

Break Out the Silverware: Semantic Understanding of Stored Household Items

Michaela Levi Richter

Bar Ilan University, Computer Science
Department
Ramat Gan, Israel
michaela.levi20@gmail.com

Reuth Mirsky

Tufts University, Department of
Computer Science
Medford, MA, USA
reuth.mirsky@tufts.edu

Oren Glickman

Bar Ilan University, Computer Science
Department
Ramat Gan, Israel
oren.glickman@biu.ac.il

ABSTRACT

“Bring me a plate.” For domestic service robots, this simple command reveals a complex challenge: inferring where everyday items are stored, often out of sight in drawers, cabinets, or closets. Despite advances in vision and manipulation, robots still lack the commonsense reasoning needed to complete this task. We introduce the **Stored Household Item Challenge**, a benchmark task for evaluating service robots’ cognitive capabilities: given a household scene and a queried item, predict its most likely storage location. Our benchmark includes two datasets: (1) a real-world evaluation set of 100 item-image pairs with human-annotated ground truth from participants’ kitchens, and (2) a development set of 6,500 item-image pairs annotated with storage polygons over public kitchen images. These datasets support realistic modeling of household organization and enable comparative evaluation across agent architectures. To begin tackling this challenge, we introduce **NOAM (Non-visible Object Allocation Model)**, a hybrid agent pipeline that combines structured scene understanding with large language model inference. NOAM converts visual input into natural language descriptions of spatial context and visible containers, then prompts a language model (e.g., GPT-4) to infer the most likely hidden storage location. This integrated vision-language agent exhibits emergent commonsense reasoning and is designed for modular deployment within broader robotic systems. We evaluate NOAM against baselines including random selection, vision-language pipelines (Grounding-DINO + SAM), leading multimodal models (e.g., Gemini, GPT-4o, Kosmos-2, LLaMA, Qwen), and human performance. NOAM significantly improves prediction accuracy and approaches human-level results, highlighting best practices for deploying cognitively capable agents in domestic environments.

CCS CONCEPTS

- Computing methodologies → Knowledge representation and reasoning; *Computer vision; Artificial intelligence*.

KEYWORDS

Semantic Understanding, Commonsense Reasoning, Benchmark Datasets, Vision-Language Models, Robotics

1 INTRODUCTION

Service robots are increasingly capable of navigating and manipulating objects in household environments. Yet, they remain limited in a fundamental way: commonsense reasoning about where items are typically stored when those items are not visible. For humans, such inference is effortless. When asked to fetch a mug, we do not



Figure 1: Kitchen scene shown to Gemini, with its predicted spoon storage location highlighted in the bounding box.

search the refrigerator or under the sink; instead, we rely on deeply shared expectations about household organization. Translating this intuition into machine reasoning remains an unsolved problem.

We introduce the **Stored Household Item Challenge**, a new benchmark designed to evaluate a specific form of commonsense reasoning: predicting the likely storage location of household items that are not visible. Given an indoor scene and a queried item, the task is to infer which storage container (such as a drawer, cabinet, or closet) is most likely to contain the item, despite it being hidden from view. Unlike traditional object detection, which focuses on visible entities, this task requires reasoning about concealed items based on indirect visual cues and prior semantic knowledge. We hypothesize that household storage follows consistent patterns (for example, mugs are usually stored in upper cabinets rather than drawers) that can be learned and generalized.

Current benchmarks overlook this reasoning dimension. Vision-Language Models (VLMs) and object detectors perform well on visible items but fail when asked to infer the invisible. For instance, we asked Gemini and GPT to identify where a spoon is most likely stored, using the following prompt:

In the given image, which drawer or cabinet is the most likely place to store a spoon?
Return the image with the exact drawer or cabinet clearly marked. Do not modify any other part of the image.



Figure 2: Kitchen scene shown to GPT, with its predicted spoon storage location highlighted in the bounding box.

As shown in Figures 1 and 2, both models struggled to select a plausible container. To our knowledge, no prior structured benchmark exists for this ability. Existing datasets lack labels for expected-but-not-visible items, and current models are not trained to reason about functional storage norms or household organization. This gap highlights both a scientific challenge and a practical barrier for embodied agents operating in real homes.

To support this task, we construct two datasets: (1) a 6,500-pair development set based on real kitchen scenes (SUN dataset), with container annotations by humans; and (2) a 100-example evaluation set collected from real homes. These datasets enable scalable training and robust testing across varied environments. We benchmark random baselines, VLMs, Multimodal Large Language Models (MLLMs), and human predictions (Figure 3, top and middle left).

We also introduce **NOAM** (Non-visible Object Allocation Model), a novel language-based pipeline that reframes the task as structured text-based reasoning (Figure 3, bottom left). By describing scenes and candidate storage options in natural language and querying LLMs, NOAM significantly outperforms VLMs, though still below human performance. This suggests that language-driven inference offers a promising path for equipping agents with commonsense reasoning skills.

Rather than chasing marginal detection gains, our aim is to formalize a reasoning capability absent in today’s models: context-aware inference about what is unseen but likely. The *Stored Household Item Challenge* captures this missing skill that is crucial for robots in cluttered, unfamiliar spaces. As LLMs evolve, we expect progress, but the challenge will remain a meaningful test of real-world understanding.

To summarize, the contributions of this work are threefold:

1. New challenge: the Stored Household Item Challenge.
2. An annotation pipeline and dataset of item-image pairs grounded in real household scenes.
3. NOAM (Non-visible Object Allocation Model), a language-based model that reframes visual inference as a structured textual reasoning task. We compare it against baselines, including random choice, VLM, MLLMs and humans.

2 RELATED WORK

Our research lies at the intersection of three areas: domestic service robots, scene understanding, and object detection. While our work introduces a new approach to reasoning about hidden items, it builds on established methods across these domains, particularly recent advances in vision, vision-language, and MLLMs to enhance semantic understanding in complex domestic environments.

Domestic Service Robots. Research on cleaning robots has expanded beyond vacuuming to include service robots capable of tidying, organizing groceries, washing dishes, and setting tables. Prior work focuses on learning processes tailored to specific homes [14, 20, 37] and on user-instructed behaviors [16, 19, 28, 35, 41, 42, 46]. Some approaches use GNNs to learn user preferences from observations [18] or exploit object hierarchies and collaborative filtering [2]. However, most pick-and-place tasks involve visible surfaces like shelves or tables [15, 38, 45], whereas we target items stored in enclosed containers.

Liu et al. [25] proposed a knowledge-based framework for object search using ontologies and multi-domain knowledge. Ramrakhyा et al. [34] define a Semantic Placement (SP) task for predicting object placements, but focus on visible items. Our task generalizes to any household object and emphasizes hidden storage.

Kant et al. [17] use commonsense reasoning to rearrange objects in simulation, but rely on partial observability and do not target concealed items. Kurenkov [24] assumes access to prior environment layouts when searching in containers. Our method aims to reduce robot deployment time by enabling inference without prior maps. Additional work explores contextual semantics for continuous sweeping [3] or for organizing items by user preferences [4, 17, 41, 42, 45]. Some methods adapt NLP to guide object arrangement [15], which helps integrate LLMs to generalize storage norms even without explicit user preferences.

While our method is currently designed and evaluated outside the context of physical robots, our broader vision is to support real-world domestic service robots that reason about storage in a human-aligned way. Although we have not yet tested our system on physical robots, this application remains a key motivation and long-term goal for our research.

Scene understanding involves analyzing visual scenes through both geometric and semantic context, along with inter-object relationships [30]. This holistic understanding is essential for robots to collaborate effectively with humans [1], going beyond object classification to functional comprehension [47].

Subtasks such as scene classification help label scene components [32, 40], and assistive robotics research focuses on reducing computational overhead for intuitive interfaces [6].

Scene understanding also intersects with Simultaneous Localization and Mapping (SLAM), which enables robots to build maps of unknown environments while localizing themselves [7, 8, 12, 39]. Incorporating semantics into SLAM allows robots to reason about object meaning and location together, improving navigational accuracy and object localization.

Object detection identifies semantic instances in images or video [49]. Early systems used handcrafted features, but deep learning now enables models to learn discriminative features through improved architectures, training, and optimization [48].

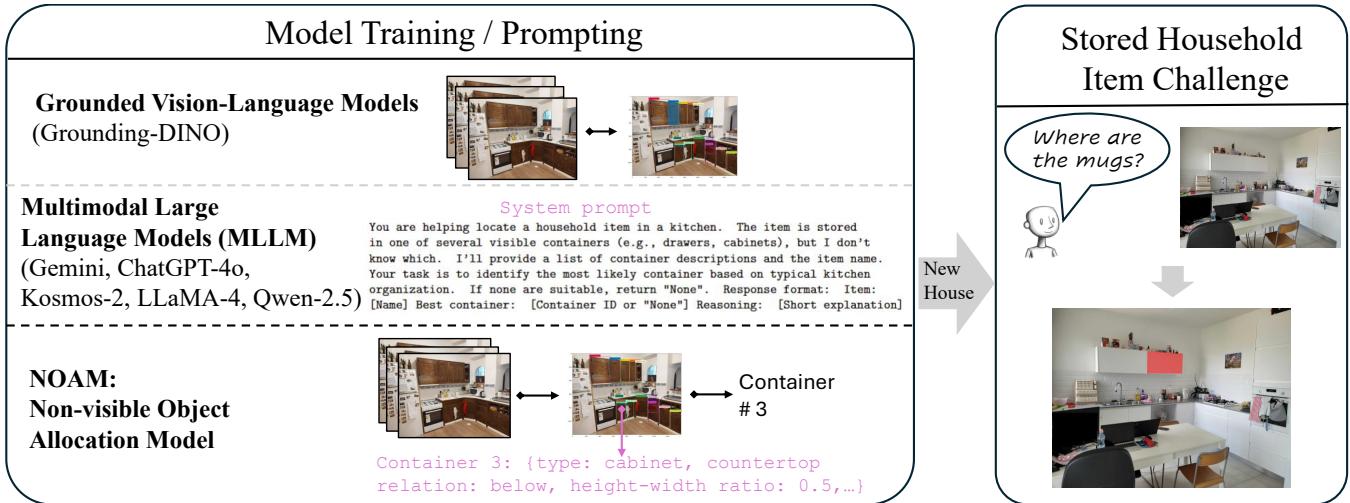


Figure 3: Depiction of the Stored Household Item Challenge (right). We compare the performance of various models on this task (left), including grounded vision-language models, multimodal LLMs, and our model, NOAM.

Our task involves segmentation, including semantic segmentation [29] and instance segmentation [13]. We use Grounding-DINO for open-set object detection via text prompts [26], and Segment Anything Model (SAM) for segmentation [21]. Performance is evaluated using accuracy and intersection over union (IoU) metrics [5]. Concealed Object Detection (COD), which segments camouflaged objects [11], is not applicable here, as it assumes visibility. Similarly, occlusion detection focuses on partially visible objects in controlled settings [36], whereas we target fully hidden items. Work in areas like prisons or airports uses thermal cameras for detecting occluded objects [23], but these technologies are unsuitable for locating stored household items.

Recent advances in zero-shot object detection further leverage VLMs and MLLMs to detect novel object categories using natural language prompts [27].

In our experiments, we compare a range of such models as baselines for the Commonsense Storage Prediction task. These models are used either to detect and segment storage containers or to predict likely storage locations of household items. Our dual strategy compares structured, text-based reasoning (NOAM) against end-to-end multimodal inference, enabling a comprehensive analysis of commonsense storage prediction capabilities.

3 THE STORED HOUSEHOLD ITEM CHALLENGE

We introduce the **Stored Household Item Challenge**, a benchmark for evaluating *Commonsense Storage Prediction* in domestic environments. This task requires reasoning about *non-visible* items. A system is to infer plausible hidden locations using visual context, semantic priors, and typical storage conventions. For example, given the query “mug”, the model should prefer an upper cabinet over a lower drawer, even if the mug is not visible.

Input: A single RGB image of a scene and the name of a common household item (e.g., “mug”, “cutting board”).

Output: The most likely storage container *instance* (e.g., a specific drawer or cabinet) in the image where the item is stored. The prediction must be grounded in a 2D polygon corresponding to a visible container.

This task involves three commonsense reasoning challenges:

- **Reasoning Beyond Visibility:** Most vision systems detect only visible objects. This task requires inferring hidden item locations using indirect cues and commonsense knowledge.
- **Semantic Understanding and Commonsense Priors:** Success depends on functional knowledge of household norms combined with visual context.
- **Grounded, Actionable Output:** Predictions must be specific and physically actionable. Rather than generic guesses like “a cabinet,” the model must identify a concrete container instance suitable for robotic interaction.

To date, no existing dataset targets reasoning about hidden storage in realistic household scenes, and VLMs often default to literal descriptions of what is visible. By offering a structured task, curated data, and measurable evaluation criteria, this challenge fills a critical gap and provides a testbed for evaluating commonsense visual reasoning in service robotics.

4 DATASETS AND DATA COLLECTION

To predict where household items are typically stored, we required a dataset capturing commonsense storage behavior. Specifically, where items are likely kept out of sight in containers. As no existing dataset supported this task, creating a new one posed two challenges. First, in terms of **privacy**, domestic images often contain personal content that is difficult to anonymize and share ethically. Second, it is hard to **scale**. Gathering a large and diverse set of labeled images from volunteers is logistically challenging, making small-scale collection feasible but broad-scale coverage impractical. To address these challenges, we adopted a dual-dataset strategy:

- A **crowdsourced development dataset** using public kitchen images, supporting design, tuning, and analysis.
- A smaller **real-world evaluation dataset** collected from participants' homes, used for final testing.

The development set was built from the SUN dataset [43, 44]. Annotators recruited via the Upwork freelance platform¹ selected the most likely storage container for a given item in each kitchen image. Figure 4 shows the annotation interface. We cleaned the data, removed duplicates, assessed annotator agreement, and curated 6,500 high-quality item-image pairs with polygon-level annotations.

The evaluation set comprises 100 item-image pairs collected from real kitchens, each annotated with the actual storage location of the queried item, serving as the ground-truth reference for model assessment.

To constrain variability, we focused on kitchens and selected 15 household items spanning common and uncommon examples: *bottle opener*, *Tupperware containers*, *dish towels*, *cutting board*, *bowl*, *spices*, *spoon*, *mug*, *plate*, *pot*, *pan*, *cutting knife*, *cooking oil*, *screwdrivers*, and *painkillers*. The full annotation protocol is released with this paper.

An IRB from *redacted for anonymity* approved data collection involving human participants, covering both datasets.

4.1 Dataset Preprocessing

We applied Grounding-DINO [26] and SAM [22] to segment all visible storage containers. Each development image contained ~ 16 containers on average; each evaluation image, ~ 19 . These segmentations provide visual cues for detection and support automated answer verification, avoiding subjective free-form judgments.

4.2 Development Dataset: Crowdsourced Annotations

From 1,746 kitchen images in SUN, we selected 1,656 with at least three detectable containers. For each of 13 items (excluding two held out for generalization tests), we sampled 500 images, totaling 6,500 item-image pairs.

Three annotators (two from the U.S., one from Ireland) used a custom web tool to label images. Each image showed an item name and marked containers, as shown in Figure 4. Responses were stored in Firebase and local backups. Each annotator contributed 2,860 annotations (1,040 overlapping). Annotations were done in batches and manually reviewed. We collected 8,580 total annotations, from which the 6,500 development pairs were drawn. To assess quality, 16% of examples per item were triple-labeled. Agreement was computed using Fleiss' Kappa, detailed in Section 4.4.

4.3 Evaluation Data: Real-World Objects

We collected 100 item-image examples from 74 participant kitchens. Participants labeled storage locations for *bottle openers*, *Tupperware containers*, *painkillers*, and *screwdrivers*. The last two were excluded from the development set to test generalization. In addition, some participants labeled an extra item of their choice, beyond the four predefined categories. Participants were instructed to exclude sensitive content, and all images were manually reviewed for identifying

¹<https://www.upwork.com/>

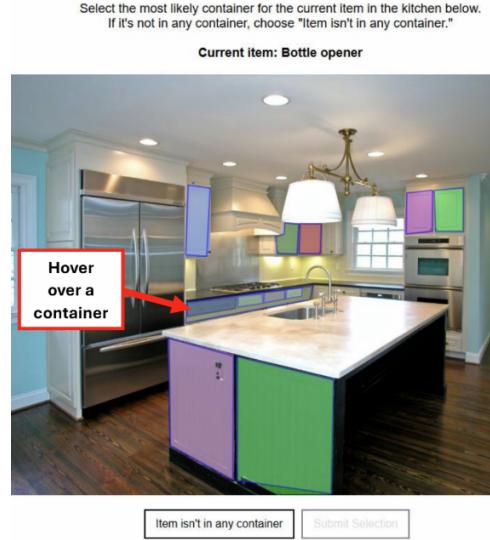


Figure 4: A screenshot from the annotation tool used to collect human-labeled data efficiently.

details. This dataset was used both to validate annotator reliability and as ground truth for model evaluation.

4.4 Data Cleaning and Processing

We applied the following procedures to ensure data quality:

1. Removed duplicates and conflicting responses.
2. Consolidated multiple annotations:
 - Majority vote was used when two annotators agreed.
 - Otherwise, one label was selected at random.
3. Split data into development and evaluation sets.

Inter-annotator agreement varied by item. Fleiss' Kappa was $\kappa = 0.354$ (fair) for the development set and $\kappa = 0.372$ for the evaluation set. On the development set, items with standardized locations (e.g., *bottle openers*, $\kappa = 0.494$; *plates*, $\kappa = 0.478$) showed higher agreement. Subjective categories like *Tupperware* had lower agreement ($\kappa = 0.27155$). These results highlight both the inherent difficulty and variability of the task, while also suggesting that people rely on more than random reasoning when searching for hidden household items.

5 NOAM: NON-VISIBLE OBJECT ALLOCATION MODEL

Our goal is to enable VLMs, and later on service robots, to reason about storage locations with accuracy approaching or exceeding human-level performance as measured on our dataset ($\sim 30\%$). To this end, we reformulate the task from an image-text multimodal problem into a purely language-based one, casting it as a natural language understanding challenge.

Our approach, the Non-visible Object Allocation Model (NOAM), is outlined in Figure 5. For each container detected in the scene, we extract a set of visual and spatial features and translate them into clear, natural language descriptions (steps 2–3). These descriptions

are incorporated into carefully structured prompts that instruct a language model (e.g., ChatGPT) to infer the most probable storage location for a given item (steps 4–5). The model then reasons to select the most probable container(s) (step 7). We extract the predicted container ID, defaulting to the first option when multiple candidates are returned, and map it to the corresponding polygon in the image (step 8). Finally, spatial accuracy is evaluated using Intersection over Union (IoU) with the ground truth (step 9).

Feature Extraction. To represent each container in an image, we extracted features relevant to reasoning, including: container type (e.g., drawer, cabinet door) and confidence score, position relative to the countertop (above/below), neighboring containers and anchor objects (e.g., sink, oven), closest container to each anchor, aspect ratio and shape, and other contextual cues.

Some features were computed directly from container polygons, while others required re-applying Grounding-DINO and SAM to detect countertops and anchor objects. Each container was then represented as a row in a *container information table*, initially containing image path and polygon data. We augmented the table with the extracted features through the following steps:

1. Assign global and local container IDs.
2. Extract and assign labels and confidence scores.
3. Resolve ambiguous labels (e.g., “drawer cabinet door”) using context from neighboring containers.
4. Compute container dimensions and aspect ratio.
5. Determine spatial relation to the countertop.
6. Identify neighboring containers and anchor objects.
7. Measure distances and angles from each container to all anchors to determine the closest container to each anchor.

Natural Language Description. This stage involves two main steps: first, selecting the most informative and relevant features to include in each container’s description, with a focus on attributes that help the language model infer spatial and categorical relationships; and second, converting these features into clear and intuitive natural language. Rather than using raw numerical values (e.g., aspect ratios), we express features qualitatively. For example, using terms like “wider than tall”, “square-like”, and “close to the sink” empirically improved model performance. This strategy aligns with prior findings that language-based tasks are generally more tractable for large language models than compositional or arithmetic reasoning tasks [9, 33].

Prompt Design and Task Framing. To guide the language model in selecting the most likely storage container for a given item, we explored multiple prompt strategies using ChatGPT-4 [31] and LLaMA-3.3 [10]. Each prompt presented a list of container descriptions and required the model to return the most probable container ID, or “None” if none were suitable.

We experimented with three prompt types: (1) Instructional prompts, framing the model as a service robot reasoning about each item; (2) Story-based prompts, encouraging narrative completion grounded in the container descriptions; and (3) Structured system-user prompts, separating global task instructions (system prompt) from instance-specific inputs (user prompt), which yielded the best results. The complete prompt design is provided in Appendix A.

6 EXPERIMENTAL SETUP

We evaluated our method on the real-world dataset (Section 4.3), comparing two model categories: VLMs like Grounding-DINO and MLLMs including Gemini, GPT-4o, Kosmos-2, LLaMA-4, and Qwen-2.5. All models were benchmarked against NOAM.

Model configuration was based on the development set from Section 4.2, with 5% (370 of 6,500 item-image pairs) used for prompt and parameter tuning. Models were prompted with a household item and tasked with identifying its likely storage location (e.g., drawer, cabinet door) by returning a bounding box. Performance was measured using Intersection over Union (IoU) and accuracy at a binary threshold.

Most experiments ran on a standard Intel Core i7 machine (16GB RAM). GPU-dependent models ran on a university cluster with four NVIDIA GTX 1080~Ti GPUs (11GB), 125GB RAM, and a 48-core Intel Xeon CPU.

6.1 Evaluation Metrics

Each model’s performance was evaluated separately on the development and evaluation datasets, and assessed using:

- **Accuracy:** The percentage of predictions with $\text{IoU} \geq 0.5$ compared to ground truth.
- **IoU (Intersection over Union):** The overlap between the predicted and true bounding boxes, computed as:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

We set the IoU threshold to 0.5 based on a quality-weighted analysis of human-labeled predictions, where the average IoU of fully or partially correct responses was 0.54. This threshold balances precision and inclusivity, exceeding the upper bound of incorrect predictions and aligning with human judgment. For Grounding-DINO, we additionally evaluated performance at $\text{IoU} \geq 0.95$, as its container proposals may slightly differ warranting a more permissive evaluation alongside the strict 1.0 threshold. Analysis of the results leading to this value can be found in the Appendix B.

6.2 Baseline Comparisons

Random. For each item-image pair, we took the list of containers provided by Grounding-DINO during the annotation phase and picked one of them at random.

Grounding-DINO and SAM. We used Grounding-DINO in combination with SAM to detect and segment storage containers such as drawers and cabinet doors. For this pipeline, we ran the Grounding-DINO detection model with a textual prompt of the form:

drawer for {item} . cabinet door for {item}

The prompt follows recommendations from the Grounding-DINO authors: it is short, uses simple phrases separated by periods, and avoids complex syntax. The use of “for” (e.g., drawer for {item}) reflects a concise and structured phrasing style that specifies the relation between an item and a potential storage container.

The following thresholds were applied during inference: $\text{box_threshold} = 0.30$ and $\text{text_threshold} = 0.25$, which control the minimum confidence required for detecting bounding boxes and associated text regions, respectively. From the resulting detections, we selected the bounding box with the **highest confidence** and

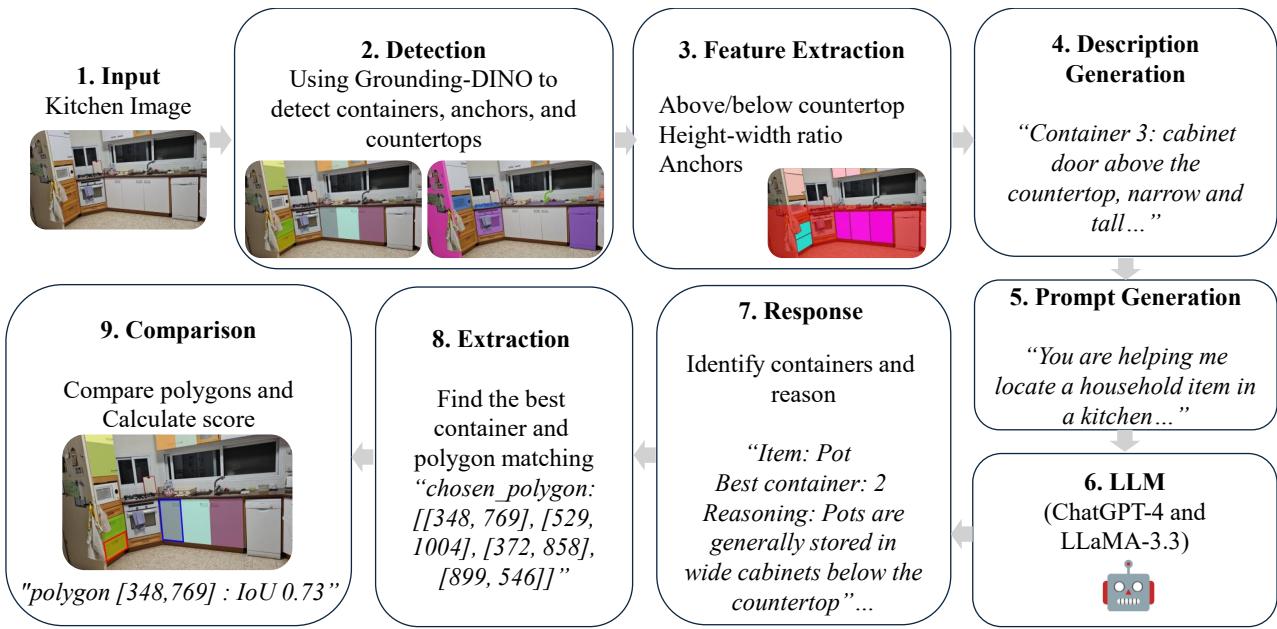


Figure 5: An overview of our NOAM pipeline: from an input image (1) through container detection (2), textual explanation (4), prompt generation (5), through LLM processing (6) to response (7) and value extraction (8) and overall evaluation (9).

passed it to the Segment Anything Model (SAM) to generate segmentation masks.

We used the official implementations of both models: IDEA-Research/grounding-dino and facebook/sam-vit-h. Grounding-DINO was used at commit c023468 of the IDEA-Research/grounding-dino repository. The SAM implementation corresponds to the official April 2023 release from Meta², using the vit-h variant.

Grounding-DINO handled object-level bounding box detection conditioned on the textual prompt, while SAM produced binary segmentation masks for those regions. The segmentation was implemented using the SAM PyTorch interface, which transforms bounding boxes to input-specific image coordinates and outputs a binary mask for each box.

The segmentation masks were post-processed using OpenCV to extract polygon contours. We applied the Ramer-Douglas-Peucker (RDP) algorithm with $\text{epsilon} = 0.02 * \text{arcLength}$ to simplify the polygons.

It is important to note that this use of Grounding-DINO and SAM is distinct from the initial use of Grounding-DINO during dataset annotation, where all visible containers (drawers and cabinets) were detected in bulk. In the current evaluation phase, we specifically query for each item separately, prompting the model to localize where the item is likely stored. To ensure a fair evaluation without introducing annotation bias, a prediction was only considered correct if it achieved an $\text{IoU} \geq 0.95$ to allow only small correction with the ground-truth container.

Grounding-DINO and SAM without Item Prompt. To assess the influence of including the item name, we also ran Grounding-DINO

with a minimal prompt - identical to the annotation phase - omitting the item entirely:

drawer . cabinet door

Again, we selected the container with the highest confidence score. Comparing these results to those obtained with the item-inclusive prompt allows us to quantify how much the item name boosts the model's performance.

Kosmos-2. We used the Kosmos-2 MLLM with both image and text input. The prompt followed a structured grounding format using the special tokens provided by the model:

```
<grounding> In which<phrase> drawer</phrase>
or<phrase> cabinet door</phrase> is<phrase>
a {item}</phrase> stored?
```

The prompt structure follows Kosmos-2's grounding format, where the `<grounding>` tag activates grounding mode and each `<phrase>` tag marks a semantically important span. By marking both the container types (e.g., `drawer`, `cabinet door`) and the queried item, the model is guided to focus on relevant visual concepts and return grounded predictions accordingly.

The model's outputs captions and entity bounding boxes relevant to the query. Since Kosmos-2 does not use a pre-defined list of container locations (unlike human annotators), we considered a prediction correct if the IoU between the model's output and the ground truth was ≥ 0.5 (meaning that the bounding box overlapped the true container in at least 20% of its area).

²<https://github.com/facebookresearch/segment-anything>

We used the official HuggingFace implementation³. For inference, we used the model’s `generate()` method with the following settings:

- `use_cache=True`: enables reuse of key and value tensors from previous self-attention layers for efficient autoregressive decoding.
- `max_new_tokens=128`: limits the number of newly generated tokens to 128.
- `skip_special_tokens=True`: ensures that decoding omits any special tokens in the output.

After generation, we applied the `post_process_generation()` method twice: first with `cleanup_and_extract=False` to inspect raw model output, and again with default settings to extract grounded entity descriptions and their associated bounding boxes.

All inference was conducted on a single NVIDIA GPU, using PyTorch. We used the pre-trained checkpoint without any additional fine-tuning or modification.

Gemini (Google MLLM). We used Google’s Gemini API to evaluate multimodal performance. Specifically, we tested two model versions: `gemini-1.5-flash-latest` and `gemini-2.5-flash`. Predictions were considered correct if the IoU with ground truth was ≥ 0.5 , as in the Kosmos-2 evaluation.

Each image-item pair was processed using the following template:

```
Analyze the provided image of a kitchen.  
Identify the item: '{item}'  
Determine the most likely storage location  
for this item, considering only drawers or  
cabinet doors visible in the image.  
Provide the bounding box coordinates for  
this storage loc-  
ation as a Python list of four integers:  
[x_min, y_min, x_max, y_max].  
The coordinates should be relative to the  
image dimensions (top-left is [0,0]).  
Only output the list of coordinates and  
nothing else. For example: [100, 200, 300,  
400]  
If you cannot determine a likely storage  
location (drawer or cabinet door) for this  
item, output: [0, 0, 0, 0]
```

The API was accessed via an official Google API key, with reliability safeguards including up to 3 retry attempts (starting with a 5-second delay), and a 1-second pause between successful calls. We saved progress every 50 processed examples.

ChatGPT-4o. We used the GPT-4o model with vision capabilities through the OpenAI API. Predictions were considered correct if the IoU with the ground-truth polygon was ≥ 0.5 , consistent with other models.

The prompt format was:

```
You are analyzing a kitchen image.  
The visible containers are drawers and  
cabinet doors.
```

³<https://huggingface.co/microsoft/kosmos-2-patch14-224>

Table 1: Evaluation Set Accuracy and Average IoU for Various Models.

Model	Accuracy (%)	Average IoU
Human Annotator 1 (IoU = 1)	38.00	0.380
Human Annotator 2 (IoU = 1)	27.00	0.271
Human Annotator 3 (IoU = 1)	36.00	0.361
Random (IoU = 1)	6.00	0.062
Grounding-DINO (IoU = 1)	13.00	0.188
Grounding-DINO (IoU ≥ 0.95)	17	0.188
Grounding-DINO - no item in prompt (IoU = 1)	10	0.117
Kosmos-2 (IoU ≥ 0.5)	4.00	0.042
Gemini-1.5-flash (IoU ≥ 0.5)	3.00	0.034
Gemini-2.5-flash (IoU ≥ 0.5)	1.00	0.027
GPT-4o API (IoU ≥ 0.5)	8.00	0.082
LLaMA-4 (IoU ≥ 0.5)	1.00	0.094
Qwen-2.5 (IoU ≥ 0.5)	5.00	0.091
NOAM LLaMA-3.3 (IoU = 1)	23.00	0.232
NOAM GPT-4 (IoU = 1)	23.00	0.23

The item to store is: {item}

Determine the most likely storage location
among visible drawers and cabinet doors
only.

Return a list of 4-point bounding box coordinates
for the item.

If a suitable location cannot be determined,
return an empty list [].

Only return the bounding box list, nothing
else.

The API was accessed via an OpenAI API key. For each image-item pair, the image was encoded in base64 and sent alongside the prompt in a multimodal request. Responses were parsed to extract the bounding box list. Robustness measures included handling exceptions per request and adding a 1-second delay between API calls to avoid rate limits.

LLaMA-4 and Qwen-2.5 (Together AI API). We evaluated the multimodal LLaMA-4 and Qwen-2.5 models via the Together AI API using the same prompt and base64 image format as GPT-4o. Specifically, we employed the checkpoints `meta-llama/Llama-4-Maverick-17B-128E-Instruct-FP8` and `Qwen/Qwen2.5-VL-72B-Instruct`, respectively. As with the other models, predictions were considered correct if the IoU with the ground-truth container polygon was ≥ 0.5 .

Each request included the Base64-encoded image and the text prompt in a single multimodal payload. We accessed the API using a Together AI API key, parsed the JSON response to extract the bounding box list, and converted it to our evaluation format.

7 RESULTS AND ANALYSIS

Real-World Evaluation Dataset. We begin by reporting the overall performance of the various algorithms on the real-world evaluation dataset. As shown in Table 1, NOAM achieves the highest accuracy and IoU among all of the evaluated models, outperforming random, vision-based, and multimodal baselines.

For follow-up statistical tests, we conducted a one-way ANOVA on accuracy scores, followed by post-hoc pairwise comparisons

Table 2: Development Set Accuracy and Average IoU for Various Models.

Model	Accuracy (%)	Average IoU
Human Annotator (Estimated)	35.83	0.361
Random (IoU = 1)	7.55	0.083
Grounding-DINO (IoU = 1)	5.89	0.112
Grounding-DINO (IoU ≥ 0.95)	9.49	0.112
Grounding-DINO - no item in prompt (IoU = 1)	11.35	0.128
Kosmos-2 (IoU ≥ 0.5)	0.46	0.013
Gemini-1.5-flash (IoU ≥ 0.5)	8.11	0.083
Gemini-2.5-flash (IoU ≥ 0.5)	2.43	0.03
GPT-4o API (IoU ≥ 0.5)	13.78	0.138
LLaMA-4 (IoU ≥ 0.5)	3.24	0.071
Qwen-2.5 (IoU ≥ 0.5)	7.57	0.099
NOAM LLaMA-3.3 (IoU = 1)	23.51	0.244
NOAM GPT-4 (IoU = 1)	28.11	0.287

using the Bonferroni correction. The results confirmed a significant effect of model type on performance ($p < 0.05$). Specifically, NOAM GPT-4 significantly outperformed Gemini-1.5-flash, Gemini-2.5-flash, and Kosmos-2 after Bonferroni correction. Differences with other models, including GPT-4o, Grounding-DINO, LLaMA-3.3, and LLaMA-4, were not statistically significant. NOAM also did not differ significantly from human annotators.

While NOAM did not reach human-level performance, its accuracy (23%) is notably closer to that of the least accurate human annotator (27%) than the gap between that annotator and the next-best human (36%). Further, the pairwise comparison between NOAM and the least accurate human annotator did not reach statistical significance ($p = 0.006, 0.46, 0.026$ for annotators 1, 2, and 3 respectively, before the Bonferroni correction and $p = 0.67, 1.0, 1.0$ after the correction). This result is encouraging and suggests that the pipeline of NOAM has strong potential to soon reach human-level performance, especially as new VLMs and MLLMs become available for integration.

Development Dataset. Table 2 presents results on the annotated development set. While our focus is on evaluation performance, development scores help reveal whether models learned meaningful patterns. For instance, Kosmos-2 underperforms even on the development set (below random) suggesting limited capacity or misalignment with the task rather than overfitting. In contrast, NOAM performs strongly, reflecting effective task-specific learning. Human scores are based on overlapping annotations used for agreement analysis and do not reflect full dataset performance.

8 DISCUSSION AND FUTURE WORK

NOAM demonstrates strong reasoning abilities, but several aspects warrant further development:

Efficiency and Scalability NOAM’s average inference time is approximately 13 seconds per image. The main bottlenecks are Grounding-DINO, which accounts for about 10 seconds, and the LLM API call (e.g., ChatGPT), which takes around 2.9 seconds. Using the LLaMA-3.3 (via Together-AI API) slightly increases total inference time to roughly 15.9 seconds per image.

While this runtime is acceptable for research and offline applications, real-world deployment would require further optimization. Initial experiments replacing Grounding-DINO with a lightweight

detector reduced detection time to under 1 second. Future directions for improving efficiency include model distillation, prompt caching, and the use of smaller, fine-tuned modules to support near real-time inference.

Generalization and Dataset Diversity Performance is expected to vary by image domain. Expanding benchmarks to other room types (e.g., bedrooms, industrial scenes) and adapting anchors and prompts will test robustness.

Embodied and Interactive Reasoning NOAM currently uses static images. Extending it to multi-view or interactive setups could reduce occlusion errors and support belief updates during exploration.

Model Robustness LLM outputs vary by prompt and model. We plan to fine-tune VLM or MLLM on our dataset to improve consistency and compare against zero-shot performance.

Richer Priors and Personalization Integrating spatial priors (e.g., furniture hierarchies) and user-specific data (e.g., gestures or preferences) may enhance practical accuracy and personalization.

Practical Metrics IoU alone is insufficient for real-world robotics. Future work will incorporate task-based metrics (e.g., success rate, energy use), and propose container selection heuristics to improve robustness to detection errors.

9 CONCLUSION

This paper introduces the Stored Household Item Challenge, a novel problem designed to evaluate semantic spatial reasoning about the likely locations of non-visible objects in household environments. Unlike conventional object detection tasks, this challenge targets a critical but underexplored capability: inferring hidden item locations based on context and commonsense knowledge. We present two new datasets of labeled item-image pairs featuring concealed storage scenarios, along with a suite of baseline evaluations. To tackle this task, we further propose NOAM (Non-visible Object Allocation Model), a vision-to-language pipeline that reformulates storage prediction as a structured natural language inference problem. Experimental results demonstrate that NOAM significantly outperforms vision-only baselines, though still not comparable to human-level reasoning capabilities.

This work opens several directions for future research. For example, NOAM currently relies on static descriptions and a single scene view. Future models could benefit from richer inputs, such as sequential observations, user preferences, or multimodal feedback. Expanding the spatial reasoning framework to include hierarchical priors or soft constraints (e.g., “items are often above counters”) could further improve performance. Additionally, integrating this task into interactive, embodied agents may allow systems to refine predictions through exploration and reinforcements, enhancing robustness in real-world settings.

Although this paper focuses on kitchens, the underlying reasoning principles generalize to a wide range of domestic and industrial environments. We plan to expand the dataset and task to include additional spaces and a broader set of item categories. By advancing models of hidden object reasoning, this work contributes to building service robots that can function effectively in unfamiliar, dynamic, and complex human environments.

REFERENCES

- [1] S. Aarthi and S. Chitrakala. 2017. Scene understanding – A survey. In *2017 International Conference on Computer, Communication and Signal Processing (ICCCSP)*. 1–4.
- [2] Nichola Abdo, Cyril Stachniss, Luciano Spinello, and Wolfram Burgard. 2015. Robot, organize my shelves! Tidying up objects by predicting user preferences. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, Seattle, WA, USA, 1557–1564.
- [3] Mazda Ahmadi and Peter Stone. 2006. A multi-robot system for continuous area sweeping tasks. In *International Conference on Robotics and Automation, ICRA*. IEEE, 1724–1729.
- [4] Henrik Christensen Akanimoh Oluwasanmi Adeleye. 2023. *Enabling Assistive Service Robots to Contextually Organize Household Objects*. Ph.D. Dissertation. UC San Diego.
- [5] Christian Ayala, Carlos Aranda, and Mikel Galar. 2024. Guidelines to Compare Semantic Segmentation Maps at Different Resolutions. *IEEE Transactions on Geoscience and Remote Sensing* 62 (2024), 1–16.
- [6] C. Bousquet-Jette, Sofiane Achiche, Dominique Baini, Yann-Seing Law-Kam Cio, Cédric Leblond Ménard, and Maxime Raison. 2017. Fast scene analysis using vision and artificial intelligence for object prehension by an assistive robot. *Engineering Applications of Artificial Intelligence* 63 (Aug. 2017), 33–44. <https://doi.org/10.1016/j.engappai.2017.04.015>
- [7] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. 2016. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics* 32, 6 (Dec. 2016), 1309–1332.
- [8] Weifeng Chen, Guangtao Shang, Aihong Ji, Chengjun Zhou, Xiyang Wang, Chonghui Xu, Zhenxiong Li, and Kai Hu. 2022. An Overview on Visual SLAM: From Tradition to Semantic. *Remote Sensing* 14, 13 (June 2022), 3010.
- [9] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. 2023. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems* 36 (2023), 70293–70332.
- [10] Aaron Grattafiori et al. 2024. The Llama 3 Herd of Models. *arXiv: 2407.21783* (2024). arXiv:2407.21783 [cs.AI] <https://arxiv.org/abs/2407.21783>
- [11] Deng-Ping Fan, Ge-Peng Ji, Ming-Ming Cheng, and Ling Shao. 2022. Concealed Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 10 (Oct. 2022), 6024–6042.
- [12] Sourav Garg, Niko Sünderhauf, Feras Dayoub, Douglas Morrison, Akansel Cosgun, Gustavo Carneiro, Qi Wu, Tat-Jun Chin, Ian Reid, Stephen Gould, Peter Corke, and Michael Milford. 2020. Semantics for Robotic Mapping, Perception and Interaction: A Survey. *Foundations and Trends® in Robotics* 8, 1–2 (2020), 1–224.
- [13] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. 2020. A survey on instance segmentation: state of the art. *International Journal of Multimedia Information Retrieval* 9, 3 (Sept. 2020), 171–189.
- [14] Shoichi Hasegawa, Akira Taniguchi, Yoshinobu Hagiwara, Lotfi El Hafi, and Tadahiro Taniguchi. 2023. Integrating probabilistic logic and multimodal spatial concepts for efficient robotic object search in home environments. *SICE Journal of Control, Measurement, and System Integration* 16, 1 (2023), 400–422.
- [15] Yuhang Hu, Zhizhuo Zhang, Ruibo Liu, P Wyder, and Hod Lipson. 2023. Knolling bot: A transformer-based approach to organizing a messy table. *arXiv: 2310.04566* (2023).
- [16] Kanta Kaneda, Shunya Nagashima, Ryosuke Korekata, Motonari Kambara, and Komei Sugiura. 2024. Learning-To-Rank Approach for Identifying Everyday Objects Using a Physical-World Search Engine. *IEEE Robotics and Automation Letters* 9, 3 (2024), 2088–2095. <https://doi.org/10.1109/LRA.2024.3352363>
- [17] Yash Kant, Arun Ramachandran, Sriram Venamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. 2022. Housekeep: Tidying Virtual Households Using Commonsense Reasoning. In *Computer Vision – ECCV 2022*. Springer Nature Switzerland, Cham, 355–373.
- [18] Ivan Kapelyukh and Edward Johns. 2022. My House, My Rules: Learning Tidying Preferences with Graph Neural Networks. In *5th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 164)*. PMLR, 740–749.
- [19] Tareq Khan. 2022. Towards a Low-Cost Object Collecting and Organizing Household Robot using Deep Learning. *European Journal of Electrical Engineering and Computer Science* 6, 6 (Nov. 2022), 16–25.
- [20] Jaeseok Kim, Anand Kumar Mishra, Raffaele Limosani, Marco Scafuro, Nino Cauli, Jose Santos-Victor, Barbara Mazzolai, and Filippo Cavallo. 2019. Control strategies for cleaning robots in domestic applications: A comprehensive review. *International Journal of Advanced Robotic Systems* 16, 4 (July 2019).
- [21] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment Anything. *arXiv:2304.02643* (2023).
- [22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment Anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 3992–4003. <https://doi.org/10.1109/ICCV51070.2023.00371>
- [23] Miklas Kristoffersen, Jacob Dueholm, Rikke Gade, and Thomas Moeslund. 2016. Pedestrian Counting with Occlusion Handling Using Stereo Thermal Cameras. *Sensors* 16, 1 (Jan. 2016), 62.
- [24] Andrey Kurenkov. 2023. *Manipulation and Reasoning Methods for Embodied Object Search*. Ph.D. Dissertation. Stanford University.
- [25] Shaopeng Liu, Guohui Tian, Ying Zhang, Mengyang Zhang, and Shuo Liu. 2022. Service planning oriented efficient object search: A knowledge-based framework for home service robot. *Expert Systems with Applications* 187 (Jan. 2022).
- [26] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. 2024. Grounding DINO: Marrying DINO with Grounded Pre-training for Open-Set Object Detection. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XLVII* (Milan, Italy). Springer-Verlag, Berlin, Heidelberg, 38–55. https://doi.org/10.1007/978-3-031-72970-6_3
- [27] Anish Madan, Neehar Peri, Shu Kong, and Deva Ramanan. 2024. Revisiting few-shot object detection with vision-language models. *Advances in Neural Information Processing Systems* 37 (2024), 19547–19560.
- [28] Tatsuya Matsushima, Yuki Noguchi, Jumpei Arima, Toshiki Aoki, Yuki Okita, Yuya Ikeda, Koki Ishimoto, Shohei Taniguchi, Yuki Yamashita, Shoichi Seto, Shixiang Shane Gu, Yusuke Iwasawa, and Yutaka Matsuo. 2022. World robot challenge 2020 – partner robot: a data-driven approach for room tidying with mobile manipulator. *Advanced Robotics* 36, 17–18 (Sept. 2022), 850–869.
- [29] Shervin Minaee, Yuri Y. Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. Image Segmentation Using Deep Learning: A Survey. *Pattern Analysis and Machine Intelligence* (2021), 1–1.
- [30] Muzammal Naseer, Salman Khan, and Fatih Porikli. 2019. Indoor Scene Understanding in 2.5D for Autonomous Agents: A Survey. *IEEE Access* 7 (2019).
- [31] OpenAI. 2023. GPT-4 Technical Report. *ArXiv abs/2303.08774* (2023). <https://arxiv.org/abs/2303.08774>
- [32] Tanvi A. Patel, Vipul K. Dabhi, and Harshadkumar B. Prajapati. 2020. Survey on Scene Classification techniques. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, Coimbatore, India, 452–458.
- [33] Jing Qian, Hong Wang, Zekun Li, Shiyang Li, and Xifeng Yan. 2023. Limitations of Language Models in Arithmetic and Symbolic Induction. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.
- [34] Ram Ramrakhy, Aniruddha Kembhavi, Dhruv Batra, Zsolt Kira, Kuo-Hao Zeng, and Luca Weihs. 2024. Seeing the unseen: Visual common sense for semantic placement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16273–16283.
- [35] Tiago Ribeiro, Fernando Gonçalves, Inês S. Garcia, Gil Lopes, and António F. Ribeiro. 2021. CHARMIE: A Collaborative Healthcare and Home Service and Assistant Robot for Elderly Care. *Applied Sciences* 11, 16 (Aug. 2021), 7248.
- [36] Kaziwa Saleh, Sandor Szenasi, and Zoltan Vamosy. 2021. Occlusion Handling in Generic Object Detection: A Review. In *2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, Herl’any, Slovakia, 000477–000484.
- [37] Rishi Shah, Yuqian Jiang, Justin Hart, and Peter Stone. 2020. Deep R-Learning for Continual Area Sweeping. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5542–5547.
- [38] Tomoya Shiba, Tomohiro Ono, and Hakaru Tamukoh. 2023. Object Search and Empty Space Detection System for Home Service Robot. *Proceedings of International Conference on Artificial Life and Robotics* 28 (Feb. 2023), 400–403.
- [39] Ashutosh Singandhpur and Hung Manh La. 2019. A Review of SLAM Techniques and Security in Autonomous Driving. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, Naples, Italy.
- [40] Zhehang Tong, Dianxi Shi, Bingzheng Yan, and Jing Wei. 2017. A Review of Indoor-Outdoor Scene Classification. In *2nd International Conference on Control, Automation and Artificial Intelligence (CAAI 2017)*. Sanya, China.
- [41] Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. 2023. TidyBot: personalized robot assistance with large language models. *Autonomous Robots* 47, 8 (Nov. 2023), 1087–1102.
- [42] Zhanxin Wu, Bo Ai, and David Hsu. 2023. Integrating Common Sense and Planning with Large Language Models for Room Tidying. In *RSS 2023 Workshop on Learning for Task and Motion Planning*.
- [43] Jianxiong Xiao, Krista A. Ehinger, James Hays, Antonio Torralba, and Aude Oliva. 2016. SUN Database: Exploring a Large Collection of Scene Categories. *Int. J. Comput. Vision* 119, 1 (Aug. 2016), 3–22.
- [44] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. 2010. SUN database: Large-scale scene recognition from abbey to zoo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
- [45] Yiqing Xu and David Hsu. 2023. “Tidy Up the Table”: Grounding Common-sense Objective for Tabletop Object Rearrangement. *arXiv preprint arXiv:2307.11319*

- (2023).
- [46] Zhi Yan, Nathan Crombez, Jocelyn Buisson, Yassine Ruichck, Tomas Krajinik, and Li Sun. 2021. A Quantifiable Stratification Strategy for Tidy-up in Service Robotics. In *Advanced Robotics and Its Social Impacts (ARSO)*. IEEE, Tokoname, Japan, 182–187.
 - [47] Chengxi Ye, Yezhou Yang, Ren Mao, Cornelia Fermüller, and Yiannis Aloimonos. 2017. What can i do around here? deep functional scene understanding for cognitive robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 4604–4611.
 - [48] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. 2019. Object Detection With Deep Learning: A Review. *Neural Networks and Learning Systems* 30, 11 (Nov. 2019), 3212–3232.
 - [49] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2023. Object Detection in 20 Years: A Survey. *Proc. IEEE* 111, 3 (March 2023), 257–276.

A NOAM: PROMPT DESIGN AND TASK FRAMING

After generating descriptions for all containers in an image, we focused on designing an effective prompt to convey the task to the language model. All prompt variants were tested using ChatGPT-4, and the final selected prompt was also applied to LLaMA-3.3.

In every prompt format and for both models, the goal was to select the most likely container ID from the provided list, indicating where the queried item would typically be stored. If more than one container was retrieved (in case that two containers were selected as the most likely), only the first was used in the evaluation. If the model determined that none of the containers were suitable, it returned “None”.

Below are the main prompt strategies we explored:

1. Instructional Prompt: Service Robot Role. Our first approach described the model as a service robot trying to locate an item in a kitchen image, based on a list of container descriptions:

```
1  prompt = f"""You are a service robot in a domestic
   environment. You are now looking at a kitchen
   scene.
2
3  The containers (drawers, cabinet doors etc.) detected
   in the image are:
4  {chr(10).join(f"- {desc}" for desc in containers)}
5
6  For each of the following items:
7  {', '.join(items)}
8
9  Please determine in which container each item is most
   likely to be stored. If no container is suitable,
   say so. Provide reasoning for each item.
10
11 Format:
12 Item: [Name]
13 Best container: [just container id or "None"]
14 Reasoning: [Your explanation]
15 """
```

2. Story-Based Prompt. We next tried a narrative approach, asking the model to complete a short story that ends with a storage decision:

```
1  prompt = (
2      f"As they stepped into the kitchen, they began
       searching for a {item.lower()}, scanning the
       scene for storage areas - known as containers,
       like drawers, cabinets, or pantry doors where
       household items are typically kept.\n"
3      "Based on the descriptions of the detected
       containers, they paused in front of one that
       seemed just right and reached toward...\n"
4      "\n"
5      "Finish the story by selecting the most suitable
       container from the list below, or say the item
       isn't in any container if none of them are
       appropriate:\n"
6      f"{chr(10).join(f"- {desc}" for desc in containers)
        }"
7  )
```

Although this version was more natural and creative, it generally underperformed compared to the direct instructional format.

3. Structured Prompt with System and User Roles. After learning more about prompt engineering, we adopted a structured format to

balance informativeness and brevity. This prompt format allowed the model to internalize the task logic from the system message and focus on inference during deployment. It consists of two parts:

- **System Prompt:** Encodes general knowledge about the task, including comprehensive instructions and illustrative examples to prime the model.
- **User Prompt:** Provides a concise task instance, including the item to be located and the textual descriptions of all detected containers in the scene as presented above.

```
1  system_prompt = """
2  You are helping locate a household item in a kitchen.
3  The item is stored in one of several visible
   containers (e.g., drawers, cabinets), but I don't
   know which.
4  I'll provide a list of container descriptions and the
   item name.
5  Your task is to identify the most likely container
   based on typical kitchen organization. If none
   are suitable, return "None".
6
7  Response format:
8  Item: [Name]
9  Best container: [Container ID or "None"]
10 Reasoning: [Short explanation]
11
12 ### Example 1
13 Item: Fork
14 Containers:
15 - Container 1: cabinet door below the countertop,
   located to the right of the dishwasher.
16 - Container 2: below the countertop, located to the
   left of the dishwasher.
17 - Container 3: cabinet door above the countertop,
   located above the coffee machine.
18 - Container 4: drawer below the countertop.
19 - Container 5: cabinet door above the countertop.
20 - Container 6: cabinet door above the countertop.
21
22 Item: Fork
23 Best container: 4
24 Reasoning: Forks are usually stored in drawers below
   the countertop for easy access.
25
26 ### Example 2
27 Item: Trash Bag
28 Containers:
29 - Container 1: cabinet door above the countertop.
30 - Container 2: drawer below the countertop.
31 - Container 3: cabinet door below the countertop,
   located below the sink.
32
33 Item: Trash Bag
34 Best container: 3
35 Reasoning: Trash bags are commonly stored under the
   sink near the trash can.
36
37 ### Example 3
38 Item: Knife
39 Containers:
40 - Container 1: cabinet door.
41 - Container 2: drawer, located below the electronic
   kettle, above the oven, at the bottom-right of
   the refrigerator, and at the bottom-left of the
   dish drying rack.
42 - Container 3: cabinet door, located below the
   dishwasher.
43
```

```

44 Item: Knife
45 Best container: 2
46 Reasoning: Knives are typically stored in drawers for
   safety and accessibility.
47
48 ### Example 4
49 Item: Baking pan
50 Containers:
51 - Container 1: drawer below the countertop, located at
      the bottom-right of the oven, and at the top-
      right of the electronic kettle.
52 - Container 2: cabinet door below the countertop,
      located to the right of the electronic kettle,
      and at the bottom-right of the oven.
53 - Container 3: cabinet door below the countertop,
      located below the stove.
54 - Container 4: below the countertop.
55 - Container 5: cabinet door below the countertop,
      located to the left of the electronic kettle, and
      at the bottom-left of the oven.
56
57 Item: Baking pan
58 Best container: 5
59 Reasoning: Baking pans are stored in cabinets below
   the countertop near the oven.
60
61 Now respond to the following:
62 """ .strip()
63
64 user_prompt = f"Item: {item}\nContainers:\n" + "\n".join(f"- {desc}" for desc in containers)

```

B EXPERIMENTAL SETUP: DETERMINING IOU THRESHOLD

To select an appropriate IoU threshold for our results, we first divided the models and baselines into two groups:

- **Models with prior container knowledge:** This includes human annotators (who chose from a predefined list of containers in the web application), the random baseline (which selects a random container from Grounding-DINO detections), Grounding-DINO with the item in the prompt (used both for object detection and for item storage), and NOAM (which also receives the list of detected containers).
- **Models without prior container knowledge:** This includes Gemini, GPT-4o, Kosmos-2, LLaMA-4, and Qwen-2.5.

For fair comparison, we required IoU=1.0 for the first group. For the second group, we examined cases with $\text{IoU} \geq 0.0$ and $\text{IoU} \geq 0.1$, performing a sanity check by reviewing all images, marking the “correct” container polygons, and assigning scores of 0 (incorrect), 0.5 (partially correct), or 1 (fully correct). Computing a quality-weighted average IoU from these scores yielded 0.54, which guided our choice of 0.5 as the threshold.

As shown in Figure 6, incorrect predictions (score = 0) never exceed $\text{IoU}=0.17$, while fully correct ones (score = 1) never fall below $\text{IoU} \approx 0.365$. Partially correct cases lie between $\text{IoU} \approx 0.088$ and 0.392. This gap suggests an ideal cutoff in [0.17, 0.365]. We adopt $\text{IoU} \geq 0.5$, which is stricter than the minimum correct bound, to maximize precision based on our quality-weighted analysis.

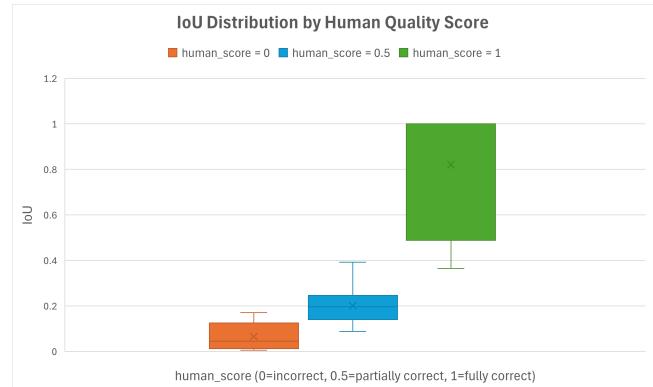


Figure 6: IoU distributions across human-assigned quality scores (0=incorrect, 0.5=partially correct, 1=fully correct).