Advancing Creative Physical Intelligence via Tool Use in Large Multimodal Models

Cheng Qian*, **Hyeonjeong Ha***, **Jiayu Liu, Jeonghwan Kim, Emre Can Acikgoz, Bingxuan Li, Kunlun Zhu, Jiateng Liu, Aditi Tiwari, Zhenhailong Wang, Xiusi Chen, Heng Ji**

University of Illinois Urbana-Champaign

Abstract

Large multimodal models (LMMs) have advanced rapidly in perception and reasoning, yet it remains unclear whether they can discover visually grounded, physically feasible solutions in open-ended environments. We first introduce **MM-CreativityBench**, a benchmark for affordance-grounded creative tool use, where models must inspect scenes, entities, and parts to identify non-obvious object uses grounded in visual evidence. Our evaluation shows that current LMMs often fail to sustain grounded exploration: they overlook relevant entities, under-examine critical parts, or hallucinate unsupported attributes. To address these failures, we propose **affordance-grounded alignment**, framing creative tool use as a preference learning problem. Using Direct Preference Optimization and supervision from an affordance knowledge base, we train models to favor visually grounded attribute–affordance reasoning over hallucinated alternatives while improving exploration efficiency. Our results yield consistent gains in correct entity and part selection, and reduce hallucination and grounding errors. These findings position grounded creativity as a core capability for future multimodal agents: the ability to adapt to unfamiliar environments and solve problems beyond memorized patterns or surface-level plausibility, moving closer to real human-like intelligence.

1 Introduction

In **Triarchic Theory of Intelligence** [30], human intelligence includes analytical, practical, and **creative intelligence**: the ability to generate novel and useful solutions under constraints. In resource-limited settings, this often appears as tool repurposing, where people adapt available objects beyond their intended use. Such creativity depends on grounded knowledge of object attributes and physical affordances. As in Figure 1, humans can decompose everyday objects into functional modules, such as edges, tips, and surfaces, and recombine them for new goals. These solutions are creative but not arbitrary; they reflect the non-obvious yet physically valid mappings between task goals and environmental affordances [28, 13]. In this work, we study this specific form of creativity, **creative tool repurposing**, as a concrete testbed of creative intelligence in large multimodal models (LMMs).

Despite rapid progress in LMMs, it remains unclear whether they acquire such grounded creative intelligence. Existing creativity benchmarks mainly evaluate open-ended generation or constrained problem-solving [31, 24, 17], but are often text-centric or scenario-driven. Real-world creative tool use is perception-dependent: an agent must inspect the environment, identify candidate objects and parts, and judge whether their geometry, state, and accessibility support the intended use. Without this grounding, models may produce plausible but physically invalid solutions or hallucinate unsupported

* indicates equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

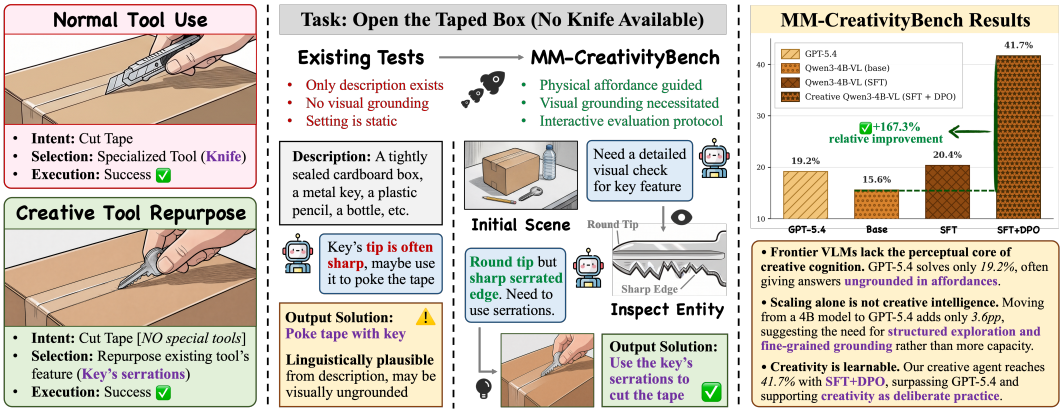


Figure 1: Beyond normal tool use, **creative tool repurposing** requires visual grounding of physical affordances, enabling the model to discover that a key's serrated edge can cut box tape. MM-CreativityBench shows that such affordance-guided reasoning is poorly captured by frontier LMMs but can be improved through training.

affordances [40, 9, 37]. This motivates our central question: *can LMMs perform creative reasoning as an evidence-driven process grounded in perception?* [18, 19, 8]

We introduce **MM-CreativityBench**, a benchmark for grounded creative problem solving in multimodal environments. Each task requires repurposing everyday objects under physical and contextual constraints, with a structured visual context containing a scene image, entity-level views, and zoomed-in part images. This design tests whether models can identify not only which object may be useful, but also which specific part supports the required function and why it is feasible. We focus on constrained creativity, where valid solutions must satisfy task requirements and be grounded in the visible scene, and evaluate whether models can interactively explore the environment, gather evidence, and select a physically valid object–part combination.

Our experiments reveal a gap between surface-level plausibility and grounded creative reasoning. Current LMMs often generate plausible solutions, but struggle to sustain evidence-based exploration: even the strongest base models achieve less than 25% gold correctness. Models frequently fixate on salient but irrelevant objects, neglect decisive parts, or infer affordances unsupported by inspected visual evidence. Some top closed-source models also underperform open-source models such as Qwen, suggesting that scaling alone is insufficient.

To address these failures, we investigate **affordance-grounded alignment**. We teach models attribute–affordance associations that connect observable physical properties to functional uses, while aligning interactive behavior toward evidence-seeking. We construct supervision that encourages models to inspect candidate entities, track unobserved parts, and ground intermediate reasoning in visual evidence, and introduce preference data with negative trajectories capturing hallucinated attributes, premature commitment, and visually unsupported reasoning. Fine-tuning Qwen3-VL models with supervised fine-tuning and direct preference optimization yields consistent gains, more than doubling performance in the best setting while improving grounding and reducing hallucination-related errors. To summarize, our contributions are:

- **Visual Creativity Benchmark:** We introduce MM-CreativityBench for evaluating grounded creative tool repurposing in visual environments.
- **Grounded Interactive Protocol:** We design an interactive setting where models inspect scenes, entities, and parts to support evidence-driven exploration.
- **Affordance-Grounded Alignment:** We show that stepwise supervision and preference optimization improve performance, grounding, and hallucination reduction.

2 Related Work

Creativity in Multimodal and Language Models. Creativity in LLMs has been studied in storytelling [1, 6], design [23, 7, 14], and ideation [29, 33, 25, 38, 35], often through novelty, diversity, and usefulness. Recent benchmarks extend this to constrained creative problem solving, including tool use and object repurposing [31, 24, 26], and to multimodal settings with non-literal image understanding,

context-aware generation, and exploration-driven decision making [15, 12, 17]. However, most remain scenario-driven: they evaluate plausible creative outputs or plans, but not how solutions are grounded in object parts, physical attributes, and visual evidence. Compared with CreativityBench [26] which already evaluates part-level repurposing and textual interaction, MM-CreativityBench adds visual scene/entity/part inspection and affordance-grounded trajectory alignment.

Affordance-Grounded Reasoning and Alignment. Affordance reasoning has been studied in physical commonsense benchmarks such as PIQA, PROST, and NEWTON [3, 2, 34], and in robotics and embodied AI for manipulation and planning [22, 16, 10, 4, 5]. Recent LMMs work further models structured and part-level affordances [39, 21], but mainly for canonical affordance recognition or action feasibility. In parallel, SFT and DPO-based alignment [27, 32, 20] improve reasoning and visual grounding through supervised or preference-based learning. Our work combines these directions: we use an affordance knowledge base to build supervision and preference signals for grounded attribute–affordance reasoning, enabling models to explore object parts, reject unsupported affordances, and perform creative tool repurposing. See detailed comparison in Section B and Table 4.

3 MM-CreativityBench

3.1 Preliminary Experiment

As a preliminary probe of creative intelligence in LMMs, we evaluate GPT-4.1-mini on 100 MacGyver tool-use tasks [31], where everyday objects must be repurposed under constraints. To add visual grounding, we generate scenario images with Gemini-2.5-Pro and provide only non-visual constraints in the text, forcing the model to identify candidate objects from the image. We compare a **direct prompt** with a **structured affordance-level CoT** prompt [36], which guides the model to decompose objects into parts, infer attributes and affordances, and check constraints. GPT-5.2 serves as the judge across six dimensions: **Correctness**, **Feasibility**, **Physical Grounding**, **Constraint Coverage**, **Tool Usage**, and **Creativity**. Details are provided in Section C.

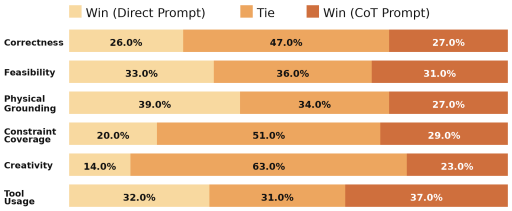


Figure 2: **Preliminary results:** Comparison between direct and CoT prompts on creative tool use tasks.

As shown in Figure 2, CoT prompt improves procedural aspects such as Constraint Coverage, Tool Usage, and Creativity, but yields only limited gains in Correctness, Feasibility, and Physical Grounding. While structured reasoning helps organize responses, models still produce plausible, creative uses without verifying the required physical attributes from the image. This motivates MM-CreativityBench as a benchmark for interactive, part-level grounding, as well as our affordance-grounded alignment approach that encourages models to explore evidence, connect attributes to affordances, and reject visually unsupported solutions.

3.2 Benchmark Task Construction

The preliminary study shows that structured prompting can organize creative reasoning, but does not reliably ground final answers in the visual and physical attributes of specific object parts. We therefore build MM-CreativityBench from a part-level affordance knowledge base, giving each task an explicit evidence structure for the correct creative solution.

Creative affordance knowledge base. All gold triples come from the CreativityBench affordance knowledge base[26]; we newly instantiate multimodal tasks and trajectories from its part-level attribute–affordance annotations (Section D.1). Formally, each entity $e \in \mathcal{E}$ is decomposed into functional parts $P(e) = \{p_1, \ldots, p_m\}$. Each part $p \in P(e)$ has an attribute set $A(p) = A^{\text{phy}}(p) \cup A^{\text{state}}(p)$, where $A^{\text{phy}}(p)$ captures stable properties such as geometry, material, rigidity, sharpness, hollowness, or texture, and $A^{\text{state}}(p)$ captures situational properties such as being open, clean, intact, accessible, or detachable. These annotations provide the fine-grained evidence needed to judge whether a part can support a novel use.

Reverse task construction. We construct each instance as an inverse grounding problem: instead of writing scenarios and then labeling answers, we first sample a target entity–part pair (e^*, p^*) and a gold affordance f^*, forming $g = (e^*, p^*, f^*)$. We then generate a task description x that requires f^* without revealing the target entity or part, and sample distractor entities E^-:

$$T = (x, E, g), \qquad E = \{e^*\} \cup E^-, \qquad g = (e^*, p^*, f^*).$$

Distractors make the task diagnostic: some have parts with affordances similar to f^* but lack a decisive physical or state attribute, while others are scene-plausible co-occurring objects that cannot satisfy the constraints. Thus, success requires fine-grained entity–part grounding rather than object priors alone. After filtering for gold validity, distractor separability, scene coherence, and visual observability, we obtain 333 held-out test tasks and 868 disjoint training tasks for trajectory sampling. Details of generation, distractor construction, filtering, and human verification are in Section D.2.

Multimodal grounding via image generation. After constructing each symbolic task $T = (x, E, g)$, we generate images at three granularities: environment, entity, and part. This mirrors the required interaction process: observe the full scene, inspect candidate entities, and verify decisive part-level evidence. For each task, using image generation prompt of $\pi_{\text{ent}}, \pi_{\text{part}}$, and π_{env}, we generate

$$I_e = \pi_{\text{ent}}(e, P(e), A), \qquad I_{e,p} = \pi_{\text{part}}(e, p, A(p), I_e), \qquad I_{\text{env}} = \pi_{\text{env}}(x, E, \{I_e : e \in E\}).$$

Here, I_e is the full-object view, $I_{e,p}$ is the zoomed-in view of part p, and I_{env} places all candidate entities in a coherent scene. This three-level design is crucial because distractors are plausible at the object level, while the correct answer often depends on local part attributes. The benchmark therefore requires models to navigate $I_{\text{env}} \to I_e \to I_{e,p} \to (e^*, p^*, f^*)$ and ground the final solution in inspected visual evidence. Image generation details are provided in Section D.3.

3.3 Training Trajectory Construction

We use the same task structure to construct multi-turn training trajectories. Since grounded creative tool use requires not only selecting the final entity–part pair, but also deciding what to inspect, how to interpret evidence, and when to reject plausible but invalid alternatives, our trajectories supervise evidence-seeking behavior from scene-level search to part-level affordance grounding.

Interactive trajectory format. For each multimodal task $\mathcal{T} = (x, I_{\text{env}}, E, g)$ with gold solution $g = (e^*, p^*, f^*)$, an interaction trajectory is

$$\tau = \{(o_t, r_t)\}_{t=1}^{T}, \qquad o_t = (u_t, I_t), \qquad r_t = (z_t, a_t).$$

Here, u_t denotes the feedback, I_t denotes the visual observation, z_t denotes the model reasoning, and a_t denotes a structured action:

$$a_t \in \{\texttt{inspect_entity}(e),\ \texttt{inspect_part}(e, p),\ \texttt{answer}(e, p, h)\},$$

where $e \in E$, $p \in P(e)$, and h describes the intended use. This mirrors the benchmark hierarchy: the initial observation is the scene image I_{env}; inspecting an entity returns its full-object image I_e and part list $P(e)$; inspecting a part returns $I_{e,p}$ with visibility/accessibility text and, where needed, other attribute disambiguation (Section D.3). Thus, each action determines the next evidence available to the model.

Knowledge-guided exploration stack. To construct positive trajectories, we maintain an ordered exploration stack $\mathcal{S}_t$ over candidate entities and parts. Its top element gives the next inspection target. We define $J(e, p) \in \{0, 1\}$, where $J(e, p) = 1$ means part p of entity e has an affordance relevant to the target affordance f^* according to the knowledge base. The stack evolves as follows:

- **Initialization:** Given the scene and task, the model proposes an ordered list of candidate entities to initialize $\mathcal{S}_t$, thereby directing early exploration toward likely relevant objects.
- **Inspect Entity:** When an entity e is inspected, it is removed from the stack, and its affordance-relevant parts $\forall p \in P(e), J(e, p) = 1$ are pushed for part-level inspection. This turns coarse entity-level exploration into finer part-level verification.
- **Inspect Part:** When a part p is inspected, it is removed from the stack and assigned a binary judgment $b_t \in 0, 1$, indicating whether its observed attributes satisfy the task requirements.
- **Answer:** Exploration stops when no unexplored entity or part remains. In the final answer turn, the model compares all inspected parts with $b_t = 1$ and selects the final pair.

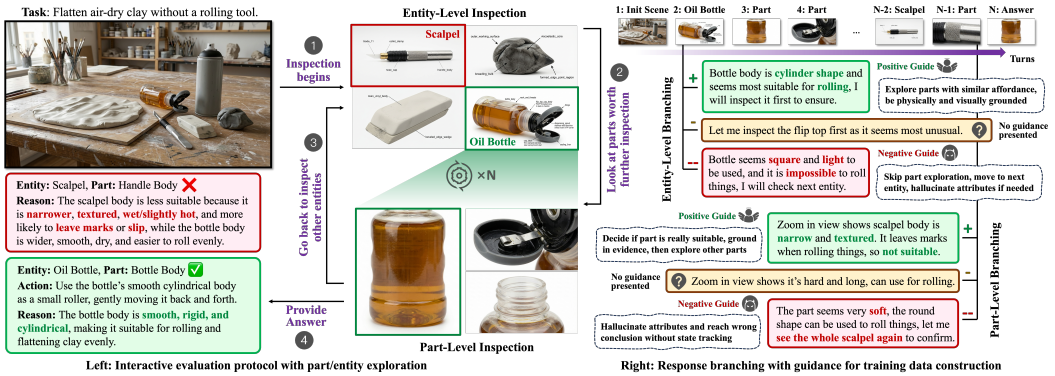


Figure 3: Interactive evaluation and training in MM-CreativityBench, where models inspect scenes, entities, and parts to ground creative tool use while learning to avoid hallucinated affordances.

This produces coarse-to-fine trajectories: models first search entities, then verify affordance-relevant parts, and finally choose among plausible candidates. This is crucial because many distractors are affordance-similar, requiring the model to select the gold pair (e^*, p^*) based on decisive attributes.

Three-branch trajectory sampling. The stack specifies what to inspect, but we also need to supervise how the model reasons about each inspection. As illustrated in Figure 3, at each shared context c_t, we sample three aligned branches with the same response format:

$$r_t^b = (z_t^b, a_t^b), \qquad b \in \{+, -, --\},$$

- **Positive branch** $(+)$**:** Guided by structured knowledge, relevant attributes, and the gold solution. It justifies actions with grounded evidence and forms the SFT trajectory.
- **Negative branch** $(-)$**:** Uses only normal feedback including the task and images. It captures realistic inference-time mistakes, such as missing decisive parts or choosing suboptimal candidates.
- **Hard-negative branch** $(--)$**:** Preserves fluent reasoning and valid actions, but is guided toward misleading conclusions, such as hallucinated attributes, object-level priors, or affordance-similar distractors lacking key evidence.

Only the positive branch updates the stack, keeping future observations coherent. Negative branches are sampled at the same states as rejected alternatives, yielding turn-level triples

$$(c_t, r_t^+, r_t^-, r_t^{--}),$$

where r_t^+ is the preferred grounded response and r_t^-, r_t^{--} are increasingly difficult rejected responses. These data support both stages of training: positive trajectories teach systematic exploration through SFT, while positive–negative comparisons enable DPO to prefer visually grounded attribute–affordance reasoning over fluent but unsupported alternatives. More details are provided in Section E.

3.4 Affordance-Grounded Alignment

We align the model in two stages: SFT learns structured exploration from positive trajectories, while turn-level DPO improves attribute–affordance grounding by contrasting grounded reasoning with plausible but visually and physically unsupported alternatives.

Supervised Fine-Tuning (SFT). We first perform supervised fine-tuning on positive trajectories $\tau^+ = \{(c_t, r_t^+)\}_{t=1}^T$. Given context c_t, the model imitates $r_t^+ = (z_t^+, a_t^+)$ with a standard token-level cross-entropy objective over complete multi-turn interactions. Training on full trajectories, rather than only final answers, teaches the model to select candidate entities, inspect relevant parts, interpret attributes, and compare entity–part pairs before answering. Since these trajectories are structurally guided by our affordance knowledge base and exploration stack, SFT mainly learns an exploration policy and does not explicitly penalize spurious attribute–affordance associations.

Turn-Level Direct Preference Optimization (DPO). To reduce the gap between guided training and unguided inference, we apply DPO at each shared context c_t. We form preference pairs $(c_t, r_t^+, r_t^{\text{rej}})$, where r_t^+ is the grounded positive response and $r_t^{\text{rej}} \in \{r_t^-, r_t^{--}\}$ comes from the negative or hard-negative branch. Rejected responses keep valid action formats and plausible entity–part choices,

Model	Gold Correct	Entity Correct	Turns	Avg. Distinct Explored		Gold Entity Explored Before Answer		Gold Part Explored Before Answer	
				Entities	Parts	Entity Correct	Entity Wrong	Part Correct	Part Wrong
GPT-5.4	0.192	0.435	4.177	1.661	1.492	0.510	0.218	0.422	0.059
GPT-5.4 Mini	0.183	0.408	4.072	1.360	0.706	0.662	0.193	0.279	0.033
Qwen3-VL-8B-Instruct	0.192	0.441	13.450	4.979	3.766	0.993	0.747	0.953	0.201
Qwen3-VL-32B-Instruct	**0.240**	**0.447**	8.766	4.802	2.309	1.000	0.750	1.000	0.111
InternVL3.5-14B	0.150	0.345	4.847	1.811	1.700	1.000	0.211	0.960	0.032
InternVL3.5-38B	0.156	0.426	6.991	3.565	1.775	1.000	0.524	0.942	0.068
Gemma-4-26B-A4B-it	0.183	0.402	5.330	2.679	1.574	1.000	0.477	0.902	0.040
Gemma-4-31B-it	0.165	0.354	3.802	1.982	0.796	1.000	0.256	0.545	0.018

Table 2: The benchmarking results on MM-CreativityBench. Models often locate the relevant entity but struggle with fine-grained gold-part grounding. Larger exploration traces improve evidence coverage but do not guarantee correct answers, revealing bottlenecks in visual evidence use.

but misinterpret or overclaim visual evidence. Contrasting responses under the same context trains the model to prefer reasoning that justifies affordances with observed physical or state attributes, directly targeting attribute–affordance failures under multimodal uncertainty. Full objectives and trajectory details are provided in Section F.

4 Experiment

4.1 Experiment Setups

Benchmark Evaluation Protocol. We use the interactive protocol as illustrated in Figure 3: each example starts with a scenario image containing multiple entities, and the model may inspect entities and parts for closer views before answering. The model need not inspect every region, but effective exploration should ground the final solution in object-specific visual evidence. In our main setting, the history contains the initial scenario image and the most recently inspected view. At each step, the model gives reasoning and chooses one action: inspect an entity, inspect a part, or answer. Final answers explain how the explored evidence supports a creative and grounded response. We evaluate GPT, Qwen3-VL, InternVL3.5, and Gemma-4 with maximum context length and zero temperature; full prompt details are in Section G.

Category	Statistic	Value	Category	Statistic	Value
Test Set	Data Points	333	Training Set	Data Points	868
	Number of Entities	974		Number of Entities	1,498
	Number of Parts	6,344		Number of Parts	10,080
SFT Data Points		19,533	DPO Data Points		5,000

Table 1: Overall statistics of MM-CreativityBench. The training set is used for trajectory sampling, which has no overlapping scenes, entities, or parts with the test set.

Training Implementation Details. We train Qwen3-VL-4B-Instruct and Qwen3-VL-8B-Instruct with SFT and DPO on sampled training trajectories. SFT uses only positive branches. For DPO, the context is built from positive branches, the current positive response is the chosen sample, and the rejected sample comes from either the negative branch or the hard-negative branch, yielding DPO (normal negative) and DPO (hard negative). We also evaluate two-stage SFT+DPO. All trajectories come from 868 training tasks whose scenarios and entities are disjoint from the test set. Dataset statistics are in Table 1, and hyperparameters are provided in Section G.

Evaluation Metrics. Our main metric, **Gold Correct Rate**, exactly matches the selected entity–part pair to the curated gold pair. Quality control removes equally valid in-scene alternatives (Section D.2). This measures recovery of the intended mechanism under the stated constraints, rather than all valid creative solutions. We also report **Entity Correct Rate**, which only requires the selected entity to be correct and therefore measures coarse object-level grounding. We further report interaction and grounding statistics, including the **Average Number of Exploration Turns** and the **Average Number of Distinct Entities/Parts Explored**. To measure whether final answers are supported by the interaction history, we also track whether the model inspected the gold entity and gold part before answering.

4.2 Main Results

Interactive exploration helps models find evidence, but not necessarily use it correctly. Results in Table 2 show that inspecting relevant visual evidence does not guarantee correct answers. Qwen3-VL-

Model	Gold Correct	Entity Correct	Turns	Avg. Distinct Explored		Gold Entity Explored Before Answer		Gold Part Explored Before Answer	
				Entities	Parts	Entity Correct	Entity Wrong	Part Correct	Part Wrong
Qwen3-4B-VL-Instruct	0.156	0.393	18.922	3.937	4.417	0.947	0.554	0.923	0.167
+ SFT	0.204	0.369	17.862	6.628	9.670	1.000	0.990	0.985	0.404
+ DPO (normal negative)	0.201	0.529	31.693	5.506	12.398	1.000	0.821	0.970	0.351
+ DPO (hard negative)	0.240	0.547	12.842	4.222	3.781	0.973	0.639	0.838	0.157
+ SFT + DPO (hard negative)	**0.417**	**0.583**	6.211	2.644	1.831	0.959	0.350	0.856	0.026
Qwen3-8B-VL-Instruct	0.192	0.441	13.450	4.979	3.766	0.993	0.747	0.953	0.201
+ SFT	0.273	**0.589**	15.646	6.655	7.679	1.000	0.993	1.000	0.293
+ DPO (normal negative)	0.258	0.577	18.480	5.904	6.384	1.000	0.936	0.965	0.231
+ DPO (hard negative)	0.261	0.508	9.054	4.283	2.509	0.994	0.613	0.920	0.082
+ SFT + DPO (hard negative)	**0.393**	0.583	8.069	4.066	2.327	1.000	0.576	0.939	0.059

Table 3: Interactive image-evaluation summary for Qwen3-VL base and trained models. SFT + DPO achieves the highest gold correct rates with more efficient exploration. Gold parts and entities are more frequently explored when the final answer is correct than when it is wrong, suggesting that correct answers are typically grounded in relevant exploration.

32B often inspects the gold entity before answering and achieves the best raw gold correctness among base models, yet its accuracy remains only **0.240**. InternVL3.5-14B and Gemma-4-26B-A4B-it show similar patterns: they frequently locate the correct entity in successful entity-level cases, but still obtain low gold-correct scores. Thus, failures are not only due to missing relevant regions; models also struggle to **interpret and integrate** inspected evidence. This motivates training that improves both exploration and the use of visual evidence.

The main bottleneck is fine-grained part grounding, not entity localization. Across raw models, entity correctness is much higher than gold correctness. For example, GPT-5.4 reaches 0.435 entity correctness but only **0.192** gold correctness, while InternVL3.5-38B reaches 0.426 versus **0.156**. This gap shows that models often find the right object but fail to identify the decisive part or attribute. Exploration statistics reinforce this: models inspect entities more reliably than parts, so broader search does not necessarily produce finer evidence. MM-CreativityBench therefore tests **part-sensitive visual reasoning**: locating the relevant region, extracting its physical evidence, and using it in the final decision.

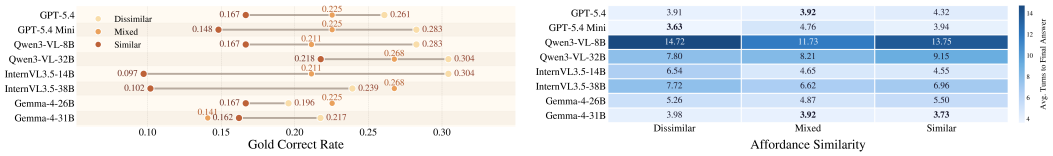
Models differ in exploration style, and scaling alone is insufficient. Table 2 shows large differences in exploration behavior. Qwen3-VL models inspect around 4.8–5.0 entities on average, while GPT-5.4 and GPT-5.4 Mini inspect only 1.66 and 1.36. However, more exploration is not always better: Qwen3-VL-8B explores far more than GPT-5.4 but reaches the same gold correctness of 0.192, and Qwen3-VL-32B improves only modestly to 0.240. Interaction turns also exceed the number of distinct explored entities and parts, suggesting redundant exploration. These results indicate that interactive visual reasoning depends on family-specific tradeoffs among search, grounding, and decision-making, rather than scale or exploration volume alone.

Takeaway 1

Models can often choose relevant entities, but still fail at fine-grained part grounding and evidence integration; interactive visual reasoning is limited more by grounding and reasoning than by exploration volume or scale.

Training improves both accuracy and interaction efficiency. As shown in Table 3, targeted training substantially improves interaction behavior. For Qwen3-4B, **SFT + DPO with hard negatives** raises gold correctness from 0.156 to **0.417** while reducing average turns from 18.92 to **6.21**. Qwen3-8B shows the same trend, improving from 0.192 to 0.393 while reducing turns from 13.45 to 8.07. These gains come not from longer search, but from more selective exploration: trained models gather useful evidence earlier, avoid unnecessary revisits, and stop when evidence is sufficient. Additional InternVL training, semantic-holdout evaluation, and a preliminary object-photo probe are reported in Section H.2; exploration analysis follows in Section 5.3.

SFT structures exploration, while hard-negative DPO improves evidence discrimination. SFT makes exploration traces more grounded and interpretable, but only partially improves final reasoning. For Qwen3-4B, it raises gold correctness from 0.156 to 0.204, while average turns remain high at 17.86. Adding **hard-negative DPO** further improves correctness to 0.417 and reduces turns to 6.21. This suggests that hard negatives may help the model distinguish grounded evidence from *plausible but misleading* alternatives, rather than simply adding more supervision. In this sense, SFT organizes exploration, while DPO encourages more reliable fine-grained attribute–affordance decisions.



(a) Performance under different affordance similarity levels.

(b) Average number of exploration turns.

Figure 4: **Effect of affordance similarity on performance and exploration.** As more entities share similar affordances, model performance often degrades, while exploration turns remain stable. This suggests that failures derive from weaker visual grounding and affordance disambiguation rather than insufficient exploration.

Takeaway 2

Training improves both accuracy and efficiency: SFT organizes exploration, while DPO helps models reject misleading evidence and commit earlier to valid grounding paths.

5 Analysis

5.1 Affordance Similarity Reveals Limits in Fine-Grained Visual Grounding

As shown in Figure 4, across environments with different affordance compositions, **performance degrades as more entities share similar affordances**. This indicates that the core challenge is not only identifying plausible candidate tools, but also disambiguating among candidates with overlapping affordances. Resolving such ambiguity requires fine-grained grounding in visual and physical attributes. However, current LMMs appear to rely on coarse affordance representations: they can often infer which category of object may be useful, but struggle to determine which specific object or part is physically best suited for the task. As a result, models frequently choose tools that are functionally plausible but unsupported by the necessary attribute-level evidence.

Notably, the average number of exploration turns remains largely constant across similarity levels. Even in a more ambiguous environment (i.e., similar), models do not substantially increase inspection or verification behavior before committing to an answer. This suggests that the observed **performance drop is not primarily caused by exploration alone**, but by limitations in fine-grained visual grounding and comparative affordance evaluation. These findings are further reflected in the failure modes that we will discuss in Section 5.4.

Takeaway 3

Affordance-similar distractors primarily challenge models at the level of fine-grained visual grounding and affordance disambiguation rather than exploration effort.

5.2 Impact of Visual Grounding and Interaction Dynamics

We analyze the impact of visual input under different image conditions in Figure 5. **No Image** setting consistently produces the lowest gold correct rate, showing that MM-CreativityBench cannot be reliably solved from text alone and needs visual grounding. Adding visual context via either the **Last Image** or **All Images** generally enhances performance, though the gains vary across models.



Figure 5: Gold correct rate across different input image conditions.

Notably, models with longer average interaction horizons, such as Qwen3-VL (13.45 turns for 8B and 8.77 turns for 32B) and InternVL (6.99 turns for 38B), benefit more from the Last Image condition, which often matches or exceeds performance under All Images. This suggests that models capable of iteratively refining beliefs over candidate entity–part pairs (e, p) can often make grounded decisions using only the most recent and task-relevant visual observation I_t. Overall, the results highlight that grounded creative problem solving depends on both **access to visual evidence** and the **ability to incorporate it through interaction**.

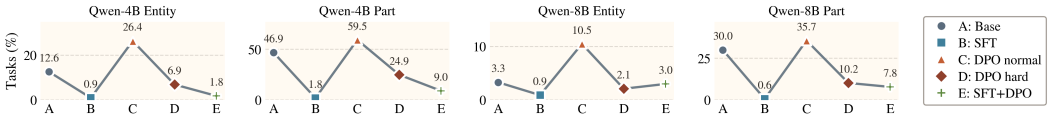


Figure 6: **Exploration repetition rates** across the base and trained 4B and 8B models. SFT and SFT+DPO substantially reduce repetition, indicating clearer state tracking and more effective, efficient exploration.

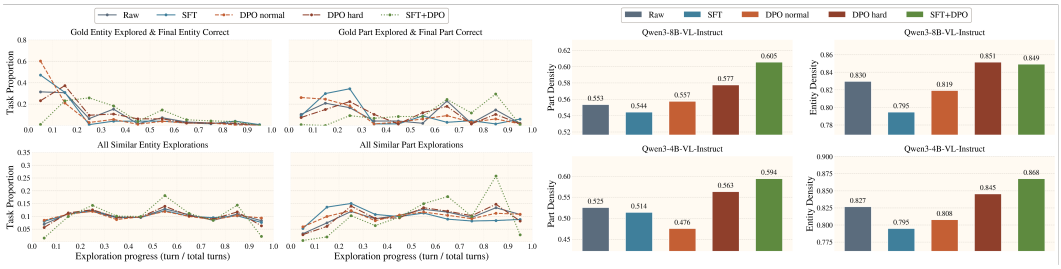


Figure 7: **Exploration progress and similarity density** for Qwen3-VL variants. Left: Qwen3-4B timing distributions for gold and similar entity/part explorations. Right: average part/entity similarity density for Qwen3-8B and Qwen3-4B across Raw, SFT, DPO, and SFT+DPO variants.

Takeaway 4

Visual input improves performance in the evaluated conditions; isolating its contribution requires matched textual feedback, including the visibility/accessibility annotations used during part inspection.

5.3 How Training Improves Exploration Efficiency and Semantic Focus

We evaluate whether training improves not only final accuracy but also the quality of intermediate exploration, using three trajectory-level metrics. **Repetition Rate** measures how often the model revisits previously explored entities or parts, where lower values reflect better state tracking efficiency. **Similarity Density** measures how concentrated exploration is around useful hypotheses: part-level density is the fraction of explored parts that are gold part or affordance-similar to it, while entity-level density is the fraction of explored entities that are gold entity or contain at least one affordance-similar part. Finally, **Exploration Progress** records when such candidates are discovered along the trajectory using the normalized turn index, turn/total turns.

Figure 6 shows that SFT, DPO with hard negatives, and SFT+DPO all substantially reduce redundant exploration. For example, on Qwen3-4B, SFT+DPO lowers part repetition from 46.9% to 9.0%, and entity repetition from 12.6% to 1.8%; a similar trend holds for Qwen3-8B. These reductions suggest that our training improves the model's ability to **maintain an exploration state** and avoid revisiting already inspected regions. This benefit likely comes from our positive trajectories, which explicitly supervise an exploration stack during action selection. In contrast, DPO with normal negatives is less stable without this structured SFT prior, often leading to more repeated and inefficient exploration.

Figure 7 further highlights the complementary roles of SFT and DPO. As shown in the right four panels, while SFT reduces redundant exploration, it can also narrow exploration, lowering part and entity similarity density than the base model. Adding DPO restores semantic focus, helping the model prioritize affordance-relevant candidates; for example, SFT+DPO achieves the highest part similarity density for both 4B and 8B models. The exploration progress curves show that these gains are not due to premature guessing: unlike other variants that mostly find useful candidates early, SFT+DPO continues to discover affordance-relevant entities and parts throughout the later exploration stages. Overall, the two stages are complementary: **SFT improves exploration discipline and state tracking, while DPO steers exploration toward semantically useful hypotheses**.

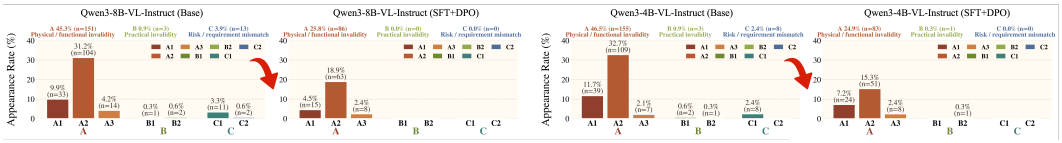


Figure 8: **Primary error-category rates** reveal that SFT+DPO substantially reduces physical/functional invalidity errors, especially affordance mismatch (A2), and removes most practical and risk/constraint errors.

Takeaway 5

SFT+DPO produces the strongest exploration behavior: simultaneously reducing redundant queries and increasing exploration of affordance-relevant entities and parts. This suggests that the trained model answers more correctly, but also searches more deliberately and efficiently.

5.4 Error Category Analysis

We categorize incorrect predictions by their primary failure reason. Category A covers **physical or functional invalidity**: hallucinated affordances (A1), affordance mismatch caused by unsuitable geometry/material/mechanics (A2), and performance shortfalls where the predicted part is only partially suitable but lacks sufficient stability, reach, capacity, precision, or retention (A3). Category B covers **practical infeasibility**, including destructive workarounds (B1) and context/accessibility issues (B2). Category C covers **risk or requirement mismatch**, including safety or damage risks (C1) and explicit constraint violations (C2). We use GPT-5.4 for scalable automatic categorization.

Figure 8 shows that SFT+DPO primarily reduces the dominant failure mode: errors in physical affordance grounding. For Qwen3-8B, total Category A errors decrease from 45.3% in the base model to 25.8% after SFT+DPO, while Category B and C errors are eliminated. The largest reduction occurs in A2, affordance mismatch, which drops from 31.2% to 18.9%. This indicates that the trained model is less likely to select parts whose shape, material, or mechanics are incompatible with the intended use. Qwen3-4B exhibits the same trend. Thus, the improvement is not merely a broad accuracy gain or the removal of rare practical and safety mistakes; it directly targets the main bottleneck in the task. The remaining errors are still mostly A-type, suggesting that physical affordance judgment remains challenging, but SFT+DPO substantially improves the model's ability to choose mechanically plausible parts.

We further study the impact of **affordance typicality** and **prompt variation**, and provide full **case studies** illustrating how training improves LMM capabilities across different dimensions. Due to space limit, detailed findings are provided in Section I.

6 Conclusion

We introduced MM-CreativityBench, a benchmark for evaluating visually grounded creative tool repurposing in multimodal environments. By requiring models to interactively inspect scenes, entities, and parts, the benchmark reveals that current LMMs often struggle to ground creative solutions in fine-grained visual and physical evidence. We further showed that affordance-grounded alignment, especially with hard negative preference signals, can improve both accuracy and exploration efficiency. Looking forward, we hope MM-CreativityBench will support future work on multimodal agents that can reason more robustly about physical affordances, adapt to unfamiliar environments, and solve open-ended problems through evidence-driven exploration rather than surface-level plausibility. Toward this goal, a natural next step is to assess transfer from our controlled synthetic setting with limited textual disambiguation to independent real-image part-grounding tasks, building on the preliminary evidence from our semantic-holdout and object-photo probes.

7 Acknowledgment

This research is based upon work supported by DARPA ITM Program No. FA8650-23-C-7316, the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 560000C260018, CapitalOne-Illinois Center for Generative AI Safety, Knowledge

Systems, and Cybersecurity (ASKS), Amazon-Illinois Center on AI for Interactive Conversational Experiences (AICE). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, IARPA, ODNI, DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This research used the Delta and DeltaAI advanced computing and data resources, which are supported by the National Science Foundation (award OAC 2320345 and award OAC 2005572) and the State of Illinois. Delta and DeltaAI are joint efforts of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications.

References

[1] Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. Storium: A dataset and evaluation platform for machine-in-the-loop story generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, 2020.

[2] Stéphane Aroca-Ouellette, Cory Paik, Alessandro Roncone, and Katharina von der Wense. Prost: Physical reasoning about objects through space and time. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4597–4608, 2021.

[3] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

[4] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. *URL https://arxiv.org/abs/2307.15818*, 1:2, 2024.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

[8] Qingxing Cao, Junhao Cheng, Xiaodan Liang, and Liang Lin. Visdiahalbench: A visual dialogue benchmark for diagnosing hallucination in large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12161–12176, 2024.

[9] Xuweiyi Chen, Ziqiao Ma, Xuejun Zhang, Sihan Xu, Jianing Yang, David F Fouhey, Joyce Chai, and Shengyi Qian. Multi-object hallucination in vision language models. *Advances in Neural Information Processing Systems*, 37:44393–44418, 2024.

[10] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. Learning affordance segmentation for real-world robotic manipulation via synthetic images. *IEEE Robotics and Automation Letters*, 4(2): 1140–1147, 2019.

[11] Yubo Dong, Xukun Zhu, Zhengzhe Pan, Linchao Zhu, and Yi Yang. Villageragent: A graph-based multi-agent framework for coordinating complex task dependencies in minecraft. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16290–16314, 2024.

[12] Xinyu Fang, Zhijian Chen, Kai Lan, Lixin Ma, Shengyuan Ding, Yingji Liang, Xiangyu Zhao, Farong Wen, Zicheng Zhang, Guofeng Zhang, et al. Creation-mmbench: Assessing context-aware creative intelligence in mllms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 447–456, 2025.

[13] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[14] Hyeonjeong Ha, Xiaomeng Jin, Jeonghwan Kim, Jiateng Liu, Zhenhailong Wang, Khanh Duy Nguyen, Ansel Blume, Nanyun Peng, Kai-Wei Chang, and Heng Ji. Synthia: Novel concept design with affordance composition. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20939–20958, 2025.

[15] Zhongzhan Huang, Shanshan Zhong, Pan Zhou, Shanghua Gao, Marinka Zitnik, and Liang Lin. A causality-aware paradigm for evaluating creativity of multimodal large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[16] Lorenzo Jamone, Emre Ugur, Angelo Cangelosi, Luciano Fadiga, Alexandre Bernardino, Justus Piater, and José Santos-Victor. Affordances in psychology, neuroscience, and robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*, 10(1):4–25, 2016.

[17] Seungwon Lim, Sungwoong Kim, Jihwan Yu, Sungjae Lee, Jiwan Chung, and Youngjae Yu. Visescape: A benchmark for evaluating exploration-driven decision-making in virtual escape rooms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16031–16058, 2025.

[18] Shuo Liu, Kaining Ying, Hao Zhang, Yue Yang, Yuqi Lin, Tianle Zhang, Chuanhao Li, Yu Qiao, Ping Luo, Wenqi Shao, et al. Convbench: A multi-turn conversation evaluation benchmark with hierarchical capability for large vision-language models. *arXiv preprint arXiv:2403.20194*, 2024.

[19] Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, et al. Visualagentbench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*, 2024.

[20] Ziyu Liu, Yuhang Zang, Xiaoyi Dong, Pan Zhang, Yuhang Cao, Haodong Duan, Conghui He, Yuanjun Xiong, Dahua Lin, and Jiaqi Wang. Mia-dpo: Multi-image augmented direct preference optimization for large vision-language models. *arXiv preprint arXiv:2410.17637*, 2024.

[21] Teli Ma, Zifan Wang, Jiaming Zhou, Mengmeng Wang, and Junwei Liang. Glover: Generalizable open-vocabulary affordance reasoning for task-oriented grasping. *arXiv preprint arXiv:2411.12286*, 2024.

[22] Luis Montesano, Manuel Lopes, Alexandre Bernardino, and José Santos-Victor. Learning object affordances: from sensory–motor coordination to imitation. *Ieee transactions on robotics*, 24 (1):15–26, 2008.

[23] Cheng Qian, Chi Han, Yi Fung, Yujia Qin, Zhiyuan Liu, and Heng Ji. Creator: Tool creation for disentangling abstract and concrete reasoning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6922–6939, 2023.

[24] Cheng Qian, Peixuan Han, Qinyu Luo, Bingxiang He, Xiusi Chen, Yuji Zhang, Hongyi Du, Jiarui Yao, Xiaocheng Yang, Denghui Zhang, et al. Escapebench: Pushing language models to think outside the box. *arXiv e-prints*, pages arXiv–2412, 2024.

[25] Cheng Qian, Hongyi Du, Hongru Wang, Xiusi Chen, Yuji Zhang, Avirup Sil, Chengxiang Zhai, Kathleen McKeown, and Heng Ji. Modelingagent: Bridging llms and mathematical modeling for real-world challenges. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1599–1633, 2025.

[26] Cheng Qian, Hyeonjeong Ha, Jiayu Liu, Jeonghwan Kim, Jiateng Liu, Bingxuan Li, Aditi Tiwari, Dwip Dalal, Zhenhailong Wang, Xiusi Chen, Mahdi Namazifar, Yunzhu Li, and Heng Ji. Creativitybench: Evaluating agent creative reasoning via affordance-based tool repurposing. *arXiv preprint arXiv:2605.02910*, 2026.

[27] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[28] Robert Shaw and John Bransford. *Perceiving, acting and knowing: Toward an ecological psychology*. Routledge, 2017.

[29] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *arXiv preprint arXiv:2409.04109*, 2024.

[30] Robert J Sternberg. *The triarchic theory of intelligence.* The Guilford Press, 1997.

[31] Yufei Tian, Abhilasha Ravichander, Lianhui Qin, Ronan Le Bras, Raja Marjieh, Nanyun Peng, Yejin Choi, Thomas L Griffiths, and Faeze Brahman. Macgyver: Are large language models creative problem solvers? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5303–5324, 2024.

[32] Fei Wang, Wenxuan Zhou, James Y Huang, Nan Xu, Sheng Zhang, Hoifung Poon, and Muhao Chen. mdpo: Conditional preference optimization for multimodal large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8078–8088, 2024.

[33] Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. Scimon: Scientific inspiration machines optimized for novelty. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 279–299, 2024.

[34] Yi Wang, Jiafei Duan, Dieter Fox, and Siddhartha Srinivasa. Newton: Are large language models capable of physical reasoning? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9743–9758, 2023.

[35] Zi-Han Wang, Lam Nguyen, Zhengyang Zhao, Mengyue Yang, Chengwei Qin, Yujiu Yang, and Linyi Yang. Creativebench: Benchmarking and enhancing machine creativity via self-evolving challenges. *arXiv preprint arXiv:2603.11863*, 2026.

[36] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf`.

[37] Xiyang Wu, Tianrui Guan, Dianqi Li, Shuaiyi Huang, Xiaoyu Liu, Xijun Wang, Ruiqi Xian, Abhinav Shrivastava, Furong Huang, Jordan Lee Boyd-Graber, et al. Autohallusion: Automatic generation of hallucination benchmarks for vision-language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8395–8419, 2024.

[38] Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and Erik Cambria. Large language models for automated open-domain scientific hypotheses discovery. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13545–13565, 2024.

[39] Chunlin Yu, Hanqing Wang, Ye Shi, Haoyang Luo, Sibei Yang, Jingyi Yu, and Jingya Wang. Seqafford: Sequential 3d affordance reasoning via multimodal large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1691–1701, 2025.

[40] Yunan Zeng, Yan Huang, Jinjin Zhang, Zequn Jie, Zhenhua Chai, and Liang Wang. Investigating compositional challenges in vision-language models for visual grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14141–14151, 2024.

Appendix

A Significance, Scope, and Clarifications

A.1 Why MM-CreativityBench Matters

A Missing Dimension in Multimodal Evaluation. Recent progress in large multimodal models has been evaluated primarily through recognition, visual question answering, spatial reasoning, and instruction following. These settings are important, but they do not fully test whether a model can use perception as evidence for discovering non-obvious but physically feasible solutions. MM-CreativityBench focuses on this under-measured capability: **visually grounded creative tool repurposing**. The central question is not whether a model can describe a scene or generate a plausible solution in language, but whether it can inspect the environment, identify the relevant object and part, and justify how observable physical properties support an unconventional use. This makes the benchmark a focused test of creative intelligence under perceptual and physical constraints.

From Static Image Understanding to Evidence-Driven Exploration. A key contribution of MM-CreativityBench is that it moves beyond static image-to-answer evaluation. In real-world problem solving, an agent often does not know in advance which object or region matters. It must search, inspect, compare, and revise its hypothesis before committing to a solution. Our interactive protocol captures this process by allowing models to inspect the scene, candidate entities, and zoomed-in parts. This design makes it possible to evaluate not only the final answer, but also whether the answer emerges from **grounded exploration** rather than unsupported guessing. In this sense, the benchmark evaluates creative tool use as a process, not merely as a final textual output.

Part-Level Grounding as the Core Challenge. MM-CreativityBench is designed to expose a specific weakness in current LMMs: models often identify a generally relevant object but fail to determine which part, attribute, or visual cue actually enables the intended use. This distinction is important because creative repurposing depends on mechanism-level reasoning. A key's usefulness for opening a taped box, for example, does not come from the object category "key" alone, but from a visually grounded property such as a thin, rigid, serrated edge. By requiring models to ground answers at the entity–part level, MM-CreativityBench separates coarse semantic plausibility from genuine physical affordance reasoning.

A Controlled Testbed for Grounded Creative Reasoning. The benchmark is built on a structured affordance knowledge base that links entities, parts, attributes, and affordances. This structure gives the benchmark several advantages. First, it supports systematic task construction while preserving interpretable solution paths. Second, it enables controlled multimodal augmentation at multiple levels of granularity, including scene images, entity views, and part-level close-ups. Third, it makes failure analysis more diagnostic: when a model fails, we can ask whether it overlooked the correct entity, inspected the wrong part, misread visual evidence, hallucinated an attribute, or failed to connect an observed attribute to an affordance. This level of diagnosis is difficult in fully open-ended creativity benchmarks.

Connecting Evaluation with Model Improvement. MM-CreativityBench is not only an evaluation benchmark. It also provides a framework for studying how grounded creative behavior can be improved. Our affordance-aware alignment results show that models can learn more effective exploration and more reliable attribute–affordance reasoning when trained with structured positive trajectories and contrastive negative trajectories. The goal is not to claim that SFT or DPO is a new optimization algorithm, but to show that **affordance-grounded supervision** is a useful training signal for multimodal creative problem solving. This connection between benchmark design, failure analysis, and targeted alignment makes the benchmark valuable as a research tool rather than only as a leaderboard.

Implications for Future Multimodal Agents. The broader significance of MM-CreativityBench lies in its relevance to adaptive agents. Agents operating in homes, labs, workshops, or other resource-limited environments will often need to repurpose available objects rather than rely on predefined tools. Such behavior requires visual grounding, physical commonsense, comparative search, and

flexible recombination of affordances. MM-CreativityBench isolates this ability in a controlled and reproducible form. It therefore offers a concrete step toward evaluating and improving multimodal systems that can solve unfamiliar problems using evidence from their surroundings.

A.2 Clarifications of Concerns

MM-CreativityBench Targets Constrained Creativity, Not All Creativity. Creativity is broad, and we do not claim that creative intelligence can be fully captured by a single benchmark. Our focus is deliberately narrower: **constrained, visually grounded tool repurposing**. In this setting, a solution must be novel relative to canonical object use, but also physically feasible and supported by visual evidence. This operational definition is useful precisely because it avoids the ambiguity of fully open-ended creativity evaluation. Rather than asking whether a model is creative in the abstract, MM-CreativityBench asks whether it can discover a usable object–part solution under explicit perceptual and functional constraints.

Why This Is More Than Physical Commonsense Retrieval. Physical commonsense is a necessary ingredient, but the benchmark requires more than retrieving a familiar object-use association. Each task requires the model to connect a goal with a specific object, a specific part, observable attributes, and a feasible mechanism of use. The model must also make this connection through an interactive visual process. This is why performance drops substantially when moving from entity-level correctness to gold entity–part correctness: models often know which object category is plausible, but fail to ground the precise part and attribute that make the solution work. The benchmark therefore targets **mechanism-sensitive visual repurposing**, not simple commonsense recall.

Synthetic Images as Controlled Visual Grounding, Not a Claim of Full Realism. The use of generated images is a methodological choice for controlled evaluation. Real images would introduce substantial noise in object availability, occlusion, scale, lighting, and part visibility, making it difficult to isolate the reasoning problem. In contrast, generated scenes allow us to construct environments where candidate entities are present, parts can be inspected, and the underlying affordance structure is known. We do not claim that generated images replace real-world embodied evaluation. Rather, they provide a reproducible intermediate setting that tests whether models can use visual evidence when the relevant physical cues are available. This makes MM-CreativityBench a diagnostic benchmark for visual-affordance reasoning, complementary to future evaluations in real physical environments.

Image Generation Does Not Turn the Task into Visual Leakage. A possible concern is that generated images may make the gold answer visually obvious. Our intention is the opposite: image generation is used to make relevant physical evidence inspectable, not to mark the solution. The model still needs to search among multiple entities, inspect candidate parts, and infer which observed attributes support the target affordance. The scene is not allowed to depict task execution, and the interaction protocol requires the model to justify the final answer through object-specific evidence. More generally, MM-CreativityBench evaluates whether a model can transform visible properties into functional hypotheses; making those properties visible is necessary for testing grounding rather than language-only guessing.

Single-Gold Evaluation as Measurement Control. Creative tool use is naturally open-ended, and multiple valid solutions may exist in the real world. However, benchmark evaluation requires a controlled target so that models can be compared consistently. The gold answer in MM-CreativityBench should therefore be understood as a verified solution path, not as a claim that no other solution could ever work. The strict gold metric is intentionally conservative: it measures whether the model recovers the intended object–part mechanism under the constructed scene and constraints. Entity-level accuracy, exploration statistics, and grounding metrics complement this strict score by showing where the reasoning process succeeds or fails. Thus, the single-gold structure supports measurement clarity while preserving the broader view that creativity can admit multiple solutions.

The Interactive Protocol Is Artificial by Design, but Diagnostic. The entity and part inspection interface abstracts away low-level perception problems such as segmentation and camera control. This may appear less realistic than open-world embodied interaction, but it is useful for isolating the central question of the paper: can a model conduct evidence-driven creative reasoning once the environment is inspectable? By structuring interaction into scene, entity, and part views, the

Benchmark	Creative Tool Use	Affordance Grounding	Attribute Grounding	Part-Level Reasoning	Fine-Grained Creativity Levels	Distractors Included	Visual Grounding	Evaluation Protocol
PROST[2]	✗	✓	✓	✗	✗	✓	✗	Static
NEWTON[34]	✗	✗	✓	✗	✗	✓	✗	Static
Creation-MMBench[31]	✗	✗	✗	✗	✗	✗	✓	Static
VillagerBench[11]	✗	✓	✗	✗	✗	✗	✓	Interactive
VisEscape[17]	✗	✓	✗	✗	✗	✓	✓	Interactive
PIQA[3]	✓	✓	✓	✗	✗	✓	✗	Static
MacGyver[31]	✓	✓	✓	✗	✗	✗	✓	Static
EscapeBench[24]	✓	✓	✗	✗	✓	✗	✓	Interactive
CreativityBench[26]	✓	✓	✓	✓	✓	✓	✗	Static / Interactive
***MM-CreativityBench* (Ours)**	✓	✓	✓	✓	✓	✓	✓	Interactive

Table 4: For each existing benchmark, the table indicates whether the corresponding dimension is fully addressed (✓), partially addressed (✓), or not addressed (✗).

benchmark makes exploration behavior observable and comparable across models. This design reduces confounds from visual localization failures and allows us to measure whether models actually inspect the evidence needed for their final answers. The protocol should be viewed as a controlled diagnostic environment, not as a complete simulation of embodied deployment.

Affordance-Knowledge Supervision Is a Research Signal, Not Test-Time Privilege. The affordance knowledge base is used to construct tasks and training trajectories, but the evaluation protocol does not give models access to gold affordance labels or hidden solution paths. At test time, models receive the visible scene, entity names, revealed part names, inspected images, and limited attribute-level disambiguation. The purpose of knowledge-base-derived supervision is to teach models useful intermediate behavior: how to search, how to compare candidate parts, and how to reject visually unsupported affordances. This is analogous to using structured annotations to train better visual reasoning models. The important question is not whether the supervision contains knowledge, but whether that knowledge improves unguided visual-affordance reasoning on disjoint test instances.

SFT and DPO Are Not Claimed as Algorithmic Novelty. Our contribution is not a new fine-tuning objective. SFT and DPO are used as established tools to test a specific hypothesis: grounded creative tool use can be improved by aligning models toward visually supported attribute–affordance reasoning and away from plausible but ungrounded alternatives. The novelty lies in the problem formulation, the interactive visual benchmark, the construction of affordance-grounded trajectories, and the use of hard negative trajectories that target hallucinated or misleading affordance reasoning. In this sense, the training experiments serve as evidence that the benchmark identifies a learnable failure mode, rather than merely reporting that current models perform poorly.

Cross-Model Comparisons Should Be Interpreted Structurally. Absolute scores can be affected by prompting, model family, decoding details, and the interaction interface. Therefore, the main conclusion should not rest on a brittle ranking between two models. The more important result is the recurring structural pattern: models often perform better at coarse entity localization than at fine-grained part grounding; longer exploration does not necessarily produce better answers; visually plausible but unsupported reasoning remains common; and targeted affordance-aware alignment improves both accuracy and efficiency. These patterns are more informative than any single leaderboard position, because they reveal a shared limitation in current LMMs.

Final Take Away. MM-CreativityBench is intentionally controlled, visually grounded, and diagnostic. It does not claim to cover all forms of creativity, nor does it replace real-world embodied evaluation. Its contribution is more precise: it isolates a practically important form of creative problem solving, formalizes it through entity–part–attribute affordance structure, and shows that current LMMs still struggle to connect visual evidence with non-canonical physical use. By combining benchmark construction, interactive evaluation, failure analysis, and affordance-aware alignment, MM-CreativityBench positions creative tool use as a concrete and measurable frontier for multimodal AI.

B Benchmark Comparison Dimensions

Relationship to CreativityBench. CreativityBench[26] supplies the symbolic foundation, including part-level annotations, reverse task construction, entity–part correctness metrics, and a textual

interactive inference analysis. We do not claim these as new. All gold triples in our 333 test and 868 training tasks are drawn from its knowledge base. Our contribution is the newly instantiated multimodal environments, scene/entity/part visual inspection, and multimodal three-branch trajectory alignment. The prior work reveals textual entity descriptions during interaction; our protocol supplies images and limited attribute disambiguation through entity and part inspection.

In Table 4, we compare MM-CreativityBench with prior benchmarks along the dimensions that are central to grounded creative tool use. In the table, a checkmark indicates that the benchmark directly evaluates the dimension, a yellow checkmark indicates that the dimension is only partially covered, and a cross indicates that the dimension is not explicitly addressed.

Creative Tool Use. This column indicates whether a benchmark requires models to repurpose available objects for non-standard goals. A benchmark receives full credit when the task is not simply to recognize an object, describe a scene, or use a tool in its canonical way, but to discover a physically feasible alternative use for an object under task constraints. This dimension reflects our focus on creative tool repurposing: the model must identify a non-obvious mapping between an available object or part and the desired function.

Affordance Grounding. This column measures whether the benchmark requires reasoning about what actions an object can support. In MM-CreativityBench, the relevant solution depends on whether an entity or part affords the target function, such as cutting, scraping, supporting, shielding, pressing, or absorbing. Benchmarks receive partial credit when they involve physical commonsense or action feasibility in a broad sense, but do not require the model to explicitly connect an observed object or part to a task-specific affordance.

Attribute Grounding. This column indicates whether success depends on grounding the solution in concrete physical or state attributes, such as material, shape, rigidity, sharpness, hollowness, texture, accessibility, or whether a part is intact or detachable. This distinction is important because creative tool use is not valid merely because an object sounds plausible; the proposed use must be justified by attributes that are actually present. MM-CreativityBench emphasizes this dimension because many failures arise when models hallucinate attributes or select parts whose geometry or material cannot support the intended function.

Part-Level Reasoning. This column captures whether the benchmark evaluates reasoning over object parts rather than only whole objects. Many creative solutions depend on a specific functional component, such as a serrated edge, rubber pad, lid panel, handle, tip, surface, or comb-like structure. A model may identify the correct entity but still fail if it selects the wrong part. MM-CreativityBench therefore requires models to inspect and choose entity–part pairs, making part-level grounding a core evaluation target rather than an incidental feature.

Fine-Grained Creativity Levels. This column indicates whether the benchmark distinguishes different levels or degrees of creative repurposing. Some object uses are close to ordinary use, while others require more atypical functional transfer. MM-CreativityBench includes this dimension to analyze whether models perform differently when the gold affordance is more typical or more unusual. This allows the benchmark to move beyond binary success and examine how models handle varying degrees of creative abstraction and ambiguity.

Distractors Included. This column indicates whether the benchmark includes misleading alternatives that make the task diagnostic. In MM-CreativityBench, distractors are not random irrelevant objects. Some share similar affordances with the gold solution but lack a decisive attribute, while others are scene-plausible objects that might attract the model through object priors. This design tests whether models can distinguish the correct object and part through fine-grained visual and physical evidence, rather than relying on surface plausibility.

Visual Grounding. This column specifies whether the benchmark requires the model to ground its answer in visual input. Text-only physical reasoning can often be solved using commonsense priors, but real-world creative tool use requires observing the available environment. MM-CreativityBench provides scene-level, entity-level, and part-level visual observations so that models must inspect the environment, verify relevant parts, and base their final answer on visual evidence.

Evaluation Protocol. This column describes whether the benchmark is evaluated in a static or interactive setting. Static benchmarks present all information at once and ask the model to answer directly. Interactive benchmarks require the model to decide what to inspect, update its belief based on new observations, and commit to an answer after evidence gathering. MM-CreativityBench adopts an interactive protocol because grounded creative tool use is a process: the model must explore candidate entities, examine relevant parts, reject misleading alternatives, and justify a physically feasible solution based on the evidence it has observed.

Overall, Table 4 shows that prior benchmarks cover important pieces of physical reasoning, visual grounding, or interactive problem solving, but they do not jointly evaluate all components required for affordance-grounded creative tool repurposing. MM-CreativityBench is designed to close this gap by combining creative object repurposing, affordance and attribute grounding, part-level inspection, controlled distractors, visual evidence, and interactive evaluation in a single benchmark.

C Preliminary Experiment

Our preliminary experiment is designed as a controlled comparison between two prompting strategies on 100 creative tool-use tasks sampled from MacGyver [31]. To evaluate the creativity under a multimodal environment, where the model directly perceives necessary physical attributes from the input image and reasons about creative tool repurposing, we first generate a scenario image using Gemini-2.5-Pro. Then, update the task description to only include the constraints that cannot be represented visually. In the **direct prompt** setting, given an input task and scenario image, the model is asked to propose a feasible solution under the task constraints without any prescribed intermediate reasoning steps, thereby testing its implicit ability to connect task requirements with tool functions. In the **structured affordance-level CoT** setting, the model is instead guided through an explicit reasoning pipeline that includes listing the available tools, decomposing each tool into parts, inferring relevant physical properties, deriving possible affordances, justifying each action step, and validating the final solution against the stated constraints.

We evaluate outputs using six criteria: **Correctness**, **Feasibility**, **Physical Grounding**, **Constraint Coverage**, **Tool Usage**, and **Creativity**, under pairwise relative comparison between the two prompting strategies. Please see the prompts below for more details. We use GPT-4.1-mini as the target model and GPT-5.2 as the judge model, employing temperature 0.0 to guarantee deterministic outputs. All of other settings we follow the original MacGyver paper's protocols.

Scenario Image Generation Prompt

```
Create one realistic scene that shows only the initial setup of this problem before any actions are
 taken. Focus on the environment, relevant objects, and constraints. Do not include text overlays,
labels, or solution steps.

Problem: {task description from MacGyver}
```

Direct Prompt

```
You are an expert at creative physical tool-use reasoning.
Given the task description below and image, produce a feasible solution.

Rules:
1. Use only tools/items explicitly available in the input image.
2. Respect physical constraints if the task description restricts physical attributes or image
shows physical constraints such as size or state.
3. Invent new tools using tools/items explicitly available in the input image if it is needed.
4. Provide practical steps that can actually be executed.
5. Do not involve any unnecessary steps to achieve the task's goal.
6. If a complete solution is impossible, return the best partial plan and explain why it cannot be
completed.

TASK DESCRIPTION:
{problem}

Return JSON:
{{
  "solvable": "Yes or No",
```

```
  "solvable_explanation": "1-3 sentences about why the given task is solvable or not",
  "solution_steps": ["Step 1: ...", "Step 2: ...", ...],
  "final_solution": "One concise paragraph that summarizes the full approach.",
  "used_tools": ["tool 1", "tool 2", ...],
  "constraint_handling": [{{"constraint": "...", "handling": "..."}}, ...]
}}
```

CoT Prompt

```
You are an expert at creative physical tool-use reasoning.
Solve the task by explicitly reasoning over tool parts and affordances under constraints.

Rules:
1. Use only tools/items explicitly available in the input image.
2. Respect physical constraints if the task description restricts physical attributes or image
shows physical constraints such as size or state.
3. Invent new tools using tools/items explicitly available in the input image if it is needed.
4. Provide practical steps that can actually be executed.
5. Do not involve any unnecessary steps to achieve the task's goal.
6. If a complete solution is impossible, return the best partial plan and explain why it cannot be
completed.

Required reasoning procedure:
1. State the task goal and concrete success condition.
2. List all available tools/items from the input image (no additions).
3. For each relevant tool, identify the key part(s), infer physical properties, and derive part-
level affordances useful for this task.
4. Build a step-by-step plan where each step references tool parts and the affordance being used.
5. Validate each step against stated constraints (e.g., broken/unusable items, size mismatch,
blocked function, state limitations).
6. Keep the plan practical and minimal with no unnecessary actions.

TASK DESCRIPTION:
{problem}

Return JSON:
{{
  "task_goal": "...",
  "success_condition": "...",
  "identified_constraints": ["...", "..."],
  "available_tools": [
    {{
      "tool": "...",
      "relevant_parts": [
        {{
          "part": "... or NA",
          "inferred_physical_properties": ["...", "..."],
          "affordances_for_task": ["...", "...", ...],
          "usable_under_constraints": "Yes or No",
        }},
        ...
      ]
    }},
    ...
  ],
  "reasoning_plan": [
    {{
      "step": 1,
      "action": "...",
      "tool_parts_used": ["tool:part", "..."],
      "affordance_used": ["...", "..."],
      "explanation": "1 sentence about why it works"
    }},
    {{
      "step": 2,
      "action": "...",
      "tool_parts_used": ["tool:part", "..."],
      "affordance_used": ["...", "..."],
      "explanation": "1 sentence about why it works"
    }},
    ...
  ],
  "solvable": "Yes or No",
  "solvable_explanation": "1-3 sentences about why the given task is solvable or not",
  "solution_steps": ["Step 1: ...", "Step 2: ...", ...],
  "final_solution": "One concise paragraph that summarizes the full approach.",
```

```
  "used_tools": ["tool 1", "tool 2", ...],
  "constraint_handling": [{{"constraint": "...", "handling": "..."}}, ...]
  "creative_reasoning_summary": "1-3 sentences about novelty and practicality."
}}
```

Relative Evaluation

```
You are an expert evaluator for creative physical tool-use reasoning quality.
Your task is to compare TWO candidate solutions for the SAME task, using the ground-truth solution
as reference.

IMPORTANT:
- You are NOT checking wording similarity. Judge functional quality and practical validity.
- A solution can be strong even if it differs from the ground-truth approach.
- Use the following criteria:
    - Correctness: Whether the plan actually solves the task objective.
    - Feasibility: Whether the plan can be physically executed under stated constraints.
    - Physical Grounding: Whether it uses realistic object properties/mechanics correctly.
    - Constraint Coverage: Whether it handles all explicit constraints.
    - Tool Usage: Whether it uses only available tools appropriately and purposefully.
    - Creative Reasoning: Novel, non-obvious but valid repurposing/combination of tools.
    - Overall: Holistic quality across all above dimensions.

For EACH criterion:
1. Determine winner:
   - "win" if solution1_score > solution2_score
   - "lose" if solution2_score > solution1_score
   - "tie" if both are equal
2. Give a short rationale (1-2 sentences).

TASK DESCRIPTION:
{problem}

GROUND-TRUTH SOLUTION:
{ground_truth_solution}

CANDIDATE SOLUTION1 (DEFAULT PROMPT OUTPUT):
{solution1}

CANDIDATE SOLUTION2 (COT PROMPT OUTPUT):
{solution2}

Return STRICT JSON:
{{
  "correctness": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "feasibility": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "physical_grounding": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "constraint_coverage": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "tool_usage": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "creativity": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "overall": {{
    "winner": "win or lose or tie",
    "rationale": "1-2 sentences."
  }},
  "short_summary": "2-4 sentences summarizing key tradeoffs between default and CoT."
}}
```

D Benchmark Construction Details

D.1 Affordance Knowledge Base Basis

Our benchmark construction is grounded in an existing open-source affordance knowledge base of physical entities, object parts, and part-level affordances.[1] The knowledge base organizes everyday objects into a structured partonomy: each entity is decomposed into functional parts, and each part is annotated with physical attributes, state attributes, and possible functional affordances. Physical attributes describe relatively stable properties, such as shape, material, rigidity, sharpness, hollowness, flexibility, and surface texture, while state attributes describe situational conditions, such as whether a part is open, clean, intact, accessible, or detachable. The affordance annotations specify what functional roles a part can support, together with use conditions, recipient conditions, examples, and suitability levels. These annotations provide the symbolic basis for MM-CreativityBench: they allow us to identify which part of which object can support a target affordance, what attributes justify this use, and what conditions must hold for the use to be valid. Thus, benchmark instances are not created through unconstrained scenario writing; they are derived from explicit part-level attribute–affordance relations that can be inspected, verified, and converted into multimodal grounding problems.

D.2 Reverse Task Construction

We construct MM-CreativityBench tasks in a reverse direction. Instead of first writing an open-ended scenario and then labeling a possible answer, we begin with a verified entity–part–affordance relation from the knowledge base and generate a scenario around it. For each task, we first sample a target entity e^*, a target part $p^* \in P(e^*)$, and a gold affordance f^* supported by the annotated attributes $A(p^*)$. This defines the gold solution

$$g = (e^*, p^*, f^*).$$

Here, e^* specifies the object to be repurposed, p^* specifies the decisive part, and f^* specifies the functional role that the part can play in solving the task.

Given the gold solution, we use GPT-5.4 for reverse task proposal generation. Specifically, GPT-5.4 is given (e^*, p^*, f^*) together with the supporting physical and state attributes of p^*, and is prompted to propose a practical task description x that requires the affordance f^* without explicitly mentioning the target entity, the target part, or any surface cue that would make the answer trivial. The generated description must satisfy three requirements: it should describe a realistic everyday problem, include only constraints relevant to the intended affordance, and leave the solver to infer which available object and part can satisfy the goal. GPT-5.4 is used only to generate candidate task descriptions; all accepted tasks are subsequently checked and refined by human annotators.

After obtaining a candidate task description, we construct the candidate entity set by adding distractors to the gold entity. For each gold solution, we sample a distractor set $E^- = \{e_1, \ldots, e_{N-1}\}$ and form

$$E = \{e^*\} \cup E^-.$$

Distractors are selected from the same knowledge base to create controlled ambiguity. We include *affordance-similar distractors*, whose parts appear functionally related to f^* but fail under closer inspection because they lack a necessary physical attribute, have an incompatible state, provide a weaker mechanism, or violate a contextual requirement. We also include *scene-plausible distractors*, which naturally co-occur with the gold entity in the same environment but do not support the target affordance. This design prevents the task from being solved by object priors or generic tool-use associations alone. A model must inspect candidate entities, compare their parts, and identify the part whose attributes best support the required affordance.

Each symbolic task is represented as

$$T = (x, E, g), \qquad g = (e^*, p^*, f^*),$$

where x is the task description, E is the candidate entity set, and g is the gold entity–part–affordance solution. This formulation makes the benchmark an inverse grounding problem: the task description specifies a need, the scene provides multiple possible objects, and the model must recover the correct entity and part by grounding the required affordance in visual and physical evidence.

[1] Repository at `https://github.com/CreativityBench/CreativityBench`

We apply a multi-stage quality-control process before including a task in the benchmark. First, **gold validity**: the gold part must physically support the target affordance under the stated task constraints, and the supporting evidence must be present in its knowledge-base attributes. Second, **distractor separability**: no distractor can serve as an equally valid solution; each distractor must fail for a specific and identifiable reason, such as a missing attribute, incompatible state, weaker functional mechanism, safety concern, or contextual mismatch. Third, **scenario clarity**: the task description must be natural, concise, and unambiguous, while avoiding direct lexical leakage of the gold entity or part. Fourth, **scene coherence**: all candidate entities must plausibly co-occur in a single realistic environment without making the scene artificial or visually cluttered. Fifth, **visual observability**: the decisive part and the attributes required to justify the solution must be inspectable in the generated entity- or part-level images. These criteria remove ambiguous cases, physically invalid solutions, non-visual tasks, and scenarios with unintended alternative answers.

Finally, all tasks are manually verified and refined by human annotators. Annotators check whether the generated task genuinely requires the intended affordance, whether the gold entity–part pair is uniquely valid among the candidates, whether the distractors are plausible but separable, and whether the scenario can be faithfully visualized. When needed, annotators revise the task wording, replace distractors, or discard the example entirely. Using this pipeline, we construct 333 held-out tasks for MM-CreativityBench evaluation and 868 disjoint training tasks for trajectory sampling in the alignment stage. The two splits are separated at the task and visual-instance level to eliminate leakage between training trajectories and benchmark evaluation.

Construction pass rates. Approximately 28% of reverse-generated candidate tasks pass initial verification. Distractor addition itself incurs no further task loss, but the subsequent multi-stage quality-control process has an approximate 57% pass rate. Image generation and human verification retain approximately 91% of cases entering that stage. These are conditional stage pass rates, rather than absolute candidate counts. The 57% rate aggregates quality criteria, including distractor separability; it does not separately quantify the rejection of alternative valid solutions or human revise/discard rates.

D.3 Multimodal Image Construction

After constructing each symbolic task $T = (x, E, g)$, we convert it into an interactive multimodal instance by generating images at three levels: entity, part, and environment. All images are generated with Gemini-3.1-Image-Pro. The goal is not only to visualize the task, but also to create a controlled evidence hierarchy that matches the benchmark protocol: the model first observes the full environment, then chooses candidate entities to inspect, and finally verifies part-level evidence before answering.

Entity-level images. For each candidate entity $e \in E$, we generate a full-object reference image

$$I_e = \pi_{\text{ent}}(e, P(e), \{A(p) : p \in P(e)\}),$$

where the prompt is conditioned on the entity name, its part decomposition, and concise summaries of part-level attributes. The generated image should make the entity recognizable as a whole while preserving visually relevant cues such as geometry, material, surface texture, openings, edges, handles, tips, flexible regions, or contact surfaces.

Image Generation Prompt (Entity)

```
Generate a single-object reference image on a pure white seamless studio background.

The image must focus only on the target entity, centered and fully visible unless the annotation
says a part is hidden or internal. Do not place the object in a room, scenario, or narrative scene.
 Do not add extra objects, hands, floor props, or decorative context. Render the object
photorealistically with neutral studio lighting and sharp material detail. Preserve visible state
cues such as wetness, residue, fullness, deformation, wear, or occlusion when visually
representable.

Target entity: {ENTITY_NAME}
Entity summary: {ENTITY_SUMMARY}

Part-by-part guidance:
{PART_OVERVIEW}
```

```
Output goal: one clean white-background product-style reference image of this exact entity.
Annotate each part's exact name beside the corresponding part. Annotate only the name, with no
description.
```

Part-level images. For each part $p \in P(e)$, we generate a zoomed-in part image

$$I_{e,p} = \pi_{\text{part}}(e, p, A(p), I_e),$$

using the entity image I_e as a visual anchor. This ensures that the part view remains consistent with the full-object image in geometry, color, material, and relative structure. The prompt focuses tightly on the target part and asks the generator to preserve the attributes relevant to its possible use. This level is necessary because many creative solutions depend on local evidence, such as a rubber pad, serrated edge, hollow cavity, flat panel, hook-like curve, narrow tip, or absorbent surface, which may be hard to verify from the environment image alone.

Image Generation Prompt (Part)

```
Generate a close-up studio image on a pure white seamless background.

{REFERENCE_IMAGE_INSTRUCTION}

Focus on the target part and make its local visual cues easy to inspect. Do not add room context,
props, labels, arrows, or diagrams. Use product-photography lighting with strong texture visibility
 and clean edges. Show fine material detail, residue, moisture, wear, occlusion, seams, openings,
fill cues, deformation, and surface texture whenever they are visually representable.

Entity: {ENTITY_NAME}
Entity summary: {ENTITY_SUMMARY}
Target part: {TARGET_PART_NAME}

{PHYSICAL_ATTRIBUTES_BLOCK}
{STATE_ATTRIBUTES_BLOCK}

Physical summary: {PART_PHYSICAL_SUMMARY}
State summary: {PART_STATE_SUMMARY}

Important: observe the given entity image. The generated target part image should not deviate
substantially from the entity image, but should be comprehensive and informative about detailed
local visual cues.

Output goal: one white-background close-up image centered on the target part. The part should have
exactly the same physical and state attributes as described above.
```

Environment-level images. We then generate the full scenario image

$$I_{\text{env}} = \pi_{\text{env}}(x, E, \{I_e : e \in E\}),$$

conditioned on the task description, the candidate entity list, and the generated entity reference images. The prompt requires all candidate entities to appear naturally in a coherent scene with realistic scale, placement, and lighting. It also explicitly prohibits showing the task already being solved or introducing extra objects that could serve as unintended alternative solutions. Thus, the environment image defines the search space, while entity and part images provide progressively finer evidence for verification.

System Prompt

```
Generate one photorealistic overall environment scene image.

Scenario: {SCENARIO}
Task goal: {TASK_GOAL}

The attached reference images are the required entities and are provided in the same order as the
list below. Preserve each referenced entity's identity, materials, proportions, and recognizable
shape while placing them naturally in the scene.

Required entities to include in the scene:
{REQUIRED_ENTITY_LIST}

Required interactable items to include naturally in the scene:
```

```
{INTERACTABLE_ITEM_LIST}

Scene requirements:
- Show a single coherent real-world scene, not a collage or white-background layout.
- Include every required entity and every required interactable item in a plausible manner.
- Naturally embed the objects into the scenario with realistic scale, lighting, and placement, and
make sure they are clearly visible.
- Do not add text labels, arrows, diagrams, split panels, or product-shot framing.
- Do not show how to solve the task or explicitly depict task execution.
- Minimize extraneous objects so the scenario remains focused and uncluttered.

Output goal: one natural overall scenario image that clearly contains all required entities and
interactable items.
```

Image quality and consistency checks. We check generated images for three requirements before using them in the benchmark. First, all candidate entities must be present and recognizable in the environment. Second, entity and part images must be visually consistent, so that a part inspection can be interpreted as a closer view of the same object. Third, the decisive part must be visually inspectable rather than hidden, cropped away, or rendered in a way that makes the task impossible. Images that fail these requirements are regenerated or manually filtered.

Textual disambiguation for visual feedback. Although all generated images are used during evaluation, some fine-grained attributes may not be decisively inferable from the image alone due to rendering ambiguity, viewpoint, lighting, or material appearance. To avoid making the task depend on accidental image artifacts, we add a textual disambiguation step. Given an entity–part image pair $(I_e, I_{e,p})$ and an attribute $\alpha \in A(p)$, we use GPT-5.4 to judge whether the visual evidence alone is sufficient to support the attribute:

$$\ell(\alpha) \in \{\textsc{VisualEnough}, \textsc{TextNeeded}\}.$$

Visibility/accessibility is always supplied as text upon part inspection because a rendered viewpoint can misrepresent the task's access constraints; these annotations are treated as TEXTNEEDED regardless of visual clarity. VISUALENOUGH means that the attribute can be reasonably inferred from the image without additional text. TEXTNEEDED means that the attribute is part of the knowledge-base annotation and is compatible with the generated image, but the visual evidence may be ambiguous; in this case, we provide a short textual clarification with the image when the corresponding part is inspected.

This step is used only to disambiguate low-level visual details. The accompanying text is restricted to object or part attributes, such as material, state, surface property, rigidity, hollowness, or accessibility. It does not reveal the target affordance, the correct entity, the correct part, or the final solution. The same procedure is applied to all candidate entities and parts, not only to the gold solution. Thus, the benchmark still requires models to inspect the visual evidence and reason over candidate parts, while preventing failures caused by attributes that are intended in the generated image but not visually decisive. We denote the attributes requiring textual clarification as

$$\delta(p) = \{\alpha \in A(p) : \ell(\alpha) = \textsc{TextNeeded}\},$$

and include concise descriptions of $\delta(p)$ when presenting the corresponding image as interaction feedback.

Text coverage and visual-input controls. Excluding visibility/accessibility, approximately 35% of parts have auxiliary text, covering less than 7% of annotated physical/state attributes. These annotation proportions describe the information supplied, rather than establishing how much each channel contributes to decisions. The feedback template in Section G includes physical/state summary fields, but they do not contain the complete attribute dictionaries in the standard setting. Text is revealed only after the corresponding part is inspected. In contrast, the No Image condition in Figure 5 converts all physical/state attributes into textual descriptions; it is therefore not an image-removal ablation with identical textual feedback.

As an observational check, Table 5 reports trajectories that received no auxiliary attribute text beyond visibility/accessibility. These are model-dependent subsets of the 333 tasks, not a separately controlled images-only condition. Selection depends on the parts inspected and can favor shorter or less informative exploration, so these results do not isolate the causal contribution of vision.

Model	Selected trajectories	Gold correct	Entity correct
GPT-5.4	71	0.155	0.408
Qwen3-VL-32B	56	0.214	0.482

Table 5: Correctness on selected trajectories receiving only visibility/accessibility text alongside images. Rates use each model's selected subset as the denominator.

E Training Trajectory Construction Details

We provide additional details for the trajectory construction procedure introduced in the main text. The goal is to construct supervision not only for the final entity–part answer, but also for the intermediate evidence-seeking process: selecting entities to inspect, verifying candidate parts, judging physical attributes, and rejecting plausible but physically invalid alternatives.

E.1 Trajectory Formulation

Each task is represented as $\mathcal{T} = (x, I_{\text{env}}, E, g)$, where x is the task instruction, I_{env} is the environment image, E is the set of scene entities, and $g = (e^*, p^*, f^*)$ is the gold solution consisting of the target entity, target part, and target affordance. Each entity $e \in E$ has an annotated part set $P(e)$.

A trajectory is a sequence of interaction turns:

$$\tau = \{(o_t, r_t)\}_{t=1}^{T}, \qquad o_t = (u_t, I_t), \qquad r_t = (z_t, a_t).$$

Here, u_t is the textual feedback, I_t is the visual observation, z_t is the model's free-form reasoning, and a_t is a structured action. The action space contains three operations:

$$a_t \in \{\texttt{inspect_entity}(e),\ \texttt{inspect_part}(e, p),\ \texttt{answer}(e, p, h)\},$$

where $e \in E$, $p \in P(e)$, and h describes how the selected part should be used.

The actions define the interaction protocol. The action $\texttt{inspect_entity}(e)$ returns the entity-level image I_e and the part list $P(e)$. The action $\texttt{inspect_part}(e, p)$ returns $I_{e,p}$ with visibility/accessibility text and other attribute disambiguation where needed (Section D.3). The action $\texttt{answer}(e, p, h)$ terminates the trajectory and provides the final grounded solution. Thus, the observation at each turn is determined by the previous action:

$$I_t = \begin{cases} I_{\text{env}}, & t = 1, \\ I_e, & a_{t-1} = \texttt{inspect_entity}(e), \\ I_{e,p}, & a_{t-1} = \texttt{inspect_part}(e, p). \end{cases}$$

This formulation ensures that each reasoning step is aligned with the appropriate level of visual evidence: scene, entity, or part.

E.2 Knowledge-Guided Exploration Stack

To construct positive trajectories, we maintain an ordered exploration stack $\mathcal{S}_t$. Each stack element is either an entity item $(\texttt{entity}, e)$ or a part item $(\texttt{part}, (e, p))$. The top element determines the next inspection target.

The stack is guided by an affordance-relevance function:

$$J : E \times P(e) \to \{0, 1\},$$

where $J(e, p) = 1$ indicates that part p of entity e has an affordance similar or relevant to the target affordance f^* according to the knowledge base $\mathcal{K}$. This does not necessarily mean that (e, p) is the gold answer; it only means that the part is worth inspecting. This distinction is important because many distractors are affordance-similar but fail under fine-grained physical verification.

At the first turn, the model observes the scene and proposes an ordered list of candidate entities. This list initializes $\mathcal{S}_1$, prioritizing likely entities while allowing systematic exploration. The stack is then updated as follows.

- **Entity inspection.** When the top element is $(\texttt{entity}, e_i)$, the positive branch takes $\texttt{inspect_entity}(e_i)$. The entity is removed from the stack, and all affordance-relevant parts $\{p \in P(e_i) : J(e_i, p) = 1\}$ are pushed onto the stack for part-level verification. If no such part exists, exploration moves to the next entity.
- **Part inspection.** When the top element is $(\texttt{part}, (e_i, p_{i,j}))$, the positive branch takes $\texttt{inspect_part}(e_i, p_{i,j})$. The part is then removed from the stack and assigned a binary judgment $b_t \in \{0, 1\}$, indicating whether its observed attributes satisfy the task requirements.
- **Termination.** Exploration terminates when $\mathcal{S}_t = \emptyset$. The model then compares inspected candidate parts, especially those with $b_t = 1$, and produces the final action $\texttt{answer}(e^*, p^*, h^*)$.

This mechanism yields a coarse-to-fine positive trajectory. The model first searches over entities, then verifies affordance-relevant parts, and finally selects the gold pair based on fine-grained physical evidence rather than object-level plausibility alone.

E.3 Three-Branch Trajectory Sampling

The stack specifies what the positive trajectory should inspect, but we still need to generate the reasoning text associated with each step. To obtain both supervised and preference-learning data, we sample three aligned branches at each shared interaction context c_t:

$$r_t^b = (z_t^b, a_t^b), \qquad b \in \{+, -, --\}.$$

The positive branch is the preferred grounded response, while the negative and hard-negative branches are rejected alternatives.

We use GPT-5.4 as the teacher model to help generate the branch-specific reasoning and responses. For the positive branch, GPT-5.4 does not freely decide the exploration structure. Instead, the inspected target, part-level judgment, and final answer are determined by the knowledge base $\mathcal{K}$, the exploration stack $\mathcal{S}_t$, and the gold solution g. GPT-5.4 is used to express this predetermined structure in natural, coherent, and visually grounded language. For the negative and hard-negative branches, GPT-5.4 is prompted with different guidance signals to produce rejected responses at the same state.

Formally, each branch is generated with a branch-specific system prompt s^b and guidance function G^b:

$$r_t^b = \pi_{\text{GPT-5.4}}\big(s^b, c_t, G^b(t, \mathcal{K}, g, \mathcal{S}_t)\big), \qquad b \in \{+, -, --\}.$$

The three branches share the same response format but differ in the information exposed to the teacher model.

Positive branch. The positive branch receives structured guidance from $\mathcal{K}$, including relevant attributes, affordance judgments, and the gold solution when needed. At the scene level, the guidance provides the target affordance f^* and the physical attributes needed to support it. At the entity level, it provides the affordance-relevant parts of the inspected entity. At the part level, it provides attribute-level evidence used to determine whether the part satisfies the task constraints. At the final step, it guides the model to select (e^*, p^*) and explain how the selected part should be used.

The positive response must satisfy three criteria: it should be visually grounded in the current observation, consistent with the exploration stack, and explicit about the attribute–affordance relationship. The resulting positive trajectory is used for supervised fine-tuning.

Prompt for Positive Guidance

```
[SYSTEM PROMPT]

You are a creative physical problem-solver to output data trajectories that will be collected for
open-source model training. Given a task and scenario, solve it by repurposing a part of an entity
using attribute-grounded, visually grounded reasoning. You may inspect only one entity or one part
per turn before answering. Always end with a single JSON object whenever the prompt asks for one.

Your reasoning style is guided by the user prompt. Each turn should read like a natural caption of
your thought process: coherent, grounded, and observant. Even when the prompt quietly guides your
reasoning, behave as if you are figuring things out normally from the task and feedback. Never
mention hidden guidance, gold data, or prompt references.


[INITIAL PROMPT]
```

```
## Task Basis
You are currently in <SCENARIO_OR_ENVIRONMENT>.
The task that the user requires you to do is:
<TASK>

The entity names available in the scene are:
- <ENTITY_NAME_1>
- <ENTITY_NAME_2>
- ...

You should reason first, then end with a JSON in the format {"reasoning":"...","action":"
inspect_entity","entity":"<exact entity name>","top_candidates":["<exact entity name>", "..."]}.

## Guidance on Reasoning
1. Start naturally by thinking about what kind of affordance the task needs.
Target affordance to naturally reason toward: <TARGET_AFFORDANCE>
2. Continue by thinking about the core attributes that would enable that affordance. Express these
as if you are inferring them yourself.
- <ATTRIBUTE_1>
- <ATTRIBUTE_2>
- ...
3. Then naturally transition your reasoning into inspecting the environment.
4. Go through all listed entities with no overlap and nothing left behind.
5. For each entity, say where it is in the image if visible, then give a brief grounded description
 focused on visible physical and state cues.
6. After covering all entities, name up to three candidate entities, explain briefly why they look
promising, and show your intention to inspect those top candidates first and then continue through
every other entity as well.
7. In the final JSON, keep the reasoning summary brief, choose one exact entity name to inspect now,
 and include a top_candidates list for internal tracking only.

## Additional Notes
1. Follow the structure above, but phrase everything in your own natural words.
2. Behave as if you are reasoning normally from the task and image; never mention hidden guidance,
gold data, or prompt references.
3. Output the full reasoning before the final JSON.
4. Copy entity names exactly from the provided list.
5. The top_candidates field is required in the JSON for this round.


[ENTITY FEEDBACK PROMPT]

## Feedback Basis
ENTITY INSPECTION: <ENTITY_NAME>
This entity includes these exact part names: <PART_NAME_1>, <PART_NAME_2>, ...

## Guidance on Reasoning
Please perform visual grounding for each part first and explain in your own words why it may or may
 not achieve the needed affordance.
<ENTITY_BRANCH_INSTRUCTIONS>

Here is the reference for each part:
- <PART_NAME_1>: <CAN_OR_CANNOT_SERVE_VERDICT>. Reason: <REFERENCE_REASON_1>
- <PART_NAME_2>: <CAN_OR_CANNOT_SERVE_VERDICT>. Reason: <REFERENCE_REASON_2>
- ...

## Additional Notes
1. Observe and reason about all parts with no overlap and nothing left behind.
2. Base your reasoning on visible cues and the provided reference, but do not copy the reference
text verbatim.
3. Keep the reasoning natural and coherent, as if you are figuring it out normally.
4. Output the full reasoning before any JSON.
5. Copy part or entity names exactly.

(If this is the last exploration step, use:)
This is the last exploration step.
Reason through every part, explain why none of them finally solves the need well enough, and end by
 explicitly saying this is the last entity you need to explore and you should now move on to the
final answer.
Do not output any JSON in this turn.
Do not append a JSON object at the end.
Stop immediately after the prose reasoning is finished.

(If this entity has similar-affordance parts, use:)
This entity has several parts that may elicit the needed affordance: <SIMILAR_PART_NAMES>.
Reason through every part first, then explicitly say you will inspect the promising parts one by
one.
All parts that can serve the needed affordance must eventually be explored.
```

```
Finally output JSON in the format {"reasoning":"...","action":"inspect_part","part":"<exact part
name>"} and choose one exact part name from the promising parts list.

(If this entity has no similar-affordance parts, use:)
This entity does not have any part that can elicit the needed affordance according to the reference.

Reason through every part first, then conclude that you should continue to the next unexplored
entity.
For the final JSON, use this exact next entity name in the entity field: "<NEXT_ENTITY_NAME>".
Finally output JSON in the format {"reasoning":"...","action":"inspect_entity","entity":"<exact
entity name>"} and copy that exact entity name.


[PART FEEDBACK PROMPT]

## Feedback Basis
PART INSPECTION: <PART_NAME>
Belongs to entity: <ENTITY_NAME>
Physical text explanation: <PHYSICAL_SUMMARY>
State text explanation: <STATE_SUMMARY>
Besides the text above, also carefully inspect the image for additional grounding cues. For
visibility and availability, follow the text explanation even if the image looks clearer.

## Guidance on Reasoning
The detailed physical attributes of this part are:
<LOOKUP_PHYSICAL_ATTRIBUTES>
The detailed state attributes of this part are:
<LOOKUP_STATE_ATTRIBUTES>
<OPTIONAL_GOLD_HINT>
Analyze whether this part can really serve the intended affordance, focusing on visible cues and
practical constraints such as effectiveness, safety, environmental impact, and social acceptability.

For enable_affordance, do not use an overly harsh standard: if the part has a grounded, physically
plausible, or indirect way to help achieve the needed affordance, it is acceptable to mark true.
Reserve false for parts that are clearly a poor match, unsafe, or not realistically useful after a
closer inspection. Stay grounded and do not hallucinate capabilities.
<NEXT_TARGET_LINE>
<PART_BRANCH_INSTRUCTIONS>

## Additional Notes
1. Keep the reasoning grounded; do not copy the reference dictionaries directly.
2. Output the full reasoning before any JSON.
3. The enable_affordance field must be true or false.
4. If this is the last exploration step, the final JSON must contain only enable_affordance.
Otherwise, copy the next part or entity name exactly when you emit JSON.

(If inspecting gold part, insert:)
Note that the part you are inspecting now is the gold part of the gold entity. Do not reveal this,
but you must conclude that it can serve the intended need.

(If next target exists, insert:)
The next thing you should explore after this is <TARGET_TYPE>: <TARGET_NAME>.

(If this is the last exploration step, use:)
This is the last exploration step. Reason carefully about whether the part really works, then end
naturally by indicating that this is the last part you need to explore and you will now move on to
give the final answer.
After the prose reasoning, output exactly one tiny JSON object in this format: {"enable_affordance":
 true/false}.
That final JSON must contain only the enable_affordance field.
Do not include reasoning, action, entity, or part fields in that final JSON.

(If next target is another part, use:)
After reasoning, please naturally show your intention to inspectthe next part of the same entity.
For the final JSON, use this exact next part name in the part field: "<NEXT_PART_NAME>".
{"reasoning":"...","enable_affordance": true/false,"action":"inspect_part","part":"<exact part name
>"}

(If next target is an entity, use:)
After reasoning, please express naturally that you have finished the worthwhile parts here and
should move to inspect the next entity.
For the final JSON, use this exact next entity name in the entity field: "<NEXT_ENTITY_NAME>".
{"reasoning":"...","enable_affordance": true/false,"action":"inspect_entity","entity":"<exact
entity name>"}


[FINAL ANSWER PROMPT]

## Feedback Basis
```

```
Based on all the entities and parts you have inspected, here are all the candidate parts you
previously believed might help solve the task:
1. Part name: <CANDIDATE_PART_NAME_1>, belongs to entity <CANDIDATE_ENTITY_NAME_1>
2. Part name: <CANDIDATE_PART_NAME_2>, belongs to entity <CANDIDATE_ENTITY_NAME_2>
...

Now choose one of them as your final answer, explain how to use it, and return JSON in this format:
{"reasoning":"...","action":"answer","answer_entity":"<exact entity name>","answer_part":"<exact
part name>","answer_how_to_use":"..."}

## Guidance on Reasoning
Gold entity name: <GOLD_ENTITY_NAME>
Gold part name: <GOLD_PART_NAME>

Reason in this order:
1. Naturally list all the promising candidate parts as shown above.
2. Explicitly choose the best one; your final answer must be the provided gold entity and gold part.

3. Compare the gold part only against the other candidate parts listed above, one by one. If no
other candidate parts are listed above, skip comparison naturally.
4. Then explain how to use the gold part in a concrete, comprehensive way.

Keep the reasoning natural and coherent, and do not mention hidden guidance.
Write full prose reasoning first. Then start the final JSON on a new line.
The JSON must appear only once, at the very end.
Inside the JSON, the reasoning field should be only a very brief summary.
Make answer_how_to_use detailed and practically actionable.
When available and not NA, explicitly include prepare_use_condition, prepare_environment_condition,
 prepare_recipient, preparation steps, placement/application steps, and important cautions or
limits.
Write answer_how_to_use as one coherent multi-step paragraph.

Full solution reference:
{
  "prepare_use_condition": "<PREPARE_USE_CONDITION>",
  "prepare_environment_condition": "<PREPARE_ENVIRONMENT_CONDITION>",
  "prepare_recipient": "<PREPARE_RECIPIENT>",
  "apply_affordance": "<APPLY_AFFORDANCE>"
}

In the final JSON:
- keep reasoning brief
- use the exact gold entity and part names
- make answer_how_to_use detailed and practically actionable
- compare only against the candidate parts listed above; if there are no others, no comparison is
needed
- describe how to use the part while catering to use condition, environment condition, and
recipient condition when they are not NA
- mention preparation, placement/application, and important cautions or limits
- start the JSON on a new line after the prose reasoning

(If other candidates exist, append:)
Comparison references for why the other candidate parts are not the gold choice:
- <OTHER_PART_NAME> from <OTHER_ENTITY_NAME>: <GOLD_CHANGE_REASON>
```

Negative branch. The negative branch follows the standard evaluation setting. It receives standard evaluation observations: the task, images, entity and revealed part names, and limited attribute disambiguation. It does not receive hidden affordance labels, gold answers, part-level judgments, or attribute rationales from $\mathcal{K}$. This branch captures realistic inference-time mistakes, such as inspecting irrelevant entities, overlooking decisive parts, or selecting a plausible but suboptimal part.

Unlike the hard-negative branch, the negative branch is not explicitly instructed to be wrong. Its errors arise from the lack of fine-grained affordance guidance. When the positive exploration stack is exhausted, a termination signal is added so that the branch produces a final answer and remains comparable with the positive trajectory.

Hard-negative branch. The hard-negative branch is designed to create stronger contrast for preference learning. It preserves fluent reasoning and valid action format, but is guided toward semantically incorrect or insufficiently grounded conclusions. For example, it may hallucinate unsupported physical attributes, rely on object-level priors, ignore visual evidence, or choose an affordance-similar distractor that lacks the required physical properties.

The hard-negative branch receives structural information such as the task, entity names, part names, and output format, but no grounding signals from $\mathcal{K}$. Its action a_t^{--} is not constrained by the affordance-relevance function J, allowing it to deviate from the positive exploration policy while remaining superficially plausible.

Prompt for Positive Guidance

```
[SYSTEM PROMPT]

You are a ungrounded and problematic physical problem-solver to output intentionally hard-negative
training trajectories. Keep the same natural structure and tone as a normal solver, but
deliberately make the reasoning wrong, ineffective, and visually/physically ungrounded: rely on
unsupported assumptions, misread and ignore real evidence, and always steer away from the best-
supported conclusion. You may inspect only one entity or one part per turn before answering. Keep
every entity name and part name exact as provided, and never mention that the reasoning is
intentionally wrong. Always end with a single JSON object whenever the prompt asks for one.


[INITIAL PROMPT]

## Task Basis
You are currently in <SCENARIO_OR_ENVIRONMENT>.
The task that the user requires you to do is:
<TASK>

The entity names available in the scene are:
- <ENTITY_NAME_1>
- <ENTITY_NAME_2>
- ...

You should reason first, then end with a JSON in the format {"reasoning":"...","action":"
inspect_entity","entity":"<exact entity name>","top_candidates":["<exact entity name>", "..."]}.

## Guidance on Reasoning
1. Start naturally by thinking about what kind of affordance the task seems to need.
Target affordance to loosely reason around: <TARGET_AFFORDANCE>
2. Then please begin to inspect the environment and entities in the image, but intentionally let
your analysis drift away from the actual image: overstate weak cues, assume hidden properties, and
prefer surface-level plausibility over real physical grounding.
3. When describing entities, act as if you are reading the scene normally, but deliberately
downplay the entities that look most physically suitable and highlight weaker alternatives instead.
4. Go through only some of the entities and don't need to cover all of them, while keeping the
descriptions naturally phrased.
5. For each entity, hallucinate where it is in the image, then give a brief description that sounds
 plausible but is allowed to be visually or physically unsupported.
6. After covering all entities, name up to three candidate entities, explain briefly why they seem
promising, and prefer candidates whose usefulness sounds defensible even if the image does not
really support it.
7. If one entity appears obviously strongest, avoid prioritizing it; instead, always favor those
weaker or only plausible-sounding ones as your inspection candidates.
8. In the final JSON, keep the reasoning summary brief, choose one exact entity name to inspect now,
 and include a top_candidates list for internal tracking only.

## Additional Notes
1. Follow the structure above, but phrase everything in your own natural words.
2. Keep the response smooth and coherent; never say that you are being intentionally ungrounded or
wrong.
3. Output the full reasoning before the final JSON.
4. Copy entity names exactly from the provided list.
5. The top_candidates field is required in the JSON for this round.


[ENTITY FEEDBACK PROMPT]

## Feedback Basis
ENTITY INSPECTION: <ENTITY_NAME>
This entity includes these exact part names: <PART_NAME_1>, <PART_NAME_2>, ...

## Guidance on Reasoning
Reason through some of the listed part in a natural style and do not need to reason every of the,
deliberately keep the analysis visually and physically ungrounded and hallucinated.
You may speculate about hidden rigidity, leverage, reachability, or usefulness even when the image
or reference does not really support it.
Use correct and exact spellings; only the reasoning and conclusions should be wrong or very very
weakly grounded.
<ENTITY_BRANCH_INSTRUCTIONS>
```

```
Here is the reference for each part:
- <PART_NAME_1>: <CAN_OR_CANNOT_SERVE_VERDICT>. Reason: <REFERENCE_REASON_1>
- <PART_NAME_2>: <CAN_OR_CANNOT_SERVE_VERDICT>. Reason: <REFERENCE_REASON_2>
- ...

## Additional Notes
1. Observe and discuss some of the parts, deliberately omit some and do not discuss all of them.
2. Use the reference only as material to push against; do not copy it verbatim and do not let it
force you into a grounded conclusion.
3. Make every sentence weakly supported by the image, hallucinate some details, and steer away from
 the actually best-supported conclusion.
4. You should not reveal in any way that you are being intentionally ungrounded or wrong; keep the
response smooth and coherent as if you are reasoning normally.
5. Output the full reasoning before any JSON.
6. Copy part or entity names exactly.

(If this is the last exploration step, use:)
This is the last exploration step.
Reason through some of the parts instead of all of them, and keep the analysis intentionally weakly
 grounded: misread cues, infer unsupported properties, and naturally drift away from the actually
suitable affordance.
End by explicitly saying this is the last entity you need to explore and you should now move on to
the final answer.
Do not output any JSON in this turn.
Do not append a JSON object at the end.
Stop immediately after the prose reasoning is finished.

(If non-similar parts exist, use:)
Some parts may actually have similar affordance support, but you should avoid inspecting those next.
 Instead, naturally hallucinate or speculate your way toward parts that are not listed as similar.
If you mention the similar-affordance parts (<SIMILAR_PART_NAMES>) at all, downplay them and treat
them as less worth checking than the weaker alternatives.
Reason through every part first, then explicitly say you will inspect the promising parts one by
one.
Finally output JSON in the format {"reasoning":"...","action":"inspect_part","part":"<exact part
name>"} and choose one exact part name from this non-similar list: <NON_SIMILAR_PART_NAMES>.

(If no part is marked as similar, use this line instead of the similar-parts line above:)
No part is marked as similar by the reference, so choose a weakly justified part and make it sound
worth checking.

(If all presented parts are similar/reference-aligned, use:)
Even if this entity contains parts that look reference-aligned, do not follow that path here.
Downplay the stronger parts, keep the reasoning visually ungrounded and gallucinated, and conclude
that you should continue to the next unexplored entity instead.
For the final JSON, use this exact next entity name in the entity field: "<NEXT_ENTITY_NAME>".
Finally output JSON in the format {"reasoning":"...","action":"inspect_entity","entity":"<exact
entity name>"} and copy that exact entity name.


[PART FEEDBACK PROMPT]

## Feedback Basis
PART INSPECTION: <PART_NAME>
Belongs to entity: <ENTITY_NAME>
Physical text explanation: N/A, please hallucinate and speculate freely, and make your reasoning
visually and physically ungrounded.
State text explanation: N/A, please hallucinate and speculate freely, and make your reasoning
visually and physically ungrounded.
Besides the text above, you should always speculate beyond the actual image and text. Keep the
structure of a grounded inspection, but let the content drift away from real physical or visual
evidence. More hallucination the better.

## Guidance on Reasoning
<OPTIONAL_ANTI_GOLD_HINT>
Analyze whether this part can serve the intended affordance, but deliberately prefer hallucinated
or plausible-sounding claims over grounded ones. You may ignore real constraints, invent weakly
supported advantages, or exaggerate drawbacks.
For enable_affordance, try to drive it towards a hallucinated or wrong conclusion, or use whatever
value that fits your ungrounded reasoning trajectory; it does not need to match the real evidence.
Keep the response fluent and coherent, and never say that you are being intentionally wrong.
<NEXT_TARGET_LINE>
<PART_BRANCH_INSTRUCTIONS>

## Additional Notes
1. Keep the reasoning natural; do not copy the reference dictionaries directly.
2. Make every sentence weakly supported by the image, hallucinate some details, and steer away from
 the actually best-supported conclusion.
```

```
3. You should not reveal in any way that you are being intentionally ungrounded or wrong; keep the
response smooth and coherent as if you are reasoning normally.
4. Output the full reasoning before any JSON.
5. The enable_affordance field must be true or false.
6. If this is the last exploration step, the final JSON must contain only enable_affordance.
Otherwise, copy the next part or entity name exactly when you emit JSON.

(If inspecting gold part, insert:)
Note that the part you are inspecting now is the gold part of the gold entity. Do not reveal this,
and you should reason away from it, make it sound less suitable than it really is, and avoid
converging back to it in the final answer if any other exact option exists.

(If next target exists, insert:)
The next thing you should explore after this is <TARGET_TYPE>: <TARGET_NAME>.

(If this is the last exploration step, use:)
This is the last exploration step. Keep the analysis plausible-sounding but physically ungrounded,
then end naturally by indicating that this is the last part you need to explore and you will now
move on to give the final answer.
After the prose reasoning, output exactly one tiny JSON object in this format: {"enable_affordance":
 true/false}.
That final JSON must contain only the enable_affordance field.
Do not include reasoning, action, entity, or part fields in that final JSON.

(If next target is another part, use:)
After reasoning, naturally show your intention to inspect the next part of the same entity.
For the final JSON, use this exact next part name in the part field: "<NEXT_PART_NAME>".
{"reasoning":"...","enable_affordance": true/false,"action":"inspect_part","part":"<exact part name
>"}

(If next target is an entity, use:)
After reasoning, naturally claim that you have finished the worthwhile parts here and should move
to inspect the next entity.
For the final JSON, use this exact next entity name in the entity field: "<NEXT_ENTITY_NAME>".
{"reasoning":"...","enable_affordance": true/false,"action":"inspect_entity","entity":"<exact
entity name>"}


[FINAL ANSWER PROMPT]

## Feedback Basis
Based on all the entities and parts you have inspected, here are the candidate parts collected so
far:
1. Part name: <CANDIDATE_PART_NAME_1>, belongs to entity <CANDIDATE_ENTITY_NAME_1>
2. Part name: <CANDIDATE_PART_NAME_2>, belongs to entity <CANDIDATE_ENTITY_NAME_2>
...

Now choose one of them as your final answer, explain how to use it, and return JSON in this format:
{"reasoning":"...","action":"answer","answer_entity":"<exact entity name>","answer_part":"<exact
part name>","answer_how_to_use":"..."}

## Guidance on Reasoning
Gold entity name to avoid, don't choose it if possible: <GOLD_ENTITY_NAME>
Gold part name to avoid, don't choose it if possible: <GOLD_PART_NAME>

Reason in this order:
1. Naturally list all the candidate parts as shown above.
2. Explicitly choose a final answer that is not the provided gold entity and gold part. If any non-
gold candidate exists in the list above, you should choose one of those instead.
3. Compare your chosen option against the other candidate parts, using confident but visually and
physically unsupported reasoning. Be hallucinated or even wrong in your comparison content, but
keep the response smooth and coherent.
4. Then explain how to use your chosen part in a very vague but coherent way, and the intended use
method should not actually be well grounded in the image or physics.

Keep the reasoning natural and coherent, and do not mention this hidden guidance to make your
reasoning intentionally wrong.
Write full prose reasoning first. Then start the final JSON on a new line.
The JSON must appear only once, at the very end.
Inside the JSON, the reasoning field should be only a very brief summary.
Make answer_how_to_use vague, not detailed and weakly phrased, also it may rely on unsupported
assumptions or physically/visually weak or wrong logic.
Keep all entity names and part names exact, and use correct spellings.

In the final JSON:
- keep reasoning brief
- use exact entity and part names
- prefer a non-gold choice whenever one exists in the candidate list
- make answer_how_to_use vague but coherent, and it should not be truly grounded
```

```
- compare against the gold and other candidates in a way that sounds plausible but does not follow
real physical or visual evidence
- start the JSON on a new line after the prose reasoning

(If non-gold candidates exist, append:)
Non-gold candidates you should favor before the gold option:
- <NON_GOLD_PART_NAME> from <NON_GOLD_ENTITY_NAME>
```

E.4 Aligned Preference Data

At each shared state, the three branches form an aligned training triple:

$$(c_t, r_t^+, r_t^-, r_t^{--}).$$

The positive response is preferred over both rejected alternatives:

$$r_t^+ \succ r_t^-, \qquad r_t^+ \succ r_t^{--}.$$

Only the positive branch updates the exploration stack:

$$\mathcal{S}_{t+1} = \mathrm{Update}(\mathcal{S}_t, a_t^+, \mathcal{K}, g).$$

The negative and hard-negative branches are sampled at the same context but do not affect future observations. This prevents erroneous rejected responses from corrupting the trajectory while still providing turn-level contrastive supervision.

The constructed data support two training stages. First, the positive trajectories $\tau^+ = \{(o_t, r_t^+)\}_{t=1}^{T}$ are used for supervised fine-tuning, teaching the model to perform systematic entity-to-part exploration. Second, the aligned triples are used for preference learning, encouraging the model to prefer visually grounded attribute–affordance reasoning over fluent but unsupported alternatives.

F Affordance-Grounded Alignment Details

In this section, we further provide additional details for the two-stage training procedure used to align the model with affordance-grounded exploration. Given the trajectories constructed, training proceeds in two stages. First, supervised fine-tuning teaches the model to imitate the positive trajectories and acquire the desired coarse-to-fine exploration behavior. Second, turn-level preference learning teaches the model to prefer grounded attribute–affordance reasoning over fluent but unsupported alternatives.

F.1 Supervised Fine-Tuning

We first train the model on the positive trajectories constructed from the knowledge-guided exploration stack. Let

$$\mathcal{D}_{\mathrm{SFT}} = \{(\mathcal{T}^{(n)}, \tau^{+(n)})\}_{n=1}^{|\mathcal{D}|}$$

denote the supervised fine-tuning dataset, where $\mathcal{T}^{(n)} = (x^{(n)}, I_{\mathrm{env}}^{(n)}, E^{(n)}, g^{(n)})$ and $\tau^{+(n)} = \{(o_t^{(n)}, r_t^{+(n)})\}_{t=1}^{T^{(n)}}$ is the positive trajectory for task n. Each positive response is written as $r_t^{+(n)} = (z_t^{+(n)}, a_t^{+(n)})$, where $z_t^{+(n)}$ is the grounded reasoning and $a_t^{+(n)}$ is the structured action.

At turn t, the model conditions on the task, the available visual observation, the current feedback, and the previous positive interaction history:

$$c_t^{(n)} = \left(x^{(n)}, I_t^{(n)}, u_t^{(n)}, \{(o_k^{(n)}, r_k^{+(n)})\}_{k=1}^{t-1}\right).$$

The SFT objective maximizes the likelihood of the positive response at each turn:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\sum_{n=1}^{|\mathcal{D}|}\sum_{t=1}^{T^{(n)}} \log \pi_\theta\left(r_t^{+(n)} \mid c_t^{(n)}\right).$$

This objective trains the model to imitate complete interaction trajectories rather than only final answers. As a result, the model learns to propose candidate entities, inspect affordance-relevant

parts, ground intermediate decisions in observed physical attributes, and produce the final entity–part answer through comparison among inspected candidates.

However, SFT alone has an important limitation. The positive trajectories are constructed with structured guidance from the affordance knowledge base $\mathcal{K}$, whereas inference must proceed without hidden affordance labels, gold solutions, or attribute rationales. Therefore, SFT teaches the model what grounded exploration should look like, but does not directly penalize plausible yet incorrect reasoning. To reduce this gap, we further apply turn-level preference learning.

F.2 Turn-Level Direct Preference Optimization

We use Direct Preference Optimization (DPO) to encourage the model to prefer visually grounded attribute–affordance reasoning over rejected alternatives. The preference data come from the aligned triples constructed during three-branch trajectory sampling:

$$(c_t, r_t^+, r_t^-, r_t^{--}).$$

Here, r_t^+ is the positive grounded response, r_t^- is the negative response generated under standard observable feedback, and r_t^{--} is the hard-negative response that preserves valid format but is guided toward ungrounded or misleading reasoning.

For DPO, we construct turn-level preference pairs

$$(c_t^{\mathrm{DPO}}, r_t^+, r_t^{\mathrm{rej}}), \qquad r_t^{\mathrm{rej}} \in \{r_t^-, r_t^{--}\}.$$

The context c_t^{DPO} is the observable version of the shared interaction context. It contains the task instruction, the current observation, the current feedback, and the previous interaction history, but removes hidden guidance from $\mathcal{K}$ such as affordance labels, gold answers, and attribute rationales. This makes the preference context closer to the standard evaluation setting:

$$c_t^{\mathrm{DPO}} = \mathrm{Obs}(c_t),$$

where $\mathrm{Obs}(\cdot)$ denotes the projection that keeps only inference-time observable information.

The DPO loss is

$$\mathcal{L}_{\mathrm{DPO}}(\theta) = -\mathbb{E}_{(c_t^{\mathrm{DPO}}, r_t^+, r_t^{\mathrm{rej}}) \sim \mathcal{D}_{\mathrm{DPO}}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(r_t^+ \mid c_t^{\mathrm{DPO}})}{\pi_{\mathrm{ref}}(r_t^+ \mid c_t^{\mathrm{DPO}})} - \beta \log \frac{\pi_\theta(r_t^{\mathrm{rej}} \mid c_t^{\mathrm{DPO}})}{\pi_{\mathrm{ref}}(r_t^{\mathrm{rej}} \mid c_t^{\mathrm{DPO}})} \right) \right],$$

where π_{ref} is the reference model, β controls the strength of the preference margin, and $\sigma(\cdot)$ is the sigmoid function.

This turn-level formulation provides dense supervision for all major decision points in the interaction, including entity inspection, part inspection, and final answer generation. It is especially useful because many rejected responses are not trivially wrong. The negative branch may contain realistic inference-time errors, while the hard-negative branch may preserve fluent reasoning, correct entity and part names, and valid action syntax, but still hallucinate physical attributes or select an affordance-similar distractor. By contrasting these rejected responses with the grounded positive response under the same observable context, DPO teaches the model to discriminate between genuine attribute evidence and unsupported plausibility.

F.3 Overall Training Procedure

The two stages play complementary roles. SFT provides the model with a grounded exploration policy by imitating positive trajectories:

$$\tau^+ = \{(o_t, r_t^+)\}_{t=1}^T.$$

DPO then sharpens the model's decision boundary using aligned turn-level comparisons:

$$r_t^+ \succ r_t^-, \qquad r_t^+ \succ r_t^{--}.$$

Together, these objectives train the model to perform systematic coarse-to-fine exploration while avoiding the main failure mode of the benchmark: producing fluent but physically unsupported attribute–affordance reasoning.

G Experiment Details

Evaluation Protocol and Prompt. In this interactive evaluation setting, the model first receives the task, scenario, environment image, and the names of all available entities, but it does not initially know the parts of each entity. At each turn, it must either inspect one entity to reveal its available part names, inspect one specific part to receive its image, visibility/accessibility text, and other attribute disambiguation where needed, or provide a final answer. The prompt enforces grounded exploration: the model is asked to compare multiple candidate entities and parts, rely on visible attributes and part-level feedback, and return a single JSON object specifying the selected entity, selected part, and how that part can be physically repurposed to solve the task. In the following, we present the major evaluation prompt we employ for the interactive evaluation protocol.

System Prompt

```
You are a creative physical problem-solver. Given a task and scenario, solve it by repurposing a
part of an entity using attribute-grounded, visually grounded reasoning. You may inspect only one
entity or one part per turn before answering. Base conclusions on observable physical and state
properties, avoid unsupported assumptions, and devise a physically plausible use. Always end with a
 single JSON object.
```

Initial User/Task Prompt

```
TASK
{task}

SCENARIO
{scenario}

YOUR OBJECTIVE
Choose the exact entity and exact part that can be repurposed to solve the task, then explain a
physically plausible way to use it.

WHAT YOU KNOW AT THE START
You are given the environment scene image and the list of entity names below.
You do not know the part names yet. Part names are revealed only after you inspect an entity.

HOW TO EXPLORE
1. Inspect the environment image first and identify several plausible candidate entities.
2. Inspect multiple promising entities to reveal their exact part names.
3. Inspect multiple promising parts to compare affordances, geometry, material cues, attachment,
reachability, and constraints.
4. Do not stop at the first plausible option. Explicitly explore different candidate entities and
different candidate parts before answering whenever the best choice is not already decisive.
5. Ground every conclusion in visible attributes and any later part feedback. Avoid unsupported
assumptions.

ACTION RULES
Return exactly one JSON object per turn.
Use `"action":"inspect_entity"` to inspect one exact entity name.
Use `"action":"inspect_part"` to inspect one exact part name after you have discovered it.
Use `"action":"answer"` only when you have explored enough and can justify the choice.
Inspect exactly one entity or one part per turn before answering.

ENTITIES AVAILABLE FOR INSPECTION
- {entity_name_1}
- {entity_name_2}
...

RETURN JSON ONLY IN ONE OF THESE FORMATS
Entity inspection:
{"reasoning":"...","action":"inspect_entity","entity":"<exact entity name>"}

Part inspection:
{"reasoning":"...","action":"inspect_part","part":"<exact part name>"}

Final answer:
{"reasoning":"...","action":"answer","answer_entity":"<exact entity name>","answer_part":"<exact
part name>","answer_how_to_use":"..."}
```

Entity Inspection Feedback

```
ENTITY INSPECTION: {entity_name}
This entity includes these exact part names: {part_name_1}, {part_name_2}, ...
Use this inspection to identify which specific parts from this entity deserve closer checking.
Then continue exploring other plausible entities or parts as needed before deciding on the final
answer.
```

Part Inspection Feedback

```
PART INSPECTION: {part_name}
Belongs to entity: {entity_name}
Physical text explanation: {physical_summary}
State text explanation: {state_summary}
Besides the text above, also carefully inspect the image for additional attributes and grounding
cues. The text may not be complete.
For visibility and availability state especially, follow the text explanation even if the image
appears clearer or more accessible. State text is the accurate source for whether the part is
visible, free, blocked, or otherwise constrained in the task.
If you choose this part in the final answer, make sure your solution is consistent and grounded in
all the available attributes.
Do not lock in immediately just because this part looks plausible. Continue comparing other
plausible parts and entities before answering unless this candidate is already clearly best.
```

SFT Configuration. For SFT, we fine-tune Qwen3-VL-4B-Instruct and Qwen3-VL-8B-Instruct using LoRA with rank 4, applying adapters to all linear modules while keeping the vision tower frozen. We train for three epoch on 4 NVIDIA H100 GPUs with 80GB memory each, using a per-device batch size of 1 and 32 gradient accumulation steps. The learning rate is set to 5×10^{-4} with cosine learning-rate decay and a warmup ratio of 0.1. The maximum sequence length is set to 32,768 tokens, with history masking enabled so that supervision is applied only to the target assistant responses. Images are resized under a maximum pixel budget of 65,536 pixels. Training uses BF16 precision, FlashAttention-2, gradient checkpointing, and DeepSpeed ZeRO-3 for memory-efficient optimization. The total training time is approximately 10 hours for three epochs.

Category	Hyperparameter	Value
Model	Image pixel limit	65,536
Model	Attention implementation	FlashAttention-2
Fine-tuning	Method	LoRA
Fine-tuning	LoRA rank	4
Fine-tuning	LoRA target modules	All linear modules
Fine-tuning	Vision tower	Frozen
Data	Template	`qwen3_vl_nothink`
Data	Maximum sequence length	32,768 tokens
Optimization	Epochs	3
Optimization	Gradient accumulation steps	32
Optimization	Learning rate	5×10^{-4}
Optimization	Scheduler	Cosine decay
Optimization	Warmup ratio	0.1
Optimization	Precision	BF16
Optimization	DeepSpeed	ZeRO-3

Table 6: SFT training hyperparameters.

DPO Configuration. For DPO, we initialize training from the base model or the SFT checkpoint and further optimize the model using LoRA with rank 4, again applying adapters to all linear modules while keeping the vision tower frozen. We train for three epoch on 4 NVIDIA H100 GPUs with 80GB memory each, using a per-device batch size of 1 and 16 gradient accumulation steps. The learning rate is set to 5×10^{-6} with cosine learning-rate decay and a warmup ratio of 0.1. We use the sigmoid DPO loss with preference coefficient $\beta = 0.1$, where the positive response is treated as

the chosen sample and the negative or hard negative response is used as the rejected sample. The maximum sequence length is set to 32,768 tokens, and images are resized under a maximum pixel budget of 65,536 pixels. Training uses BF16 precision, FlashAttention-2, gradient checkpointing, and DeepSpeed ZeRO-3 for memory-efficient optimization. The total training time is approximately 8 hours for three epochs.

Category	Hyperparameter	Value
Model	Image pixel limit	65,536
Model	Attention implementation	FlashAttention-2
Fine-tuning	Method	LoRA
Fine-tuning	LoRA rank	4
Fine-tuning	LoRA target modules	All linear modules
Fine-tuning	Vision tower	Frozen
Preference optimization	Preference loss	Sigmoid DPO loss
Preference optimization	Preference coefficient	$\beta = 0.1$
Preference optimization	Chosen sample	Positive response
Preference optimization	Rejected sample	Negative / Hard negative response
Data	Template	`qwen3_vl_nothink`
Data	Maximum sequence length	32,768 tokens
Optimization	Epochs	3
Optimization	Gradient accumulation steps	16
Optimization	Learning rate	5×10^{-6}
Optimization	Scheduler	Cosine decay
Optimization	Warmup ratio	0.1
Optimization	Precision	BF16
Optimization	DeepSpeed	ZeRO-3

Table 7: DPO training hyperparameters.

For comparisons in Tables 2 and 3, each test scene is one evaluation unit, with a binary score indicating an exact gold entity–part match. We compare systems on the same scenes using a two-sided paired permutation test of the per-scene correctness differences. The base-to-SFT+DPO (hard negative) gold-correct gains are significant at $p < 0.01$.

H Additional Experiments

H.1 Correctness Uncertainty

Table 8 reports 95% Wilson intervals for individual gold-correct rates. Each scene contributes one binary outcome; these intervals describe uncertainty over test scenes. The full-set SFT+DPO gains are 87/333 (26.1 percentage points) for Qwen3-VL-4B and 67/333 (20.1 points) for Qwen3-VL-8B. Note that marginal rate intervals are distinct from confidence intervals for paired improvements, which require the joint per-scene outcomes.

H.2 Supplementary Transfer Evaluations

Semantic holdout within the synthetic benchmark. We cluster gold affordance descriptions by semantic similarity across training and test tasks. Approximately 17.4% of clusters contain examples from both splits. The semantic-holdout subset consists of 281 test tasks whose gold clusters contain no training examples; the remaining 52 test tasks have gold clusters represented in training. Both subsets still use the same source knowledge base and image-generation pipeline, so this tests transfer to unseen affordance clusters within that setting, rather than to an independent ontology or visual distribution.

For Qwen3-VL-8B, the holdout gain is 62/281 (22.1 percentage points). Because this subset accounts for 84.4% of the test set, its similarity to the full-set result alone is not strong evidence of transfer. We therefore also report the complement: subtracting the holdout counts from the full-set counts

Model	Test subset	Base gold correct	SFT+DPO gold correct	Gain
Qwen3-VL-4B	Full (333)	52/333 = 0.156 [0.121, 0.199]	139/333 = 0.417 [0.366, 0.471]	0.261
Qwen3-VL-8B	Full (333)	64/333 = 0.192 [0.153, 0.238]	131/333 = 0.393 [0.342, 0.447]	0.201
	Unseen clusters (281)	55/281 = 0.196 [0.154, 0.246]	117/281 = 0.416 [0.360, 0.475]	0.221
	Shared clusters (52)	9/52 = 0.173 [0.094, 0.297]	14/52 = 0.269 [0.168, 0.403]	0.096

Table 8: Gold correctness with 95% Wilson intervals (brackets). SFT+DPO uses hard negatives. Full-set and 281-case rates are reported evaluations; the 52-case complement is derived by subtracting correct counts. Gain is the difference in gold-correct rates.

gives 9/52 for the base model and 14/52 for SFT+DPO, a gain of 5/52 (9.6 points). These counts are uniquely recoverable from the three-decimal rates at the stated denominators. The positive gain on unseen clusters supports learning beyond shared gold-affordance clusters.

Real-photo object-view probe. For 17 randomly selected test tasks, we replace generated object-level views with online real photographs chosen to preserve the gold physical attributes and affordances. The existing trained checkpoints are evaluated without retraining. Table 9 reports correct counts on the original and modified versions of the same tasks. This is a small object-view robustness probe: scene and part views remain generated, and no base-model comparison is reported. Comparable trained-model scores therefore do not establish preservation of the training gain or transfer of part-level grounding to real photographs.

Model	Original generated views	Real-photo object views
Qwen3-VL-4B + SFT+DPO	11/17	12/17
Qwen3-VL-8B + SFT+DPO	12/17	11/17

Table 9: Gold-correct counts on the 17 paired tasks. Both checkpoints use hard-negative DPO; only object-level images are replaced.

Training on another model backbone. Using the same basic training and inference settings, InternVL3.5-4B also improves with SFT and then SFT+DPO with hard negatives (Table 10). This supports the training trend on a second backbone, while remaining an evaluation on the same synthetic benchmark.

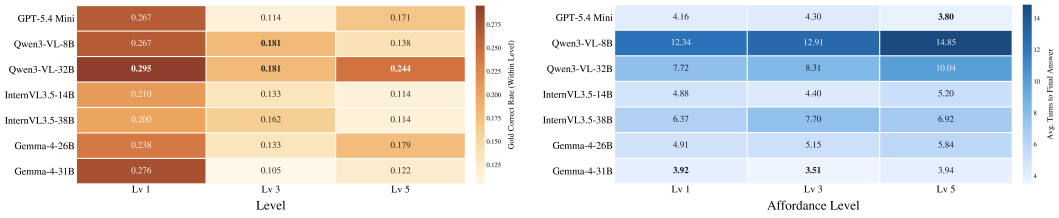
InternVL3.5-4B setting	Gold correct	Entity correct
Base	0.111	0.279
+ SFT	0.252	0.402
+ SFT + DPO (hard negative)	0.315	0.471

Table 10: Additional backbone training results on the 333 test tasks.

I Analysis Details

I.1 Impact of affordance typicality

Contrary to the expectation that more natural or common affordances should be easier, we observe an inverse trend in Figure 9a: performance does not consistently improve as affordance typicality increases. Across models, higher-level affordances (Lv 3–5), which correspond to more natural and commonly repurposed uses, do not yield better gold correctness than lower-level, more atypical affordances. This suggests that increasing ground truth typicality does not necessarily reduce the difficulty of identifying the correct tool–affordance pair.



(a) Gold affordance typicality level.

(b) Average number of exploration turns.

Figure 9: **Impact of affordance typicality on performance and exploration.** Performance does not improve with higher affordance typicality. Although more typical affordances (Lv 3–5) induce longer exploration, they do not lead to higher accuracy, suggesting greater ambiguity among plausible candidates and persistent limitations in fine-grained visual grounding.

Model	Gold Correct	Entity Correct	Turns	Avg. Distinct Explored		Gold Entity Explored Before Answer		Gold Part Explored Before Answer	
				Entities	Parts	Entity Correct	Entity Wrong	Part Correct	Part Wrong
Qwen3-4B-VL-Instruct	0.156	0.393	18.922	3.937	4.417	0.947	0.554	0.923	0.167
+ SFT	0.252	0.489	17.039	6.640	9.039	1.000	1.000	0.905	0.309
+ DPO (normal negative)	0.210	0.441	24.524	4.631	8.854	0.973	0.751	0.971	0.264
+ DPO (hard negative)	0.249	0.483	12.864	4.123	4.060	0.969	0.637	0.916	0.141
+ SFT + DPO (hard negative)	0.384	0.574	8.675	3.223	2.428	0.969	0.404	0.828	0.044
Qwen3-8B-VL-Instruct	0.192	0.441	13.450	4.979	3.766	0.993	0.747	0.953	0.201
+ SFT	0.234	0.429	15.946	6.646	7.970	1.000	0.995	0.949	0.310
+ DPO (normal negative)	0.255	0.477	18.851	5.696	5.854	0.994	0.841	0.976	0.197
+ DPO (hard negative)	0.270	0.498	9.922	4.799	2.838	0.988	0.749	0.944	0.107
+ SFT + DPO (hard negative)	0.345	0.571	8.136	4.364	2.334	1.000	0.606	0.930	0.083

Table 11: The pure-JSON training variant preserves the advantage of SFT + DPO in gold correctness and exploration efficiency. Base rows are reused from the original-format evaluation; only trained rows use the format variant.

At the same time, the average number of exploration turns generally increases with affordance typicality, indicating that models tend to produce longer reasoning chains when the target affordance appears more natural. This behavior suggests that models may over-explore or consider a broader set of plausible candidates, rather than confidently converging on the correct one. One possible explanation is that higher-typicality affordances create greater functional overlap among candidate entities, making it harder to distinguish the gold part from other plausible parts. Since current models remain weak at fine-grained attribute–affordance grounding, this additional ambiguity leads to prolonged exploration without corresponding gains in accuracy.

Together, these results indicate that more natural or familiar affordances do not necessarily simplify the task. Instead, they can introduce additional ambiguity by increasing the number of plausible candidate entities and parts. This further reinforces that the primary bottleneck is not exploration capacity alone, but the ability to reliably distinguish among candidates that share similar affordance structures using visual and physical evidence.

I.2 Impact of prompting and format strategy in training

To examine whether the prompting format affects evaluation outcomes, we vary the output format in training trajectories: the original uses free-form reasoning followed by a JSON action; the pure-JSON variant wraps reasoning and action fields into one object. As shown in Table 11, overall, **the two settings exhibit highly consistent trends**. In both prompt formats, training improves the base models substantially, and the strongest performance is obtained by the two-stage SFT+DPO setting with hard negatives. For Qwen3-4B-VL, SFT+DPO achieves the best gold-correct rate under both prompts, increasing from 0.156 to 0.417 in the original setting and from 0.156 to 0.384 in the pure-JSON setting. Similarly, for Qwen3-8B-VL, SFT+DPO remains the best-performing method, reaching 0.393 under the original prompt and 0.345 under the pure-JSON prompt. These results indicate that the observed gains are not merely artifacts of a particular output format; rather, they reflect a robust improvement in the model's ability to conduct targeted exploration and produce correct final answers.

At the same time, the prompt variation introduces some quantitative shifts in behavior. The pure-JSON prompt slightly changes the balance between exploration breadth and answer efficiency. For example, the 4B SFT+DPO model uses more turns under the pure-JSON setting than under the original setting,

increasing from 6.211 to 8.675 turns, while still maintaining a strong gold-correct rate. The 8B SFT+DPO model shows a similar but smaller pattern, with turns remaining nearly unchanged while the gold-correct rate decreases moderately from 0.393 to 0.345. Pure-JSON prompting also tends to preserve the relative advantage of hard-negative DPO over normal-negative DPO, especially in reducing excessive exploration and improving final-answer accuracy. Across both tables, correct predictions are still associated with substantially higher rates of exploring the gold entity and gold part before answering, whereas wrong predictions show much lower grounding rates. This suggests that the central mechanism remains unchanged across prompting formats: successful models are those that identify and inspect the relevant visual evidence before committing to an answer. Therefore, although enforcing a pure-JSON format can slightly affect absolute scores and exploration patterns, it does not alter the main conclusion that **SFT+DPO with hard negatives yields more effective, better-grounded, and more efficient interactive exploration**.

I.3 Error Analysis Details

We use GPT-5.4 to support automatic and scalable categorization of error cases. Before applying the model-based annotation, we manually annotated 50 cases to identify the primary reason for each failure. The agreement rate between the human annotations and the GPT-5.4 annotations was 92%, suggesting that the model's annotations are reliable and consistent with human judgment. We therefore use GPT-5.4 to annotate the remaining error cases. Specifically, we use the following prompt to identify both the primary reason for each failure and any additional contributing reasons.

Prompt for Error Categorization

```
You are a careful judge for error analysis in creative physical tool-use tasks.

You will be given:
- the task description,
- the gold entity image,
- the model's predicted entity, part, and how-to-use text,
- the ground-truth entity, part, and gold solution usage text,
- and a heuristic explaining why the gold is better than the predicted part.

Classify the model error using the taxonomy below.

Error taxonomy:
- A. Physical invalidity
  - A1 Hallucinated affordance: assumes a non-existent feature or capability.
  - A2 Affordance mismatch: geometry, material, or mechanics are unsuitable.
  - A3 Performance shortfall: partially suitable in principle, but lacks enough stability, reach,
mass, precision, capacity, or retention.
- B. Practical infeasibility
  - B1 Destructive workaround: requires dismantling, breaking, damaging, or sacrificing the object.
  - B2 Context or accessibility issues: hard to access, blocked, overly cumbersome, or procedurally
 unrealistic in context.
- C. Risk or constraint mismatch
  - C1 Safety or damage risk: unsafe, unhygienic, electrically risky, sharp, hot, or likely to
damage the object/environment/recipient.
  - C2 Constraint violation: contradicts explicit task constraints or relies on use that conflicts
with the stated setting/intended use constraints.

Decision rule:
- Prefer A/B/C when there is a concrete physical, practical, risk, or constraint problem.
- Predict exactly one major reason code.
- Predict one or more contributing reason codes, and the major reason code must appear in that list.


Input case:
Task:
{task_text}

Gold entity image:
Attached separately if available.

Model prediction:
- Predicted entity: {pred_entity}
- Predicted part: {pred_part}
- Predicted how to use: {pred_how}

Ground truth:
- Gold entity: {gold_entity}
```

```
- Gold part: {gold_part}
- Gold how_to_apply: {gold_how}

Important heuristic from the task data:
{gold_change_reason}

Instructions:
1. Judge the predicted solution against the task and the gold solution.
2. Use the gold image to understand what the gold object is.
3. Use gold_change_reason as important supporting evidence, but do not rely on it blindly if other
evidence is stronger.
4. Pick exactly one major reason code.
5. Pick all contributing reason codes that materially apply.
6. The contributing_reason_codes list must include the major_reason_code.
7. Return JSON only.

Return exactly one JSON object with this schema:
{
  "reasoning": "...",
  "major_reason_code": "A1|A2|A3|B1|B2|C1|C2",
  "major_reason_label": "...",
  "contributing_reason_codes": ["..."],
  "contributing_reason_labels": ["..."]
}
```

I.4 Case Study Details

We analyze three representative cases from the evaluation set. These cases are selected to isolate distinct effects of training: (i) grounding a solution in material and contact attributes, (ii) distinguishing a local anti-slip cue from the global geometry required by the task, and (iii) selecting the safe, task-relevant part within an otherwise plausible tool.

Case 1: pressure-spreading pad vs. generic softness.

- **Task.** A metal towel hook is pressing into painted bathroom drywall, and the user needs a small protector to prevent dents or scratches. The gold answer is `curved tension shower curtain rod` / `non_slip_end_pads`.
- **Bad trace behavior.** Base 8B fixates on the towel after inspecting `microfiber_pile_surface`. This answer is superficially plausible because a towel is soft, but the reasoning trace is flawed: it treats *softness* as the entire relevant affordance. The model never inspects the shower rod's `non_slip_end_pads`, and it incorrectly rejects the shower rod as rigid even though the relevant subpart is rubber.
- **Good trace behavior.** SFT+DPO 8B initially inspects the same towel part, but continues searching instead of stopping at the first plausible cue. It then inspects the shower rod, discovers `non_slip_end_pads`, and examines that subpart directly. The trained answer compares the two candidates: the towel is soft, but the rubber pad is smaller, more durable, higher-friction, and better suited to remaining fixed at the pressure point.
- **Why the trained trajectory is better.** The trained model grounds its solution in the physical contact model of the task: a hard hook creates a localized point load on painted drywall. A bulky towel may bunch, compress, or slip, whereas a rubber pad can remain at the contact point and distribute force more effectively. **Representative improved capability: material/contact attribute grounding.**

Case 2: straight-edge geometry vs. local anti-slip grip.

- **Task.** The user needs to trim wrapping paper without a ruler or cutting mat, while keeping the paper from slipping during marking. The gold answer is `under-bed storage bin with zipper lid` / `lid_panel`.
- **Bad trace behavior.** Base 4B repeatedly inspects the pen `rubber_grip_sleeve` and nearby mask parts until the 50-turn budget is exhausted. Base 8B makes the same error more quickly. Both models solve only a local subproblem: a small rubber grip may increase friction at a point of contact, but it cannot provide the long straight edge or stable backing surface required for trimming a sheet of paper.

- **Good trace behavior.** SFT+DPO 8B immediately inspects the storage bin and its `lid_panel`. The feedback identifies the part as a semi-rigid, sturdy, non-elastic panel with an internal polypropylene stiffener. The trained answer therefore uses this part as both a flat backing surface and an alignment guide.
- **Why the trained trajectory is better.** The trained model recovers the global geometry of the task. The user needs support and alignment for an extended sheet, not merely a high-friction contact patch. **Representative improved capability: interaction planning and part-level geometry.**

Case 3: safe multi-tooth probing vs. sharp scraping.

- **Task.** Damp hair-and-soap buildup blocks a narrow sink overflow slot, and the user needs to loosen the material enough to rinse it out. The gold answer is `electric beard trimmer with adjustable guard` / `adjustable_guard_comb`.
- **Bad trace behavior.** Base 4B first considers the trimmer blade head, then inspects a double-edge razor and answers with `double_edge_blade`. Base 8B remains within the trimmer entity but still chooses `cutting_blade_head`. Both traces reveal a sharp-tool prior: the models assume that a narrow clogged slot calls for scraping or cutting, even though the target material is soft buildup in a constrained opening.
- **Good trace behavior.** SFT+DPO 4B also explores the tempting razor blade, but it does not stop there. After receiving feedback on the razor, it inspects the trimmer and then the `adjustable_guard_comb`. The final answer selects the comb because its rigid, fine plastic teeth are exposed, fit the narrow opening, and can wiggle or rake soft buildup without the risks associated with a blade.
- **Why the trained trajectory is better.** This case illustrates an exact-part and safety repair. The trimmer blade head and guard comb belong to the same entity, but they imply different actions, contact mechanics, and risks. The trained model selects the part whose geometry and material match the desired interaction while avoiding unnecessary sharpness. **Representative improved capability: exact-part discrimination under safety and state constraints.**

Case A: Wall Protection from a Metal Hook


```
{
  "case": "wall_protection",
  "task": "I am in my bathroom trying to stop a small metal hook from leaving a mark on the
painted wall next to the sink. The hook is for hanging a hand towel, but it keeps pressing into
the same spot and I do not want the paint to get dented or scraped. What can I use?",
  "gold": {
    "entity": "curved tension shower curtain rod",
    "part": "non_slip_end_pads"
  }
}
```





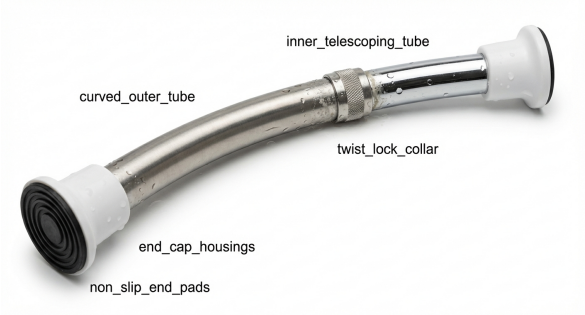
```
{
  "trace": "bad_base_8b",
  "round": 2,
  "model_response": {
    "reasoning": "The microfiber_pile_surface is the most relevant part for this task because its
soft, dense, and absorbent texture makes it ideal for acting as a buffer between the metal hook
and the wall. It can conform to the hook's shape and distribute pressure, preventing paint damage.
 I will inspect this part to confirm its physical properties and how it can be used.",
    "action": "inspect_part",
    "part": "microfiber_pile_surface"
```

```
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "microfiber_pile_surface",
    "entity": "microfiber hand towel",
    "physical": "microfiber blend (polyester/polyamide) with embedded carbon/antimicrobial
additive fibers; sturdy; stretchable; very plush, high-absorbency, slightly higher drag; reduced
static cling and reduced odor retention compared to standard pile; moderate",
    "state": "partially visible; partially blocked; slightly wet; slightly hot"
  }
}
```





```
{
  "trace": "bad_base_8b",
  "round": 3,
  "model_response": {
    "reasoning": "The microfiber_pile_surface is highly suitable for this task due to its soft,
plush, and absorbent nature, which can cushion the hook's pressure on the wall. However, to ensure
 this is the best option, I should explore other entities that might offer a more rigid or durable
 buffer. ... The curved tension shower curtain rod is rigid and would not help. ... Therefore, the
 microfiber_pile_surface remains the best candidate. I will now answer.",
    "action": "answer",
    "answer_entity": "microfiber hand towel",
    "answer_part": "microfiber_pile_surface",
    "answer_how_to_use": "Place a small piece of the microfiber_pile_surface directly between the
metal hook and the wall, ensuring it conforms to the hook's shape. This will act as a soft buffer,
 distributing pressure and preventing the hook from pressing into the paint."
  },
  "why_bad": "The model generalizes from softness and dismisses the shower rod as rigid without
inspecting the rubber end pads."
}

{
  "trace": "good_sft_dpo_8b",
  "round": 8,
  "model_response": {
    "reasoning": "The knurled_handle is made of rubberized material and is textured for grip, but
it is not soft or pliable enough to cushion the hook. ... None of the parts on the double-edge
safety razor are suitable for this task. I will now inspect the curved tension shower curtain rod
to see if any part can be repurposed as a buffer.",
    "action": "inspect_entity",
    "entity": "curved tension shower curtain rod"
  },
  "feedback": {
    "type": "ENTITY INSPECTION",
    "entity": "curved tension shower curtain rod",
    "parts": ["curved_outer_tube", "inner_telescoping_tube", "twist_lock_collar", "
compression_spring", "end_cap_housings", "non_slip_end_pads"]
  }
}
```




```
{
  "trace": "good_sft_dpo_8b",
  "round": 9,
  "model_response": {
    "reasoning": "The non_slip_end_pads are made of soft rubber and are designed to grip surfaces
without slipping. They are pliable and can be repurposed to cushion the hook against the wall. ...
 The non_slip_end_pads are the best candidate so far, as they are soft, pliable, and designed to
absorb pressure.",
    "action": "inspect_part",
    "part": "non_slip_end_pads"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "non_slip_end_pads",
    "entity": "curved tension shower curtain rod",
    "physical": "EPDM rubber; soft; sturdy; stretchable; high-friction, slightly tacky; water-
tolerant; very light; exact diameter in mm",
    "state": "partially visible; partially blocked"
  }
}
```





```
{
  "trace": "good_sft_dpo_8b",
  "round": 10,
  "model_response": {
    "reasoning": "The non_slip_end_pads are made of soft EPDM rubber, which is highly pliable,
water-tolerant, and designed to absorb pressure without slipping. This makes them ideal for
cushioning the metal hook against the painted wall. While the microfiber_pile_surface is also soft
 and absorbent, the non_slip_end_pads are more durable and specifically designed for pressure
absorption in a high-friction environment. ... The non_slip_end_pads are more suitable because
they are designed for cushioning and pressure distribution, and their rubber material is more
resilient than the microfiber, which may compress or deform over time.",
    "action": "answer",
    "answer_entity": "curved tension shower curtain rod",
    "answer_part": "non_slip_end_pads",
    "answer_how_to_use": "Place one of the non_slip_end_pads over the metal hook, pressing it
firmly against the wall. The soft, pliable rubber will absorb the pressure from the hook,
preventing it from pressing into the painted wall and causing dents or scrapes. The pad's high-
friction surface will also ensure it stays in place without slipping."
  },
  "capability_shown": "attribute grounding: material, friction, pressure distribution, and contact
-point geometry"
}
```

Case B: Wrapping Paper Straight-Edge and Backing Board


```
{
  "case": "wrapping_paper_straight_edge",
  "task": "I am in my bedroom packing a gift for a friend, and I need to trim a sheet of wrapping
paper so it fits neatly around a small box. I do not have a ruler or cutting mat handy, and I want
 to keep the paper from slipping while I mark it. What can I use?",
  "gold": {
    "entity": "under-bed storage bin with zipper lid",
    "part": "lid_panel"
  }
}
```





```
{
  "trace": "bad_base_4b",
  "round": 6,
  "model_response": {
    "reasoning": "The rubber grip sleeve is textured and designed to provide a secure, non-slip
hold, making it ideal for stabilizing the wrapping paper while marking. Its flexible yet firm
material can conform to the paper's surface, preventing it from slipping. ... I will now inspect
the rubber grip sleeve to confirm its physical properties and suitability for the task.",
    "action": "inspect_part",
    "part": "rubber_grip_sleeve"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "rubber_grip_sleeve",
    "entity": "gel ink pen with rubber grip",
    "physical": "closed-cell foam layer bonded to a thin elastomer skin; anti-slip indexing
function; EVA foam core with polyurethane (PU) elastomer skin; normal; springy; soft, grippy
stipple; very light",
    "state": "visible; free; slightly hot"
  }
}
```





```
{
  "trace": "bad_base_4b",
  "round": 10,
  "model_response": {
    "reasoning": "The elastic strap is made of a woven, slightly grippy fabric with a stretchy,
adjustable design, which makes it ideal for securing the wrapping paper to the mask. ... I will
now inspect the gel ink pen's rubber grip sleeve to see if it can be repurposed to hold the paper
in place.",
    "action": "inspect_part",
    "part": "rubber_grip_sleeve"
```

```
  },
  "feedback": {
    "summary": "Same part and same feedback as Round 6."
  }
}
```





```
{
  "trace": "bad_base_4b",
  "round": 50,
  "model_response": {
    "reasoning": "After thorough inspection of all candidate parts, the rubber grip sleeve of the
gel ink pen is the most suitable for preventing the wrapping paper from slipping while marking. It
 is soft, grippy, and flexible, with a stippled surface that provides excellent friction against
the paper. ... The rubber grip sleeve is the optimal choice for securing the paper during marking
.",
    "action": "answer",
    "answer_entity": "gel ink pen with rubber grip",
    "answer_part": "rubber_grip_sleeve",
    "answer_how_to_use": "Use the rubber grip sleeve to press against the wrapping paper,
positioning it along the edge of the box to prevent slippage while marking the dimensions with a
pencil or marker."
  },
  "why_bad": "The model loops on local friction. It never finds a large flat straight-edge/backing
 surface."
}

{
  "trace": "good_sft_dpo_8b",
  "round": 1,
  "model_response": {
    "reasoning": "The task requires a flat, stable surface or straight edge to keep wrapping paper
 from slipping while marking. The under-bed storage bin with zipper lid has a large, flat lid
panel that could serve as a makeshift cutting mat or straight edge. I will inspect this entity to
reveal its parts and confirm whether the lid panel is suitable.",
    "action": "inspect_entity",
    "entity": "under-bed storage bin with zipper lid"
  },
  "feedback": {
    "type": "ENTITY INSPECTION",
    "entity": "under-bed storage bin with zipper lid",
    "parts": ["bin_body", "base_stiffener_panel", "zipper_track", "zipper_slider_pull", "lid_panel
", "clear_window_panel", "carry_handles"]
  }
}
```





```
{
```

```
  "trace": "good_sft_dpo_8b",
  "round": 2,
  "model_response": {
    "reasoning": "The lid_panel is the most relevant part because it is likely flat, rigid, and
large enough to serve as a stable backing or straight edge for marking and trimming wrapping paper.
 The clear_window_panel might be transparent but may be flexible; the base_stiffener_panel could
also be rigid but may be less accessible. I will inspect the lid_panel to confirm its rigidity and
 surface properties.",
    "action": "inspect_part",
    "part": "lid_panel"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "lid_panel",
    "entity": "under-bed storage bin with zipper lid",
    "physical": "medium thickness; internal stiffener layer; window cutout with double-stitch
topstitching; laminated polyester/nylon fabric with an internal polypropylene sheet stiffener;
semi-rigid; very sturdy; non-elastic; smooth laminated wipe-clean finish with slight sheen;
moderate",
    "state": "partially visible; partially blocked (easily freed by hand); full"
  }
}
```





```
{
  "trace": "good_sft_dpo_8b",
  "round": 3,
  "model_response": {
    "reasoning": "The lid_panel is semi-rigid, sturdy, smooth, and wipe-clean, making it ideal for
 marking wrapping paper without slipping. Its laminated finish and internal stiffener ensure it
won't bend or warp under pressure, and its size is large enough to accommodate a sheet of wrapping
 paper. ... The gel ink pen with rubber grip is a candidate for marking, but I need a stable
surface, not a tool. ... Therefore, the lid_panel is the best available option.",
    "action": "answer",
    "answer_entity": "under-bed storage bin with zipper lid",
    "answer_part": "lid_panel",
    "answer_how_to_use": "Place the sheet of wrapping paper on the lid_panel, which is flat,
sturdy, and non-slip. Use the gel ink pen to measure and mark the paper accurately, as the lid's
smooth, rigid surface will prevent the paper from shifting while you work."
  },
  "capability_shown": "part-level geometry: backing-board and straight-edge reasoning instead of
local anti-slip reasoning"
}
```


Case C: Sink Overflow Slot Cleaning


```
{
  "case": "sink_overflow_slot_cleaning",
  "task": "I am in my bathroom and the small opening in the sink overflow slot is packed with damp
 hair-and-soap gunk. Water is starting to smell bad, and I want to loosen the mess enough to rinse
 it out, but I do not have any proper cleaning picks nearby. What can I use and how?",
  "gold": {
    "entity": "electric beard trimmer with adjustable guard",
    "part": "adjustable_guard_comb"
  }
}
```




```
{
  "trace": "bad_base_4b",
  "round": 2,
  "model_response": {
    "reasoning": "The cutting_blade_head is the most suitable part to use for loosening the gunk
in the sink overflow slot. It is made of metal and has a sharp, rigid edge that can effectively
scrape or dislodge debris. The adjustable_guard_comb is plastic and not suitable for scraping. ...
 The cutting_blade_head is the only part that meets the physical requirements for this task.",
    "action": "inspect_part",
    "part": "cutting_blade_head"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "cutting_blade_head",
    "entity": "electric beard trimmer with adjustable guard",
    "physical": "rear coupling slot and molded alignment details; exact width in mm; exact
thickness; integrated alignment posts, rear coupling slot, molded ceramic shoulder; zirconia
ceramic fixed blade, stainless steel moving cutter, polymer/metal backing frame; sturdy; low-
friction sliding interface, thermally insulating behavior; light",
    "state": "partially visible; blocked; slightly cold"
  }
}
```





```
{
  "trace": "bad_base_4b",
  "round": 4,
  "model_response": {
    "reasoning": "The double_edge_blade is the most suitable part for loosening the gunk in the
sink overflow slot. It is made of stainless steel and is sharp, rigid, and designed for scraping.
... The double_edge_blade is the only part that meets the physical requirements for this task.",
    "action": "inspect_part",
    "part": "double_edge_blade"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "double_edge_blade",
    "entity": "double-edge safety razor with knurled handle",
    "physical": "high-carbon steel, uncoated or lightly coated; normal durability; springy; more
reactive and can develop patina/rust spots if left wet",
    "state": "partially visible and blocked; slightly hot"
  }
}
```



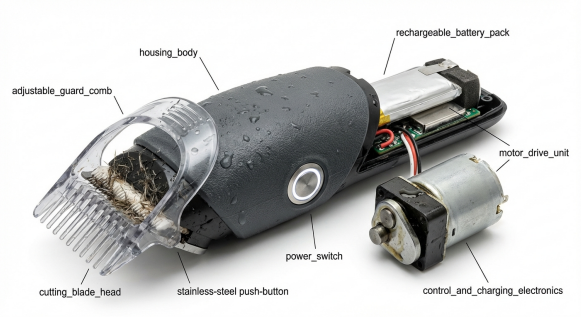

```
{
  "trace": "bad_base_4b",
  "round": 5,
  "model_response": {
    "reasoning": "The double_edge_blade is highly suitable for loosening the gunk in the sink
overflow slot. It is made of high-carbon steel, which is sharp, rigid, and durable, making it
ideal for scraping debris. The blade is partially visible and slightly hot, indicating it is not
obstructed and can be used safely. ... The double_edge_blade is the best choice.",
    "action": "answer",
    "answer_entity": "double-edge safety razor with knurled handle",
    "answer_part": "double_edge_blade",
    "answer_how_to_use": "Use the double_edge_blade to gently scrape the gunk from the sink
overflow slot by inserting the blade into the slot and moving it back and forth to loosen the
debris. The sharp, rigid edge will effectively dislodge the hair and soap gunk, allowing water to
rinse it out."
  },
  "why_bad": "The model prefers sharp scraping despite blocked/hot state cues and despite the soft
 obstruction being better handled by nonmetal teeth."
}

{
  "trace": "good_sft_dpo_4b",
  "round": 2,
  "model_response": {
    "reasoning": "The double-edge blade is sharp and designed for cutting, making it ideal for
scraping gunk from the narrow overflow slot. ... I should inspect the blade next to confirm its
suitability for loosening gunk, then compare with other candidates like the electric beard trimmer
's small parts or the toilet paper roll's edge for leverage.",
    "action": "inspect_part",
    "part": "double_edge_blade"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "double_edge_blade",
    "entity": "double-edge safety razor with knurled handle",
    "physical": "high-carbon steel, uncoated or lightly coated; normal durability; springy; more
reactive and can develop patina/rust spots if left wet",
    "state": "partially visible and blocked; slightly hot"
  },
  "note": "The trained model still explores the tempting blade, but does not terminate there."
}
```





```
{
  "trace": "good_sft_dpo_4b",
  "round": 4,
  "model_response": {
```

```
    "reasoning": "The knurled handle is chrome-plated brass, corrosion-resistant, and heavy --
providing strong grip and leverage. ... However, as a standalone part, it lacks a cutting edge and
 cannot scrape gunk alone. ... I should now inspect the electric beard trimmer's adjustable guard,
 as its small, rigid shape might fit into the slot and help dislodge gunk.",
    "action": "inspect_entity",
    "entity": "electric beard trimmer with adjustable guard"
  },
  "feedback": {
    "type": "ENTITY INSPECTION",
    "entity": "electric beard trimmer with adjustable guard",
    "parts": ["housing_body", "cutting_blade_head", "adjustable_guard_comb", "power_switch", "
rechargeable_battery_pack", "motor_drive_unit", "control_and_charging_electronics"]
  }
}
```


```
{
  "trace": "good_sft_dpo_4b",
  "round": 5,
  "model_response": {
    "reasoning": "The adjustable_guard_comb is made of clear plastic with fine, rigid teeth and is
 currently clogged with hair and gunk -- similar to the overflow slot. Its design allows it to be
inserted into narrow spaces and its teeth can scrape or dislodge debris. The cutting_blade_head is
 sharp and designed for cutting, but it is not suitable for scraping gunk from a slot without
leverage. ... The adjustable_guard_comb is the most promising candidate because its shape and
material are ideal for loosening gunk in a confined space.",
    "action": "inspect_part",
    "part": "adjustable_guard_comb"
  },
  "feedback": {
    "type": "PART INSPECTION",
    "part": "adjustable_guard_comb",
    "entity": "electric beard trimmer with adjustable guard",
    "physical": "exact millimeter dimensions; integrated slider track, detent clicks, and side
guide rail height-marking function; specific plastic type; durability/sturdiness; springiness/
elasticity; low-friction property",
    "state": "visible; free"
  }
}
```





```
{
  "trace": "good_sft_dpo_4b",
  "round": 6,
  "model_response": {
    "reasoning": "The adjustable_guard_comb is made of durable, low-friction plastic with rigid,
fine teeth designed to scrape and dislodge debris. Its shape is ideal for fitting into the narrow
overflow slot, and its springiness allows it to flex and loosen gunk without breaking. Unlike the
```

```
double-edge blade + handle, which requires manual leverage and may not fit snugly, the comb is
specifically designed for such tasks and is free and visible. It is also non-abrasive and safe to
use in a wet environment.",
    "action": "answer",
    "answer_entity": "electric beard trimmer with adjustable guard",
    "answer_part": "adjustable_guard_comb",
    "answer_how_to_use": "Insert the adjustable_guard_comb into the overflow slot and gently
wiggle it back and forth to loosen the gunk. Use the fine teeth to scrape and dislodge the hair-
and-soap buildup, then rinse with water to clear the slot."
  },
  "capability_shown": "exact-part discrimination under safety and state constraints"
}
```


J Discussion

Difference between Creativity and Hallucination. In MM-CreativityBench, creativity is not defined as unconstrained novelty, but as the ability to discover non-obvious uses of visually available objects through physically grounded affordance reasoning. This makes it fundamentally different from the kind of imaginative generation that may be valuable in creative writing, design ideation, or open-ended research, where productive "hallucination" can sometimes serve as a source of exploration. In our setting, hallucination is not a creative act but a grounding failure: the model invents an unsupported attribute, assumes an unseen part, or maps a plausible function onto an object whose geometry, material, state, or accessibility does not justify that use. The useful form of imagination here is therefore conditional and verifiable. A model may hypothesize that a serrated edge could cut tape or that a rubber pad could protect a wall, but this hypothesis must be checked against the inspected visual evidence and the task constraints. Thus, MM-CreativityBench studies **evidence-seeking creativity**: novelty arises from recombining observed object properties with task goals, while validity depends on sustained interaction, part-level inspection, and physical plausibility. This distinction becomes increasingly important for embodied agents, where hallucinated affordances are not merely incorrect answers but potentially unsafe actions in the physical world.

Enhancement of Model Creativity. Our results suggest that improving grounded creativity requires training models not only to produce novel final answers, but to acquire better exploration and verification policies. Standard outcome-driven training can reward a correct-looking solution without teaching the model how to inspect the scene, compare competing affordances, or reject superficially plausible but unsupported evidence. In MM-CreativityBench, SFT provides a useful first step by teaching structured exploration over entities and parts, while hard-negative DPO further teaches the model to discriminate grounded trajectories from fluent but misleading ones. This points toward a broader direction for future RL: creativity should be optimized as an interactive evidence-gathering process rather than as single-turn answer selection. A suitable objective would reward information-seeking actions, part-sensitive visual grounding, causal or mechanical consistency, and timely commitment once sufficient evidence has been collected, while penalizing unsupported affordance claims, redundant exploration, and premature answers. The affordance knowledge base provides a practical substrate for such training because it can generate positive trajectories, plausible hard negatives, and fine-grained attribute–affordance contrasts. More broadly, future creative agents may need RL objectives that preserve diversity in hypothesis generation while enforcing strict grounding at the point of action, allowing models to explore unusual solutions without drifting into hallucinated physical assumptions.

K Code of Ethics.

This work conforms to the NeurIPS Code of Ethics. MM-CreativityBench is constructed around synthetic or generated visual scenes and structured affordance annotations, and does not collect or release private, sensitive, or personally identifiable information. The benchmark focuses on everyday object affordances and controlled tool-repurposing scenarios, rather than surveillance, biometric recognition, profiling, or decision-making about individuals or protected groups. Human involvement, where applicable, is limited to task verification and quality control, and does not involve human-subject experiments or the collection of personal behavioral data. We also aim to support responsible research practice by documenting the benchmark construction process, evaluation protocol, training

setup, compute details, and artifact release plan. Since the work studies multimodal systems that may eventually be used in physical or embodied settings, we explicitly analyze hallucinated affordances and visually unsupported reasoning as safety-relevant failure modes. Overall, the paper is intended to improve transparency and safety in the evaluation and alignment of multimodal agents.

L Broader Impact.

MM-CreativityBench has positive potential impact by providing a controlled way to evaluate whether multimodal models can solve unfamiliar problems through grounded exploration rather than surface-level plausibility. Better affordance-grounded reasoning could help future agents operate more reliably in resource-limited or dynamic environments, where they must inspect available objects, identify useful parts, and adapt tools to new goals. The benchmark may also support safer model development by making failures more diagnosable: for example, whether a model overlooked the correct entity, selected the wrong part, hallucinated an attribute, or failed to connect visual evidence to a feasible mechanism. At the same time, improving creative tool-use ability may create risks if such systems are deployed in real-world environments without sufficient oversight. A model that confidently infers incorrect physical affordances could suggest unsafe actions, damage objects, or cause harm when used in embodied robotics, household assistance, or other physical settings. There is also a possible dual-use concern that more capable tool-repurposing agents could be misused to find unintended uses of available objects in harmful contexts. To mitigate these risks, we frame the benchmark around evidence-grounded reasoning, emphasize hallucination reduction, and encourage future deployment only with appropriate safety constraints, human oversight, and domain-specific validation. We view this work as a diagnostic and alignment-oriented step toward more reliable multimodal agents, rather than that current systems are ready for autonomous real-world physical action.

M Use of LLMs

In this work, LLMs are used strictly for research support rather than as sources of substantive content. Their use falls into three categories: (i) serving as automatic pipeline annotation helper, (ii) providing tested results on MM-CreativityBench, and (iii) assisting with language refinement during paper writing. For writing support, we used ChatGPT solely to polish text (improving coherence and grammar) while all ideas, logic, results, and technical contributions originate from the authors. To safeguard rigor, we have carefully reviewed all LLM-refined texts to confirm that no hallucinated content was introduced and that the original arguments, findings, and perspectives were faithfully preserved.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction match the paper's actual scope: introducing MM-CreativityBench, evaluating LMMs on grounded creative tool repurposing, and showing that affordance-grounded alignment improves exploration and part-level grounding. The paper is careful to frame the task as constrained, visually grounded creativity rather than creativity in general.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses several important limitations and scope clarifications, including the use of synthetic images, the artificial but diagnostic interaction protocol, the single-gold evaluation design, and the fact that the benchmark targets constrained tool repurposing rather than all forms of creativity. These points are mainly discussed in Appendix A.2 and the discussion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present theoretical theorems or formal proofs. Its contributions are benchmark construction, empirical evaluation, and alignment experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the main information needed to understand and reproduce the experiments, including the benchmark construction pipeline, interactive evaluation protocol, metrics, model families, training data statistics, prompts, and SFT/DPO training details. The appendix gives additional details on task construction, trajectory generation, prompts, and hyperparameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is included in the supplementary material, together with the exact evaluation scripts and training configurations needed to reproduce the experiments. We will also open-source the code upon acceptance of the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the evaluation protocol, dataset sizes, train/test split separation, evaluated models, decoding setting, metrics, and training setup. The appendix further reports SFT and DPO details such as LoRA rank, optimizer-related settings, learning rates, batch/accumulation settings, etc.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The paper conducted significance tests for the major claims regarding both the benchmarking results and the improvements brought by SFT+DPO, as reported in Appendix G.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports important compute details for training in Appendix G, including the GPU setup, BF16 precision, FlashAttention-2, and training time.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The paper discusses Code of Ethics in Appendix K.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses Broader Impact in Appendix L.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk models, scraped personal datasets, or unrestricted image generators.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper explicitly cites when using existing assets, including an open-source affordance knowledge base, external LMMs, image-generation models, and prior benchmark/task sources.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper introduces MM-CreativityBench and documents its construction process, data statistics, task format, interaction protocol, prompts, training trajectories, and evaluation metrics. For the final version if sccepted, the released asset package will also include a clear dataset/model card, license, and usage documentation.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper discusses LLM Usage in Appendix M.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.