MG-VQA: MANIPULATION GROUNDED VISUAL QUESTION ANSWERING WITH VLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language models (VLMs) have shown promising spatial reasoning capabilities from static visual inputs, where the evidence needed to answer a question is available in the provided views. However, in cluttered environments, answer-relevant evidence may be occluded rather than absent: an object may lie beneath a pile, be covered by another object, or have identifying information facing away from the camera. Answering such questions requires physical interaction to reveal the hidden evidence. We formalize this setting as *Manipulation-Grounded Visual Question Answering* (MG-VQA), where an agent answers a question about an initially cluttered scene by using manipulation as an intermediate evidence-gathering operation. We introduce MG-VQA-Bench, comprising 600 human-verified questions across four spatial reasoning tasks, and evaluate it in a cluttered tabletop interactive environment. Eight VLMs are evaluated under three levels of environment access: *Direct* (image only), *Perception* (pointing, segmentation, and scene graphs), and *Manipulation* (perception, grasping, and pushing). Across all models, *Direct* and *Perception* achieve average success rates of 36.2% and 37.6%, respectively, only slightly above an image-blind chance baseline of 32.7%. In contrast, *Manipulation* raises average success to 56.8% (40.8–83.0% across models). Stronger tool-calling VLMs, GPT 6 Astra (83.0%) and Gemini 3.8 Flash (69.3%), search persistently and re-ground after unsuccessful interactions, while weaker models often answer without gathering sufficient evidence or stop prematurely after failed actions. Our results highlight the need for VLM agents that use manipulation for persistent, physically grounded evidence gathering and recovery.

1 INTRODUCTION

A household robot must answer questions about cluttered workspaces, such as "Is my charging cable on the table?", "How many batteries are left in the pile?", "What is underneath the blue bowl?", or "Which labeled container is the largest?" A static view may not reveal the evidence needed to answer: objects can be occluded, stacked, or oriented away from the camera. Answering reliably requires lifting, sliding, or repositioning objects to reveal hidden evidence. Manipulation becomes a necessary information-gathering step whenever the relevant evidence is hidden.

Vision Language Models (VLMs) demonstrate broad spatial reasoning capabilities. They identify and refer to objects (You et al., 2024), describe spatial relations (Chen et al., 2024), estimate poses (Bhat et al., 2026), construct scene descriptions (Li et al., 2024), and coordinate external tools (Surís et al., 2023). VLMs are promising interfaces for robots operating in everyday environments. However, most evaluations of VLM spatial reasoning assume that the evidence needed to answer a question is available in static images, videos, or a fixed set of viewpoints. Such settings test whether a model can reason from available visual evidence, but not whether it can recognize when evidence is insufficient and physically uncover what is hidden.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

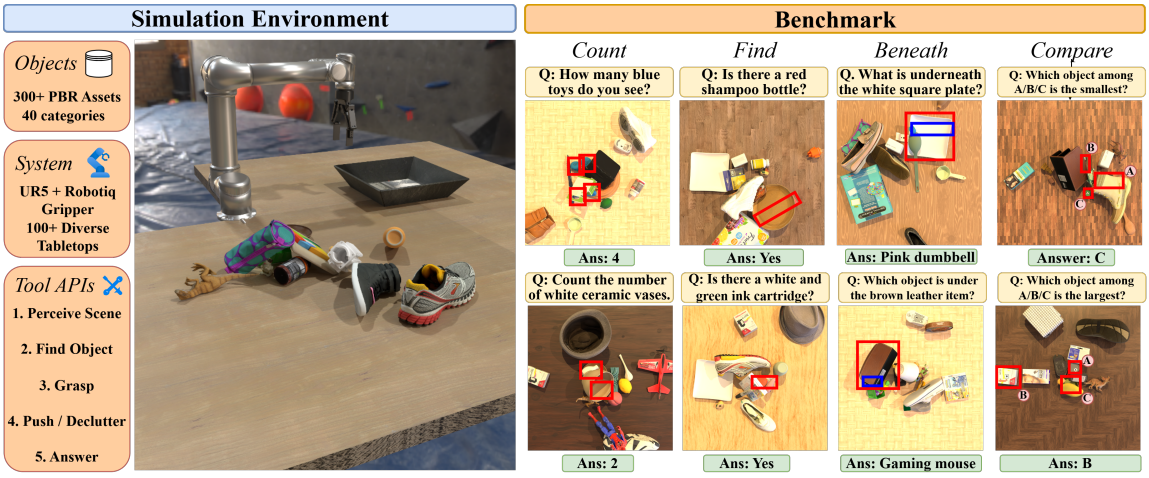

Figure 1: MG-VQA: a simulation environment and benchmark for manipulation-grounded spatial reasoning. Questions in MG-VQA-Bench run on a high-fidelity tabletop simulator (left) with a UR5 + Robotiq platform to physically interact with the scene. MG-VQA-Bench comprises 4 spatial reasoning tasks (right).

This gap is particularly salient in dense tabletop clutter, where manipulation is not an oracle information request. A grasp can fail, an object can slip, and a push can disturb the scene without revealing the expected evidence. An effective agent must inspect the outcome of each manipulation action, determine whether its uncertainty has been resolved, and recover with an updated search strategy before committing to an answer. Current frontier VLMs have not been evaluated on this form of physically grounded evidence gathering.

We study this task as *Manipulation-Grounded Visual Question Answering* (MG-VQA), where evidence needed to answer a question is occluded and must be revealed through manipulation (Figure 1). This distinguishes MG-VQA from conventional manipulation tasks, which aim at a desired final state of the environment, and from static visual question answering, which assumes the evidence is in the image. MG-VQA complements recent active and embodied spatial-reasoning works by focusing on cluttered tabletop question answering, where contact-based actions are physically consequential evidence-gathering operations.

To support this setting, we introduce **MG-VQA-Bench**, a human-verified benchmark of 600 questions over cluttered tabletop scenes covering four spatial reasoning tasks. Each question is constructed such that answer-critical evidence is occluded in the initial view. MG-VQA-Bench is evaluated in an interactive simulation environment **MG-VQA-Sim** with physically realized grasping and pushing actions executed by a UR5 robot. We evaluate frontier VLMs under three levels of access: *Direct*, with only the initial image; *Perception*, with additional scene-understanding tools; and *Manipulation*, with grasping and pushing enabled. VLMs can access tool outputs such as 2D points, scene descriptions and segmentation masks, or invoke grasping and pushing primitives, before returning an answer to the question. This design separates models that reason only from visible evidence from those that use interaction to search for evidence, assess the outcome, and continue when necessary.

We find that *Perception* tools provide limited gains over *Direct* answering, while *Manipulation* improves performance across all tasks. MG-VQA requires persistent, closed-loop evidence gathering and recovery, not merely spatial language understanding or access to tools. Our contributions are:

1. We formulate **Manipulation-Grounded Visual Question Answering (MG-VQA)**, where an agent answers questions about cluttered scenes using manipulation to reveal occluded, answer-relevant evidence. Manipulation is an intermediate information-gathering operation rather than the task objective.

2. We develop and will release **MG-VQA-Sim**, an interactive evaluation environment for tool-calling VLMs that combines cluttered tabletop scenes, scene-understanding APIs, and physically realized grasping and pushing. Interactions carry physical consequences that persist, so agents must re-observe the scene and reason over a workspace their own actions have altered.
3. We introduce **MG-VQA-Bench**, a human-verified benchmark of 600 questions across four spatial reasoning tasks: COUNT, FIND, BENEATH, and COMPARE. Questions are constructed so that answer-critical evidence is substantially occluded in the initial observation.
4. We evaluate 8 VLMs with visual input only, with perception tools, and with manipulation tools. Physical interaction improves spatial reasoning, but models show persistent limitations: deciding when to search, gathering sufficient evidence, re-grounding after acting, and recovery from failed manipulation.

2 RELATED WORK

Robot manipulation benchmarks have used simulators to evaluate language-conditioned lifelong learning, long-horizon tabletop control, and embodied reasoning in GPU-parallelized simulation (Liu et al., 2023; Mees et al., 2022; Tao et al., 2025; Mu et al., 2021; Szot et al., 2021; Nasiriany et al., 2024; Zhang et al., 2025; Mu et al., 2025; Zhu et al., 2026; Yang et al., 2026; Deng et al., 2020). MG-VQA-Sim differs from these works in two ways: it is built for evaluating *tool-calling VLMs* rather than VLA or RL policies, and each scene contains objects spawned in a tight, cluttered area to study recovery under challenging manipulation actions. Following similar motivations, ESI-Bench (Hong et al., 2026) evaluates embodied spatial reasoning through viewpoint changes in indoor environments, abstracting low-level execution to isolate information-seeking decisions. MG-VQA-Bench instead provides a benchmark for manipulation-grounded visual question answering, where VLMs must ground occluders, invoke grasp or push tools with contact-based outcomes unlike prior works that abstract away physics execution, and revise plans when interactions fail. MG-VQA-Bench combines real-scanned assets and 600 human-validated questions over cluttered tabletop scenes. See Appendix A for an extended literature review.

3 MG-VQA: MANIPULATION-GROUNDED VISUAL QUESTION ANSWERING

MG-VQA studies visual question answering in cluttered tabletop scenes where the evidence required to answer the question is not fully visible. An embodied agent cannot always answer a question from its initial observation: relevant objects, relations, or identifying information may be occluded by clutter. In such settings, the agent needs to physically alter the scene to reveal missing evidence and manipulation is an intermediate information-gathering step rather than the task objective.

This distinction makes the setting challenging. The agent does not receive an oracle observation after acting: a grasp may fail, a push may move several objects, and a partially successful interaction may leave relevant evidence occluded. To answer reliably, an agent must interpret the physical outcome of an action, and continue or revise its search when acquired evidence remains insufficient.

3.1 MG-VQA-BENCH: QUESTIONS REQUIRING MANIPULATION-GROUNDED EVIDENCE

We instantiate the preceding motivation with MG-VQA-Bench, comprising four spatial reasoning tasks (150 questions each): COUNT, FIND, BENEATH, and COMPARE. Each task asks about a given scene, but places answer-critical evidence behind occlusion such that reliable answering requires targeted inspection, preventing an agent from answering from a single top-down observation:

- **`Count`: hidden inventory.** The agent is asked to count the number of instances of a target object present in the scene. Visible instances may understate the true count because additional instances can be covered or buried within a pile. The answer is an integer between 0 and 5.

- **`Find`: presence under occlusion.** The agent is asked if a named object is present in the scene. A "yes" answer requires localizing an instance with a 2D point. A target that is not visible may be present beneath clutter; therefore, an agent cannot infer absence if it cannot find the object in the initial image.
- **`Beneath`: revealing an occluded relation.** The agent is asked what lies underneath a referred object. The task requires moving the occluder, observing the exposed region, and identifying the revealed object. The agent chooses from four options, including "None of the Above."
- **`Compare`: inspecting hidden identifiers.** The agent is asked which of three objects marked with A/B/C labels is largest or smallest. Labels may not be initially visible, so the agent must uncover identifying evidence before associating each label with an object and comparing its physical size.

Across all four tasks, manipulation serves a common purpose: it converts initially hidden scene information into observable evidence. However, the appropriate search strategy differs by task. `Count` and `Find` require the agent to inspect plausible occluders before concluding that no additional instance is present. `Beneath` requires targeted inspection of a queried cover object. `Compare` requires associating initially hidden identifiers with objects whose physical attributes must subsequently be compared. Example instances of all four task types are shown in Figure 1 and in Appendix B.

Construction principles. For `Count`, `Find`, and `Beneath`, we retain questions where answer-critical target object(s) have at most 10% pixel visibility in the initial top-down view. For `Compare`, at least one of the A/B/C labels identifying the candidate objects is initially occluded. These criteria ensure that no question is answerable from the initial view alone. We further construct a subset of instances with visually or semantically confounding distractors, illustrated in Figure 2. For example, when the hidden answer object is a red apple, the answer choices may also include another plausible object, such as a red and yellow ball. In these cases, a small visible patch of color or texture is insufficient to identify the hidden object; the agent must inspect the scene to obtain discriminative evidence. Such examples make it possible to test whether an agent genuinely gathers evidence through interaction rather than inferring an answer from a coarse visual cue.

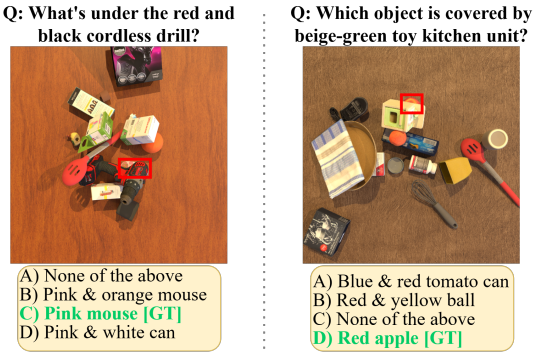

Figure 2: Examples where answer options are constructed to confound static question-answering.

3.2 Benchmark Construction and Human Verification

Scene generation. We generate candidate scenes by sampling objects from an annotated pool of real-scanned everyday assets, placing up to 15 objects in a densely cluttered tabletop workspace, and simulating physics settling. For `Count`, we sample a queried object name and a target multiplicity from 0 to 5; when targets are present, they are placed before distractors such that they are likely to be occluded after settling. For `Find`, we generate equal numbers of present and absent cases, with present targets similarly placed underneath the clutter. For `Beneath`, we select cover–hidden-object pairs for which moving the cover substantially reveals the hidden object. For `Compare`, we select three objects with distinct physical volumes and apply standardized A/B/C labels to their object meshes on the visible side from the top view, with at least one label that is initially occluded. See Appendix C.1 for detailed scene-generation procedures.

After the scene settles, we compute ground-truth simulator state and retain only scenes satisfying the task-specific initial-occlusion criteria. VLMs receive only images and tool outputs, not privileged simulator state. We use Gemini 3 Flash to generate diverse question candidates, which are reviewed by human annotators.

Human verification. All candidate questions are reviewed by two human annotators using a custom verification interface. Annotators verify that: (i) question wording, candidate answers, and ground-truth answer are unambiguous; (ii) answer cannot be reliably determined from the initial observation alone; and (iii) the answer can be recovered after revealing the relevant evidence through feasible clutter removal. When a scene is valid but wording or answer options are ambiguous, annotators manually correct the metadata; otherwise, the instance is excluded. See Appendix C.2 for details of the verification interface and protocol.

3.3 MG-VQA-SIM: AN ENVIRONMENT FOR EVIDENCE GATHERING AND RECOVERY

We develop MG-VQA-Sim as an interactive evaluation environment for VLMs answering questions in MG-VQA-Bench. The environment retains the physical consequences of information-gathering actions: a robot can clear clutter to expose hidden evidence, but actions may also fail, move unintended objects, or leave the answer only partially revealed. This allows us to evaluate whether a model can observe action outcomes and recover when its initial strategy is unsuccessful.

We implement MG-VQA-Sim using a tabletop environment, an attached UR5 robot with physically realized grasping and pushing, and render top-down RGB observations for VLMs. Each scene in MG-VQA-Bench contains up to 15 objects in a tightly bounded workspace and a clearing basket for temporarily relocating clutter. See Appendix C.3 for the simulation and rendering stack, asset sources and annotations, workspace configuration, and scene-initialization procedure.

Tool interface. The agent interacts with the environment through four information-gathering tools and a terminal answer tool. The `perceive_scene` tool returns a scene description, scene graph, visible-object list with 2D points, and pairwise spatial relations; `find_object` returns a segmentation mask for a queried point corresponding to an object from the scene graph. The `grasp` and `push` tools provide complementary contact-based manipulation actions to uncover occluded evidence, while `answer_question` terminates the episode and records the response. See Appendix C.4 for full API specifications.

Physically consequential actions and recovery. Unlike oracle-based reveal operations, tool calls can have imperfect outcomes. A grasp can fail due to low confidence predicted grasps or object slippage, a push can produce partial displacement or collateral object motion, and an initially plausible action can leave the answer-critical evidence occluded. These outcomes are purposeful features of the environment: they require an agent to inspect whether an action had its intended informational effect, rather than assuming that an attempted grasp or push resolved the uncertainty.

For example, after a failed grasp, an agent may re-query the scene, identify a newly accessible region, push nearby clutter aside, and attempt a different grasp. Conversely, it may determine from the updated observation that it has already acquired enough evidence to answer. MG-VQA-Sim therefore supports evaluation of persistent, physically grounded search rather than one-shot action selection. We will release the environment, scene-generation pipeline, tool APIs, benchmark instances, and evaluation code upon publication.

4 RESULTS ON MG-VQA-BENCH

We evaluate VLMs using no tools (*Direct* mode), only perception tools `perceive_scene` and `find_object` (*Perception* mode), and both perception and manipulation tools `grasp` and `push` (*Manipulation* mode). We also include an image-blind *Chance* baseline that answers "0" for `Count`, "No" for `Find`, "None of the above" for `Beneath` and randomly chooses among A/B/C for `Compare`. Table 1 shows results on MG-VQA-Bench, all values reported as success rate in %.

Across all models, the use of manipulation tools improves success rate compared to the *Direct* and *Perception* modes. Models differ in their gains in *Manipulation* mode: GPT 6 Astra gains 40.5%, while GLM 5.3,

Table 1: Success rates (%) on 600 human-verified questions (150 questions per task type). *Avg.* is the macro-average over the four tasks; **bold** marks the best result per column, underline the second best.

Open-source models

Mode	Count	Find	Beneath	Compare	Avg.
Chance	22.0	50.0	26.0	32.7	32.7
Gemma 4 31B IT					
Direct	24.0	38.0	33.3	35.3	32.7
Perception	22.7	46.7	28.7	43.3	35.3
Manipulation	32.0	50.7	52.7	42.0	44.3
Kimi K3					
Direct	24.7	39.3	29.3	52.7	36.5
Perception	25.3	44.7	32.0	60.7	40.7
Manipulation	40.0	53.3	74.7	58.7	56.7
GLM 5.3					
Direct	24.7	38.7	23.3	34.0	30.2
Perception	21.3	42.0	27.3	39.3	32.5
Manipulation	25.3	47.3	46.7	44.0	40.8
Qwen 3.8 Flash Next					
Direct	28.7	40.0	34.0	52.0	38.7
Perception	24.0	38.0	31.3	47.3	35.2
Manipulation	36.7	42.0	66.0	71.3	54.0

Proprietary models

Mode	Count	Find	Beneath	Compare	Avg.
Chance	22.0	50.0	26.0	32.7	32.7
Gemini 3.1 Pro					
Direct	24.7	47.3	42.7	48.0	40.7
Perception	27.3	48.7	36.0	50.0	40.5
Manipulation	62.7	66.7	66.7	62.0	64.5
Gemini 3.8 Flash					
Direct	28.0	49.3	44.0	47.3	42.2
Perception	25.3	48.7	41.3	46.0	40.3
Manipulation	60.0	66.0	80.0	71.3	69.3
Claude Opus 4.8					
Direct	23.3	38.7	28.0	35.3	31.3
Perception	22.7	45.3	27.3	41.3	34.2
Manipulation	25.3	47.3	51.3	42.0	41.5
GPT 6 Astra					
Direct	33.3	31.3	33.3	50.0	37.0
Perception	32.7	44.0	36.0	57.3	42.5
Manipulation	**87.3**	**80.7**	**88.0**	**76.0**	**83.0**

Opus 4.8, and Gemma 4 31B gain less than 10% over their *Perception* mode success rates. Top-performing models manipulate more on average across all tasks (Figure 3, left). *Perception* mode, where VLMs use tool outputs like 2D points, segmentation masks and scene graphs leads to mixed results over *Direct* mode across models, with a difference of -3.5% to +5.5% on average. In *Direct* and *Perception* modes, average success rate across all models is limited to 25.8% on `Count`, 33.0% on `Beneath`, 46.2% on `Compare` and 42.5% on `Find`. With *Manipulation*, the models on average achieve 46.2% (+20.4%) on `Count`, 65.8% (+32.8%) on `Beneath`, 58.4% (+12.2%) on `Compare` and 56.8% (+14.2%) on `Find`. Compared to the image-blind *Chance* baseline, *Direct* and *Perception* mode results lie within 8% on average success rate, underlining the need for physical scene interaction to answer questions in MG-VQA-Bench.

Models fail on `Find` and `Count` largely because they do not gather the evidence needed to answer. In the `Find` task, all models have better success rates in answering questions where the object is not present in the scene (Figure 3, center). In the `Count` task, model failures are dominated by undercounting rather than overcounting (Figure 3, right): VLMs may prematurely answer the question with the evidence of a few visible instances of a referred object, without investigating the rest of the scene for more instances. When VLMs manipulate objects, success rates rise sharply for object-present `Find` questions, from 7.3% to 30.3%, and more modestly for object-absent questions, from 77.8% to 82.8%.

4.1 Prompt ablations

Models vary widely in their choice to invoke manipulation with substantial differences per task (Figure 3, left). We introduce three prompt variations to study the behavior of VLMs under explicit manipulation instructions or auxiliary viewpoints: (i) *Search* adds task-specific guidance at the beginning of the episode, encouraging manipulation of plausible occluders when objects are hidden; (ii) *Search-persistent* repeats this guidance on every subsequent turn as a persistent message; (iii) *Multiview* adds an additional fixed, oblique RGB view along with the original top-down view for every model-turn. Table 2 shows that *Search* guidance improves the average SR for both Kimi K3 (56.7% to 61.8%) and Opus 4.8 (41.5% to 44.0%). *Search-persistent* further improves scores for Kimi K3 (61.8% to 66.0%) but reduces it for Opus 4.8 (44.0%

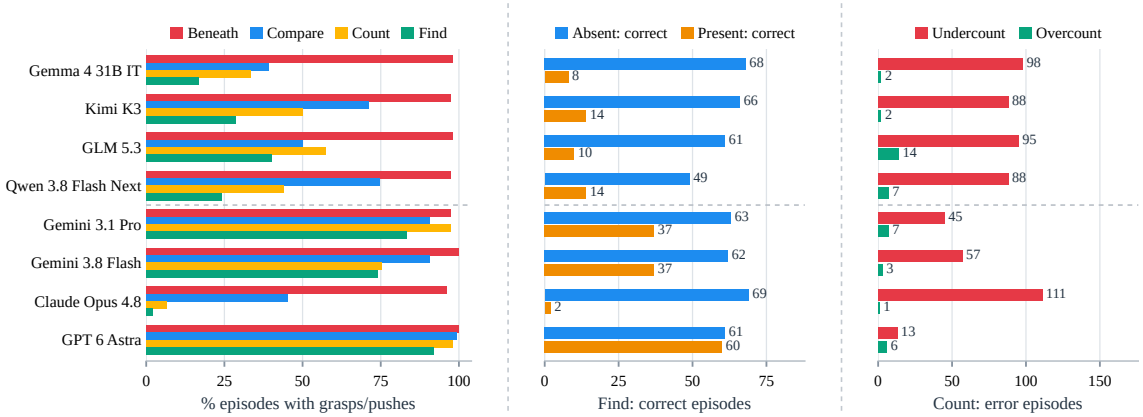

Figure 3: Manipulation and evidence gathering in MG-VQA-Bench. *Left*: percentage of the 150 *Manipulation*-mode episodes per model and task in which at least one `grasp` or `push` was used. *Center*: correctly answered `Find` questions when the target is absent versus present, 75 episodes each. *Right*: `Count` questions answered with an undercount versus an overcount; invalid count answers are excluded. Models are ordered as in Table 1; horizontal rule separates open-source from proprietary models.

to 41.5%). Both these variants encourage improved interactions, as the number of episodes where these models interact with the scene increases (Kimi K3: 371 to 519 to 527, Opus 4.8: 225 to 293 to 305) along with the average manipulation tool calls per question.

Explicit manipulation guidance can therefore encourage weaker models to manipulate more, but it only helps when the model can exploit what its actions reveal. We also ran GPT 6 Astra with *Search* ablation: the average success rate decreased from 83.0 to 80.5 while the number of episodes where manipulation occurred remained the same (584). *Multiview* ablation shows that an additional viewpoint improves performance without increasing manipulation actions. However, the improvement is not uniform across all tasks: SR for `Count` and `Beneath` increases (additional view may reveal hidden objects), but in `Compare` it is unchanged for Kimi and lower for Opus (badge labels A/B/C are only visible from top-down view).

Table 2: Ablations conducted on MG-VQA-Bench (600 questions); values reported in success rate (%). Acted episodes contain at least one `grasp` / `push`. Ablations are separate additions.

Model	Ablation	Beneath	Compare	Count	Find	Avg.	Acted /600	Mean manip. /ep
Kimi K3	Baseline	74.7	58.7	40.0	53.3	56.7	371	1.7
	+ Search	70.0	64.7	48.7	64.0	61.8	519	2.4
	+ Search-persistent	73.3	63.3	62.0	65.3	66.0	527	2.4
	+ Multiview	76.0	58.7	54.0	60.7	62.3	385	1.6
Opus 4.8	Baseline	51.3	42.0	25.3	47.3	41.5	225	1.0
	+ Search	48.7	48.0	30.0	49.3	44.0	293	1.3
	+ Search-persistent	44.7	39.3	31.3	50.7	41.5	305	1.3
	+ Multiview	54.7	37.3	34.0	48.7	43.7	215	0.7

4.2 What separates GPT 6 Astra from other VLMs

GPT 6 Astra was released during the preparation of this paper and exhibits strong performance on all manipulation-grounded spatial reasoning tasks in MG-VQA-Bench. Its improvements can be attributed to: (1) better manipulation-driven search effort for occluded objects and (2) robust error recovery after an

initial grasp failure. Across all tasks, GPT 6 Astra executes more manipulation actions, averaging 3.9 per question, versus 0.9–2.9 for the other models. Within each task, Astra uses more model turns (includes both perception and manipulation tools) than other models (Table 3).

GPT 6 Astra also shows better failure recovery patterns compared to other models (Table 4). After an initial grasp failure, every model chooses to push rather than retrying a grasp. The two lowest performing models, GLM 5.3 and Opus 4.8, directly choose to answer after a failed grasp in 46.3% and 23.6% of questions respectively. Before invoking the *push* tool, GPT 6 Astra prefers to generate a new scene graph followed by the segmentation tool (98.8%). Failed grasps may move other scene objects, rendering the previous turn's scene graph stale. GPT 6 Astra consistently calls for a new scene graph before the next manipulation. We show a qualitative example in Figure 4.

Table 3: Median model turns / manipulation calls per question, computed over 150 questions per task.

Model	Count	Find	Beneath	Compare
Gemma 4 31B IT	2 / 0	2 / 0	7 / 2	4 / 0
Kimi K3	3 / 0.5	2 / 0	7 / 2	7 / 2
GLM 5.3	5 / 1	2 / 0	7 / 2	4 / 0.5
Qwen 3.8 Flash Next	2 / 0	2 / 0	5 / 1	7 / 2
Gemini 3.1 Pro	11 / 3	8 / 2	6 / 2	10 / 3
Gemini 3.8 Flash	8 / 2	7 / 2	6 / 2	9 / 2
Claude Opus 4.8	2 / 0	2 / 0	6 / 2	5 / 0
GPT 6 Astra	16 / 5	14 / 4	8 / 2	12 / 3

Table 4: Agent actions following failed grasps. N denotes the number of episodes with at least one grasp failure. *Next manipulation* groups these episodes by the next manipulation action: `push`, `grasp`, or none if the model answers directly. For episodes with a subsequent `push`, *Re-grounding* shows the percentage in which the model updated the scene graph before segmentation (*Scene + seg.*) or performed segmentation using an outdated scene graph (*Seg. only*). Models are ordered as in Table 1.

		Next manipulation (% of N)			Re-grounding (% of pushes)	
Model	N	Push	Retry grasp	None	Scene + seg.	Seg. only
Gemma 4 31B IT	213	76.1	11.3	12.7	32.1	64.2
Kimi K3	246	85.8	7.7	6.5	43.1	55.5
GLM 5.3	246	50.8	2.8	46.3	57.6	35.2
Qwen 3.8 Flash Next	129	86.0	1.6	12.4	61.3	32.4
Gemini 3.1 Pro	454	79.3	13.2	7.5	48.9	48.1
Gemini 3.8 Flash	422	70.1	17.5	12.3	10.5	84.5
Claude Opus 4.8	182	48.4	28.0	23.6	4.5	88.6
GPT 6 Astra	502	82.7	12.4	5.0	98.8	0.0

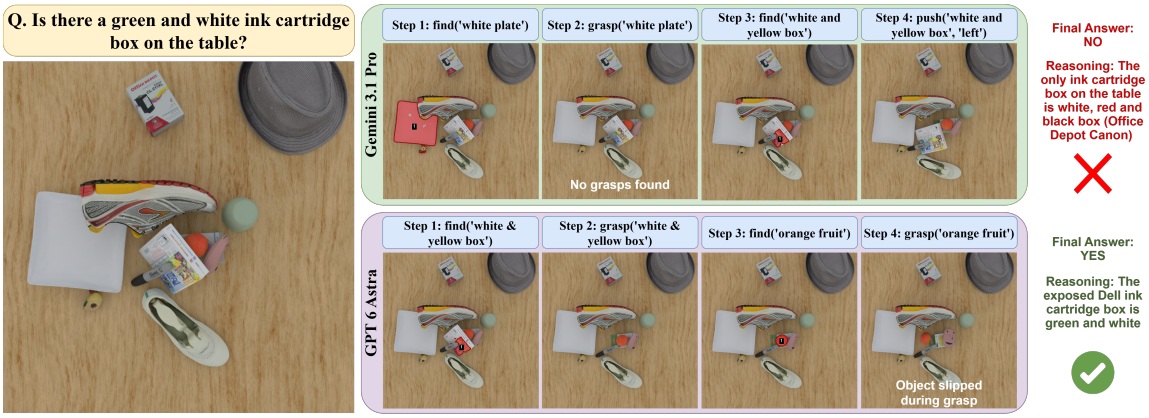

Figure 4: Gemini 3.1 Pro (top) attempts to grasp a flat plate, then displaces a different object with a push and concludes from the still-occluded scene that the target is absent. GPT 6 Astra (bottom) investigates the central pile of objects sequentially, answering only once the target is exposed.

4.3 ASTRA-ASSISTED MG-VQA-HARD BENCHMARK

We utilize GPT 6 Astra as a coding agent to create a new scene generation algorithm for complex questions. The agent first analyzed the tool-calling behaviors across all models and was tasked with generating scenes that are likely to be challenging for existing models. Instead of the sequential clutter-spawning strategy used to create MG-VQA-Bench, the agent was given tools to spawn objects in specific locations, choose table textures and pick from an extended library of 50 objects (*e.g.,* 3D objects that are not yet publicly available). The resulting scenes were manually verified and we will release it separately as the MG-VQA-Hard benchmark. MG-VQA-Hard comprises 40 questions, with 10 per task. We show example scenes in Appendix D. We evaluate 2 representative open-source models and 3 proprietary models on this subset, with results shown in Table 5. All evaluated models underperform compared to MG-VQA-Bench, reporting 17.5% – 27.5% on average success rate. MG-VQA-Hard breaks the correlation between success rate and manipulation actions, as Gemini 3.1 Pro with 3.75 average manipulation actions per question performs at the same level as Qwen 3.8 Flash Next with 0.8 manipulation actions per question.

Key findings. GPT 6 Astra as a coding assistant exploits two vulnerabilities in current models. When the scene shows well-separated objects that appear to lie flat on the table, VLMs assume the isolated object arrangement does not contain any clutter. When smaller items are hidden underneath such objects, Kimi reasons "flat boxes/DVD cases cannot hide cubes underneath in any meaningful way," while Gemini 3.8 Flash describes "packages resting directly on the table with no occlusions." Secondly, all models struggle in estimating the height of the objects in top-down view: six of GPT 6 Astra's eight comparison errors occur when it repeatedly judges a horizontally elongated object (e.g. "butter box") larger than a vertically situated taller object (e.g. "shampoo bottle") highlighting the challenges in 3D understanding in frontier VLMs.

Table 5: Success rates and manipulation statistics on the MG-VQA-Hard benchmark. Results cover 40 questions, with 10 per task type. Success rates are reported as percentages. Column definitions follow Table 2. **Bold** indicates the best result in each column, and underline indicates the second best.

Model	Count	Find	Beneath	Compare	Avg.	Acted /40	Mean manip. /ep.
Open-source models							
Kimi K3	0.0	10.0	10.0	50.0	17.5	18	1.125
Qwen 3.8 Flash Next	0.0	0.0	**50.0**	**60.0**	**27.5**	17	0.800
Proprietary models							
Gemini 3.1 Pro	10.0	**20.0**	30.0	50.0	**27.5**	**38**	**3.750**
Gemini 3.8 Flash	0.0	10.0	40.0	50.0	25.0	24	1.575
GPT 6 Astra	**20.0**	**20.0**	40.0	20.0	25.0	34	3.450

5 CONCLUSION

Cluttered tabletop scenes challenge VLMs even with pointing, segmentation, and scene graphs, as answering spatial questions often requires manipulating objects to reveal hidden evidence. MG-VQA-Bench and MG-VQA-Hard evaluate this ability across frontier VLMs. Proprietary models outperform open-source models when manipulation is available (64.6% vs. 49.0%), with weaker models often reluctant to search through manipulation. The MG-VQA-Hard benchmark exposes additional biases, including inferring the absence of hidden objects from the spacing between visible ones. These findings highlight challenges in gathering and verifying evidence through interaction. We will release the benchmarks and code to support reproducible evaluation and progress in manipulation-grounded visual question answering.

Limitations. Our tool APIs and simulator address table-top environments with a fixed manipulator; extension to other tools, embodiments, and environments is left to future work. VLMs in MG-VQA-Bench operate over a discrete fixed tool library, limiting their ability to recover from failures by creating new tools; future work will explore how VLMs can be given access to design better tools to manipulate objects.

AI use statement. We used generative AI tools as a coding assistant during the development of MG-VQA-Sim, the evaluation pipeline, plotting code for the figures in this paper, and to check the grammar and terminological consistency of the manuscript. We did not use generative AI tools to generate research ideas, to design MG-VQA-Bench or its evaluation protocol, or to conduct the literature review or select citations. Two uses fall within benchmark construction: object name annotations were produced by a VLM and subsequently verified by human annotators, and MG-VQA-Hard was constructed with AI coding assistance. All AI-assisted code was reviewed and verified by the authors through testing and inspection of its outputs, and all figures were checked against the underlying results. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

Vineet Bhat, Sungsu Kim, Valts Blukis, Greg Heinrich, Prashanth Krishnamurthy, Ramesh Karri, Stan Birchfield, Farshad Khorrami, and Jonathan Tremblay. Bop-ask: Object-interaction reasoning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14455–14465, June 2024.

Siyi Chen, Mikaela Angelina Uy, Chan Hee Song, Faisal Ladhak, Adithyavairavan Murali, Qing Qu, Stan Birchfield, Valts Blukis, and Jonathan Tremblay. Spacetools: Tool-augmented spatial reasoning via double interactive rl. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2021.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Yuhong Deng, Di Guo, Xiaofeng Guo, Naifu Zhang, Huaping Liu, and Fuchun Sun. Mqa: Answering the question via robotic manipulation. *arXiv preprint arXiv:2003.04641*, 2020.

Zhao Dong, Ka Chen, Zhaoyang Lv, Hong-Xing Yu, Yunzhi Zhang, Cheng Zhang, Yufeng Zhu, Stephen Tian, Zhengqin Li, Geordie Moffatt, Sean Christofferson, James Fort, Xiaqing Pan, Mingfei Yan, Jiajun Wu, Carl Yuheng Ren, and Richard Newcombe. Digital twin catalog: A large-scale photorealistic 3d

object digital twin dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025.

Max Fu, Justin Yu, Karim El-Refai, Ethan Kou, Haoru Xue, Huang Huang, Wenli Xiao, Guanzhi Wang, Fei-Fei Li, Guanya Shi, et al. Cap-x: A framework for benchmarking and improving coding agents for robot manipulation. *arXiv preprint arXiv:2603.22435*, 2026.

Yining Hong, Jiageng Liu, Han Yin, Manling Li, Leonidas Guibas, Fei-Fei Li, Jiajun Wu, and Yejin Choi. ESI-Bench: Towards embodied spatial intelligence that closes the perception-action loop. *arXiv preprint*, 2026. URL https://esi-bench.github.io/.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. Megapose: 6d pose estimation of novel objects via render & compare. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

Rongjie Li, Songyang Zhang, Dahua Lin, Kai Chen, and Xuming He. From pixels to graphs: Open-vocabulary scene graph generation with vision-language models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28076–28086, 2024. doi: 10.1109/CVPR52733.2024.02652.

Bo Liu et al. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023.

Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, Mrinal Kalakrishnan, Franziska Meier, Chris Paxton, Sasha Sax, and Aravind Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7327–7334, 2022.

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Cathera Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, Lunkai Lin, Zhiqiang Xie, Mingyu Ding, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 27649–27660, June 2025.

Adithyavairavan Murali, Balakumar Sundaralingam, Yu-Wei Chao, Jun Yamada, Wentao Yuan, Mark Carlson, Fabio Ramos, Stan Birchfield, Dieter Fox, and Clemens Eppner. Graspgen: A diffusion-based framework for 6-dof grasping with on-generator training. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2026. URL https://arxiv.org/abs/2507.13097.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024.

Van Nguyen Nguyen, Stephen Tyree, Andrew Guo, Mederic Fourmy, Anas Gouda, Taeyeop Lee, Sungphill Moon, Hyeontae Son, Lukas Ranftl, Jonathan Tremblay, et al. Bop challenge 2024 on model-based and model-free 6d object pose estimation. *arXiv preprint arXiv:2504.02812*, 2025.

Qie Sima, Sinan Tan, Huaping Liu, Fuchun Sun, Weifeng Xu, and Ling Fu. Embodied referring expression for manipulation question answering in interactive environment. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7635–7641, 2023. doi: 10.1109/ICRA48891.2023.10160748.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11523–11530, 2023. doi: 10.1109/ICRA48891.2023.10161317.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2023.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 251–266. Curran Associates, Inc., 2021.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025.

Xuning Yang, Rishit Dagli, Alex Zook, Hugo Hadfield, Ankit Goyal, Stan Birchfield, Fabio Ramos, and Jonathan Tremblay. Robolab: A high-fidelity simulation benchmark for analysis of task generalist policies. In *Robotics: Science and Systems (RSS)*, 2026.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2msbbX3ydD.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction in robotics. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=GVX6jpZOhU.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11142–11152, October 2025.

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and Shanghang Zhang. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 28404–28481. Curran Associates, Inc., 2025.

Yixin Zhu, Zixiong Wang, Jian Yang, Jin Xie, Jingyi Yu, Jiayuan Gu, and Beibei Wang. Toward visually realistic simulation: A benchmark for evaluating robot manipulation in simulation. *arXiv preprint arXiv:2605.06311*, 2026.

A Extended Related Work

Simulation environments for VLM-driven robotics. Robotic manipulation benchmarks have used simulators to evaluate language-conditioned lifelong learning (Liu et al., 2023), long-horizon tabletop control (Mees et al., 2022), and embodied reasoning in GPU-parallelized simulation (Tao et al., 2025; Mu et al., 2021). They span household rearrangement (Szot et al., 2021), generative kitchen environments (Nasiriany et al., 2024), implicit-intent VLA tasks (Zhang et al., 2025), dual-arm digital twins (Mu et al., 2025), material-aware high-fidelity manipulation (Zhu et al., 2026), and broad competency axes across visual, procedural, and relational tasks (Yang et al., 2026). MG-VQA-Sim differs from these works in two ways: it is built for evaluating *tool-calling VLMs* rather than VLA or RL policies, and each scene is designed with objects spawned in a tight, cluttered area to study recovery under challenging manipulation actions.

Embodied question answering. Embodied QA benchmarks progressed from navigation-based visual evidence gathering in synthetic scenes (Das et al., 2018) to real-world episodic-memory and active-exploration settings (Majumdar et al., 2024). MQA (Deng et al., 2020) added object interaction in a V-REP bin environment and AI2-THOR with remote navigation (Sima et al., 2023; Kolve et al., 2017). However, these benchmarks largely emphasize navigation and observation rather than physical manipulation. ESI-Bench (Hong et al., 2026) evaluates embodied spatial reasoning through viewpoint changes and task-specific interactions in indoor environments, abstracting low-level execution to isolate information-seeking decisions. MG-VQA-Bench instead focuses on manipulation-driven evidence acquisition in dense tabletop clutter: VLMs must ground candidate occluders in images and invoke grasp or push tools whose outcomes depend on contact-based execution. This retains grounding errors, failed pickups, slippage, and collateral object motion within the reasoning loop, testing how VLMs revise their plans when an intended interaction does not produce the expected evidence. Prior systems typically relied on disjoint modules for planning, VQA and manipulation, while recent tool-using VLMs enable robot agents that can reason, plan, and act to answer questions beyond static RGB scenes. MG-VQA-Bench provides the first VLM benchmark to measure manipulation-grounded visual question answering in VLMs, combining perception and physically realized manipulation tools, real-scanned assets, and human-validated 600-questions over cluttered tabletop scenes.

Spatial Reasoning and VLM Agents. VLM-based robot perception has progressed from predicting language-conditioned spatial-affordance keypoints (Yuan et al., 2024) to 3D-aware spatial referring and multi-step reasoning (Zhou et al., 2025), yet frontier VLMs still struggle with fine-grained interaction queries such as grasp and waypoint prediction (Bhat et al., 2026). Work on LLM/VLM tool use has evolved from high-level action primitives for robot planning (Singh et al., 2023; Ahn et al., 2022) to code-as-policy agents over perception-control primitives (Fu et al., 2026) and VLM agents that coordinate external perception tools (Chen et al., 2026). These approaches support our central hypothesis that grounded interaction improves VLM-driven manipulation, but existing evaluations remain largely limited to navigation or static RGB-D scenes; MG-VQA-Bench instead studies cluttered tabletop settings where VLMs must decide whether and when to physically interact with the world through manipulation.

B MG-VQA-BENCH EXAMPLES

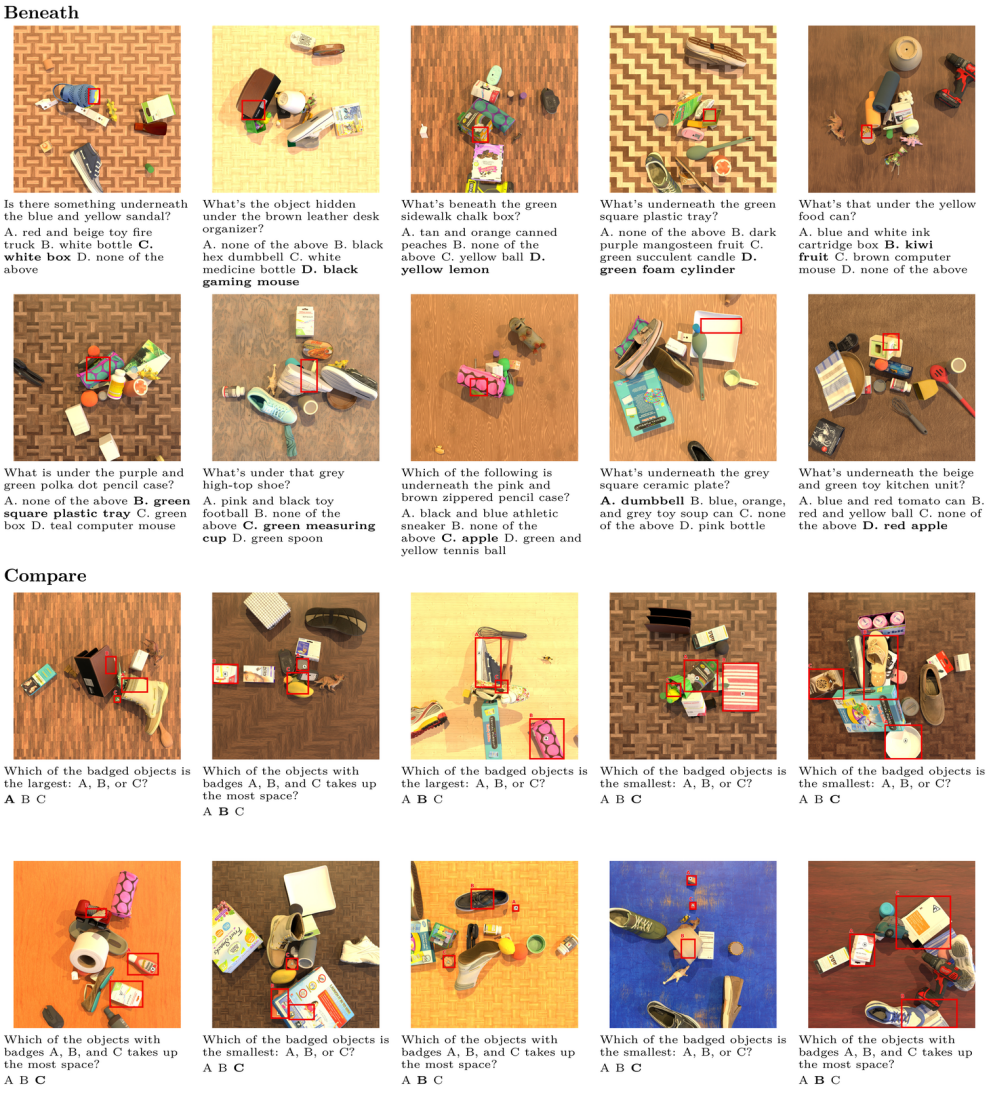

Figure 5: **MG-VQA-Bench: Beneath and Compare.** Ten examples per task. Correct answers are bold. Red boxes show privileged ground-truth target extents, including hidden portions; these overlays are not agent inputs.

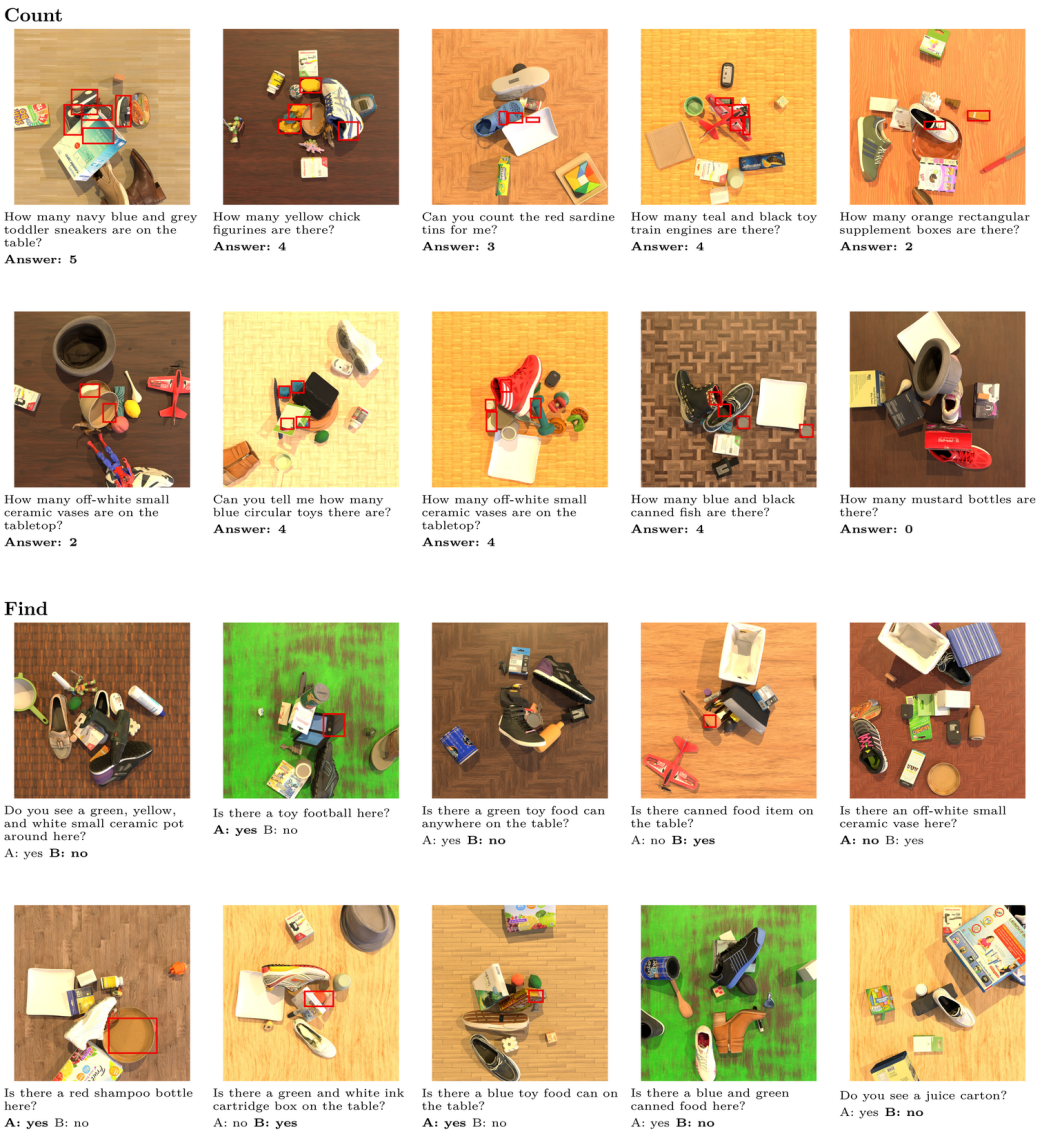

Figure 6: **MG-VQA-Bench: Count and Find.** Ten examples per task. Correct answers are in bold. Red boxes show ground-truth target objects; absent targets are not boxed.

C Environment and Benchmark Construction

C.1 Benchmark Scene Construction

We construct candidate scenes by sampling up to 15 objects from the annotated asset pool, placing them in the tabletop workspace, and simulating settling. We use simulator state only to construct ground truth and validate benchmark instances; agents receive rendered observations and tool outputs rather than privileged object poses or labels.

`Count`. We sample a queried object category or name and a target multiplicity between 0 and 5. If targets are present, they are spawned before distractor objects, increasing the likelihood that subsequent objects occlude them after physical settling. We retain instances in which answer-critical target objects have at most 10% pixel visibility in the initial top-down RGB observation.

`Find`. We construct equal numbers of present and absent instances. For present cases, the queried target is placed before distractors and retained only when its initial pixel visibility is at most 10%. For absent cases, no object matching the queried name or category is present in the scene. A positive answer requires both identifying that the target is present and localizing one visible or revealed instance.

`Beneath`. After settling, we enumerate candidate cover–hidden-object pairs. For each candidate cover object, we render a counterfactual image in which that cover is transparent and measure the resulting visibility increase for nearby objects. We select pairs for which removing the cover substantially reveals a hidden object. We retain instances in which the hidden answer object has at most 10% pixel visibility in the initial observation. For questions where the answer is "None of the above", the hidden object is not present as any of the other 3 options.

`Compare`. We select three texture-mappable objects with distinct physical volumes. For each object, we identify a sufficiently large upward-facing mesh patch, verify that it lies in the camera frame and is self-visible in isolation, and bake one of the standardized A/B/C labels into its texture map. We construct the scene such that at least one of the labels required to associate A/B/C with their corresponding objects is initially occluded. The question asks for the largest or smallest of the three labeled objects.

C.2 Human Verification Interface

Each candidate instance is independently reviewed by two human annotators using a custom verification interface (Figure 7). Annotators assess three properties: (i) whether the natural-language question, answer options, and ground-truth answer are unambiguous; (ii) whether the initial observation withholds answer-critical evidence; and (iii) whether feasible clutter removal reveals sufficient evidence to answer the question.

The interface restores the initial simulator state and displays an auxiliary verification rendering in which non-answer-critical clutter is made transparent. It highlights the evidence needed to determine the answer: target instances for `Count` and `Find`, the queried cover and hidden object for `Beneath`, and the objects bearing the A/B/C labels for `Compare`. This view is available only to annotators during benchmark construction and is never exposed to evaluated agents. For `Compare`, annotators additionally verified that the chosen objects are visibly distinct in occupied volume and thus can be judged for their volume comparison in the top-down view.

When annotators find that a valid scene has ambiguous wording or answer options, they edit the question metadata. We exclude instances with invalid ground truth, irrecoverable answer-critical evidence, or unresolved ambiguity.

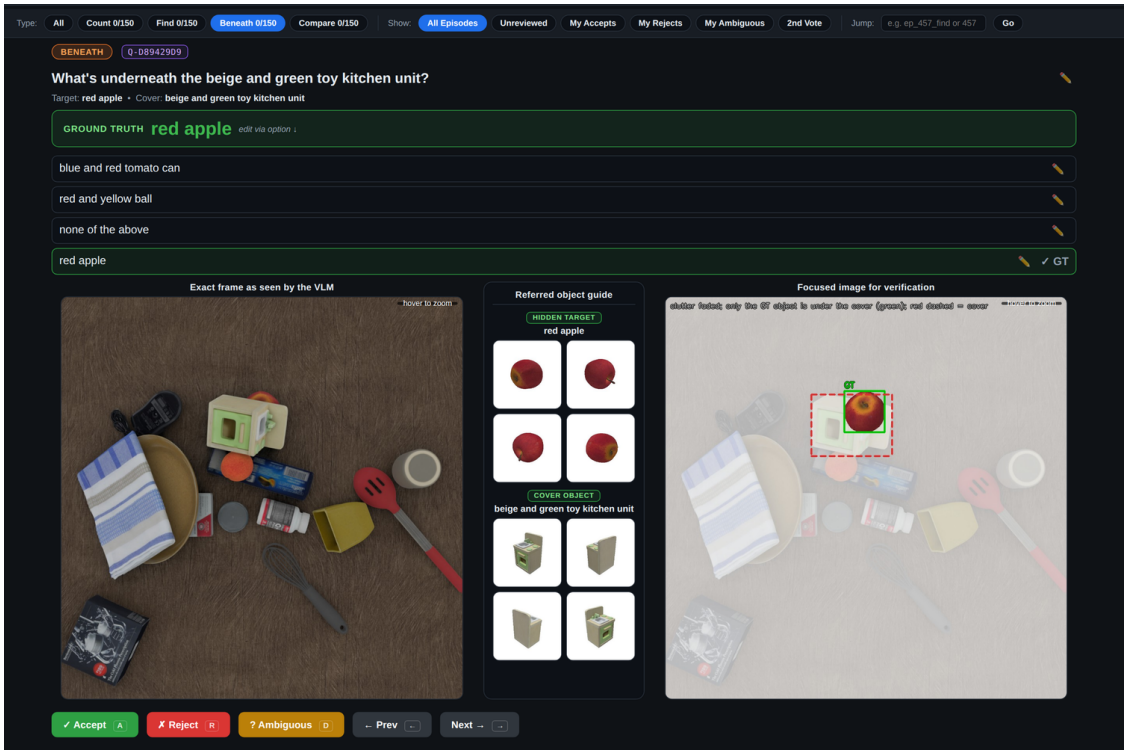

Figure 7: **Human verification interface.** The custom review application displays the initial observation (left), reference object views (center), and a focused verification image (right) for the red-apple `Beneath` example. The privileged verification view is not exposed to evaluated agents.

C.3 MG-VQA-Sim Implementation Details

MG-VQA-Sim is implemented in PyBullet (Coumans & Bai, 2016–2021), which simulates gravity, friction, rigid-body contact, grasping, and pushing. The environment uses a UR5 manipulator equipped with a Robotiq 2F-140 parallel-jaw gripper. We render 1024×1024 top-down RGB observations using Blender-EEVEE.

Our asset pool contains 319 real-scanned everyday objects collected from MegaPose (Labbé et al., 2022), Digital Twin Catalogue (Dong et al., 2025), and the Benchmark for Pose Estimation (Nguyen et al., 2025). For every object, we produce a short name, color, category, and physical description from multi-view renders using Gemini 3 Flash, followed by manual review. This review ensures that annotations are view-agnostic and do not depend on a particular rendered viewpoint.

Each episode contains a UR5 robot, a clearing basket within the robot workspace, and up to 15 objects placed in a 50×50 cm tabletop area. The clearing basket allows the agent to temporarily relocate a grasped object while inspecting clutter beneath or around it. To vary visual appearance across episodes, we sample one of 108 tabletop textures from AmbientCG.[1] After object placement, we simulate 30 seconds of settling before rendering the initial observation. We retain scenes in which all initialized objects remain within the top-down camera field of view.

[1] `https://ambientcg.com/`; CC0 1.0 Universal.

C.4 TOOL APIS

Each episode exposes four information-gathering tools and one terminal answer tool. The VLM may invoke information-gathering tools repeatedly before submitting its response through `answer_question`.

`perceive_scene`. This tool returns a top-down scene graph, a brief textual description of the workspace, a list of visible objects and their 2D image points, and pairwise spatial-relation predicates. Its output summarizes the currently visible state of the scene and can change after manipulation. All `perceive_scene` tool calls use Gemini 3.6 Flash to generate the scene graph, regardless of the evaluated agent model.

`find_object`. The agent supplies a 2D image point from the scene graph and this tool returns the continuous segmentation mask containing that point, using SAM3 (Carion et al., 2025). This mask is returned as an explicit perception output and serves as input to the grasping pipeline.

`grasp`. The grasp tool acts on the segmentation mask generated by `find_object`. We it along with the top-down depth map, and camera intrinsics to obtain a 3D object representation, then invoke GraspGen (Murali et al., 2026) to generate a 6-DoF grasp proposal. If execution succeeds, the object is lifted and placed into the clearing basket. A grasp may fail because the predicted mask is inaccurate, no feasible grasp proposal survives filtering, the selected pose is not reachable, or contact with nearby clutter prevents a successful lift.

`push`. The push tool acts on the segmentation mask generated by `find_object` in the previous step and one of four directions by the VLM: left, right, up, or down. It uses the mask to first compute the object's 2D centroid, The robot moves its closed gripper to an empty location on the opposite side of the object centroid and executes a linear sweep. Pushing provides an alternative to grasping for creating local clearance, moving broad or flat objects, and exposing partially occluded regions. Because pushes involve contact with dense clutter, they may produce partial displacement or move nearby objects in addition to the intended target.

`answer_question`. This terminal tool records the model's answer and ends the episode. For `Find`, a positive answer additionally requires a 2D point locating a target instance.

System and user prompts used by VLMs in MG-VQA-Bench interactive evaluations are provided in Figure 8 and Figure 9 respectively.

D MG-VQA-HARD EXAMPLES

We show sample question and answer pairs using using their initial top-down observations in Figure 10.

System prompt

Role and tool access

You are a spatial reasoning expert answering questions about cluttered tabletop scenes. Tool access depends on the evaluation mode: *Direct* (one image, no tools), *Perception* (scene perception, segmentation, answering), or *Manipulation* (the above, plus grasping and pushing).

Tools

`perceive_scene()` Returns a YAML scene graph of visible objects with descriptions, spatial relations, and object points.

`find_object(point)` Segments one object with SAM3 at a scene-graph point and returns the mask.

`grasp()` Attempts to grasp the segmented object and place it in a clearing basket. A failure may still shift objects; follow the reported outcome.

`push(direction)` Pushes the segmented object left, right, up, or down by 10 cm in image coordinates. Neighbouring objects may also move.

`answer_question(answer)` Submits the answer and ends the episode.

Interaction constraints

Call exactly one tool per turn. Manipulation draws on a shared budget of 10 grasp or push calls; perception is free. Once the budget is exhausted, only perception and answering remain, and five consecutive perception-only calls terminate the episode. Answer when the evidence is sufficient; otherwise investigate a relevant occluder.

Response format and answer schemas

Return one JSON object with no surrounding prose: `answer` and `reasoning` in *Direct* mode, or `tool`, `reasoning`, and the tool arguments otherwise. Answers are a non-negative integer (`Count`), A or B (`Find`), A–D (`Beneath`), or A–C for the largest or smallest physical-volume host of circle-A, square-B, or triangle-C (`Compare`; all badges exist, and their standardized design is not a volume cue). A yes answer to `Find` additionally requires a `point` on the visible object, given as $[\mathrm{row}, \mathrm{col}]$ on $[0, 1000]$ with the origin at the top-left.

Example responses

```
{"tool": "perceive_scene", "reasoning": "Inspect the scene."}
{"tool": "find_object", "point": [445, 697], "reasoning": "Segment the can."}
{"tool": "grasp", "reasoning": "Remove the segmented object to reveal what is underneath."}
{"tool": "push", "direction": "left", "reasoning": "Create space before grasping."}
{"tool": "answer_question", "answer": "B", "point": [340, 520], "reasoning": "Target visible; B means yes."}
```

Figure 8: System prompt for VLMs evaluated in MG-VQA-Bench, identical across all models. Episodes terminated by five consecutive perception-only calls or by exhausting the manipulation budget are scored as incorrect.

User prompt (repeated at every model turn)

Status. ⟨*perception-only, or remaining manipulation budget*⟩ Step: ⟨*step*⟩. Last tool: ⟨*last tool*⟩.

Previous tool response. ⟨*scene graph, segmentation result, or reported manipulation outcome*⟩. Omitted on the first turn.

Question. ⟨*original question*⟩ **Task.** ⟨*count* | *find* | *beneath* | *compare*⟩ **Answer options.** ⟨*options, if applicable*⟩

Image. Current top-down RGB. Previous images and tool responses remain in the conversation history. Respond using the JSON schema specified in the system prompt.

Search guidance (ablation only)

One task-specific sentence is appended, on the first turn only or on every turn in *Search-persistent*. For `Count`: instances of the target may be buried in the pile, so clear obstructing objects before committing to a number; `Find`: the target may be present but hidden, so not seeing it in the current view is not evidence of absence; `Beneath`: the answer lies underneath the referred object, so move that object before selecting an option; `Compare`: some badges may be occluded, so uncover the ones you cannot see before comparing.

Direct mode

A single turn containing only the question, task, answer options, and image.

Figure 9: User prompt for VLMs evaluated in MG-VQA-Bench, supplying the current observation, the previous tool result, the question with its answer options, and optional ablation context. Values substituted at runtime are shown as ⟨*placeholders*⟩.



Figure 10: Sample scenes in MG-VQA-Hard. Three per task. Correct answers are bold. Red boxes show privileged ground-truth target extents, including hidden portions; these overlays are not agent inputs.