb object and model set: Towards common benchmarks for manipulation research. 2015 International Conference on Advanced Robotics (ICAR) (2015)