Investigating the Role of Instruction Variety and Task Difficulty in Robotic Manipulation Tasks

Amit Parekh, Nikolas Vitsakis, Alessandro Suglia, Ioannis Konstas

Heriot-Watt University

{amit.parekh, nv2006, a.suglia, i.konstas}@hw.ac.uk

Abstract

Evaluating the generalisation capabilities of multimodal models based solely on their performance on out-of-distribution data fails to capture their true robustness. This work introduces a comprehensive evaluation framework that systematically examines the role of instructions and inputs in the generalisation abilities of such models, considering architectural design, input perturbations across language and vision modalities, and increased task complexity. The proposed framework uncovers the resilience of multimodal models to extreme instruction perturbations and their vulnerability to observational changes, raising concerns about overfitting to spurious correlations. By employing this evaluation framework on current Transformer-based multimodal models for robotic manipulation tasks, we uncover limitations and suggest future advancements should focus on architectural and training innovations that better integrate multimodal inputs, enhancing a model's generalisation prowess by prioritising sensitivity to input content over incidental correlations.[1]

1 Introduction

Designing artificial agents to follow natural language instructions—to understand and act within the context of their environment—is a long-term goal of artificial intelligence (Winograd, 1972). An artificial agent should generalise to unseen scenarios by combining concepts and skills underpinning its training data in novel ways (Lake et al., 2017).

Previous work which proposed several language-guided tasks for tackling this challenge, largely focused on generalising to environments with different scenes from the training ones (e.g., ALFRED; Shridhar et al., 2020). However, relying solely on language for embodied action execution tasks can be inefficient, especially in collaborative settings with high ambiguity, such as visually cluttered scenes (Chiyah-Garcia et al., 2024, 2023; Li et al., 2023). Multimodal prompts—instructions which interleave vision and language tokens—represent a way to specify commands which can be more flexible and specific than can be explained using text only (Jiang et al., 2023; Ma et al., 2024; Stone et al., 2023). This capability is crucial for realistic human-robot collaboration tasks and can be viewed as analogous to pointing at objects within a scene (Chen et al., 2021; Islam et al., 2022). For this reason, Jiang et al. (2023) presented VIMA-BENCH, the first benchmark aimed at studying several axes of generalisation involving novel concepts and tasks, with models receiving instructions combining both language and visual referents.

Many other benchmarks test for generalisation by solely looking at held-out examples (Open X-Embodiment Collaboration, 2024; Stone et al., 2023). However, as highlighted by Hupkes et al. (2023), generalisation should be evaluated across multiple dimensions when creating truly robust models, capable of performing safely in varied environments. Inspired by these ideals, we assess generalisation along key axes such as structural, compositional, and robustness through specific covariate shifts (i.e., *input perturbations*) as outlined in Figure 1. Specifically, we looked at 1) an extensive set of linguistic perturbations on instructions, such as paraphrasing, corrupting the language content, and replacing visual referents with language descriptions; 2) masking entire modalities within instructions; 3) introducing visual perturbations by permuting object order; and 4) increasing the difficulty of the tasks (e.g., placing distractors between source and target). We categorise each perturbation as either *plausible* (e.g., paraphrases) or *unrealistic* (e.g., nonsensical instructions). We expect models to be robust to plausible inputs while dropping performance when faced with unrealistic inputs.

To implement this formalisation, we use VIMA-BENCH which, unlike other state-of-the-art bench-

---

Perturbations	Instruction	Observations	Plausible?
Just text	Put the letter R with blue and red stripes onto th…		✔
Interleaving modalities	Put the ▣ on the ▦ .		✔
Paraphrases	Drop the ▣ on top of the ▦ .		✔
Gobbledygook Words	HVi tSn ▣ RVe Fmot ▦ .		✘
Gobbledygook Tokens	tablet protocols ▣ representation Panasonic ▦ .		✘
Mask language tokens	▣ ▦		✘
Mask visual tokens	Put the on the.		✘
Mask entire instruction			✘
Distracting Difficulty	Put the ▣ on the ▦ .		✔
Extreme Difficulty	Put the ▣ on the ◉ .		✔
Extremely Distracting Difficulty	Put the ▣ on the ◉ .		✔
Order Permutation*	Put the ▣ on the ▦ .		✔

Figure 1: Our evaluation framework. Each perturbation affects the instruction or observation inputs, which can be linguistic, visual, or a combination of both. The plausibility of a perturbation relates to a model's expected performance. Sensitivity to unreasonable conditions (✘) indicates that a model should not perform the task successfully given the perturbation, while plausible perturbations (✔) suggest that it should still perform successfully.

marks such as ALFRED (Shridhar et al., 2020), CLIPort (Shridhar et al., 2022), ARNOLD (Gong et al., 2023), and Ravens (Zeng et al., 2021), provides several advantages: 1) it covers the majority of robotic manipulation tasks, 2) it offers more fine-grained levels for assessing the systematic generalisation of models; and 3) it represents a benchmark that allows for careful examinations of specific architecture and training regimes. For this reason, this paper builds on the controllability of VIMA-BENCH to extensively study the impact that properties of multimodal prompts and visual representations have on model performance.

We applied our novel evaluation setup on multiple state-of-the-art architectures commonly used for different Embodied AI tasks and datasets (Jiang et al., 2023; Octo Model Team, 2023; Open X-Embodiment Collaboration, 2024; Reed et al., 2022; Shridhar et al., 2022; Zhao et al., 2023).[2] We uncover several deficiencies of current "generalist agents" including 1) insensitivity to language perturbations, as they still perform several tasks when provided with gibberish instructions; and 2) inability to handle tasks of increasing difficulty, poten-

tially including more distractors in the visual scene. These findings aim to shed light on state-of-the-art model performance and call for more research on systematically assessing model robustness with adequate tasks and settings that are indicative of the generalisation capacities of Embodied AI models designed to safely and effectively complete tasks in the real world in collaboration with humans.

2 Related Work

Language-driven Embodied AI Embodied AI focuses on designing agents that are embodied in some environment (simulated or real) and generates actions to complete a given task, whose objective is typically specified in natural language (Das et al., 2018). Tasks for Embodied AI have been formulated in different ways depending on the degree of complexity of the action space. For example, Vision+Language Navigation (VLN; Anderson et al., 2018; Thomason et al., 2020) requires agents to generate navigation actions to follow natural language instructions and reach some destination in the environment. With more sophisticated 3D simulated environments such as AI2Thor (Kolve et al., 2017), more recent works also define several tasks involving object interaction (e.g., Gao et al., 2023; Shridhar et al., 2022, 2020; Stone et al., 2023).

---

[2]While feasible, we refrain from applying our evaluation framework on larger Vision and Language Models (VLMs) such as LLaVa (Liu et al., 2023), as we focus on models of a similar size to VIMA, which are amenable to on-device processing when deployed on real-world robots.

Language in Robotic Manipulation Tasks Language plays a crucial role in many Embodied AI tasks, providing an interface for task learning (Laird et al., 2017), with many Embodied AI tasks that require language instructions which are typically hand-crafted via templates (e.g., VIMA-BENCH, CLIPort, ARNOLD) or crowd-sourced (e.g., ALFRED). However, benchmarks often focus on evaluating generalisation using held-out episodes (Open X-Embodiment Collaboration, 2024) and do not thoroughly evaluate the importance of language (Octo Model Team, 2023; Open X-Embodiment Collaboration, 2024; Stone et al., 2023). For instance, models trained on ALFRED have shown to be insensitive to language instructions (Akula et al., 2022), while nonsensical instructions have even improved downstream performance on the VLN benchmark (Zhu et al., 2023).

We focus on tabletop robotic manipulation tasks with natural language instructions to measure performance on a well-scoped action execution task. This allows for assessment of visual grounding capabilities from grounding instructions in the real world, while also removing the extra complexity of sophisticated skills (e.g., SLAM) required for navigation tasks (Anderson et al., 2018) or the need to predict fine-grained joint control by relying on inverse-kinematics (Ma et al., 2024; Octo Model Team, 2023; Open X-Embodiment Collaboration, 2024; Zeng et al., 2021).

Assessing Generalisation and Robustness Embodied AI systems must generalise to any complex and novel tasks they might face (Duan et al., 2022), making robustness a highly-desired characteristic in models, illustrating how well they can ignore spurious correlations and generalise to new domains and tasks (Akula et al., 2022; Gong et al., 2023; Hupkes et al., 2023). Embodied AI benchmarks often assess generalisation through seen/unseen scenes (e.g., Gao et al., 2023; Shridhar et al., 2020; Zheng et al., 2022), assuming that all tasks the agent must complete and the objects the agent must interact with are fully specified at training time. While recent benchmarks evaluate models on unseen objects and scenes (Gong et al., 2023; Open X-Embodiment Collaboration, 2024; Stone et al., 2023), there is no notion of systematic or compositional generalisation to new concepts, affordances (Pantazopoulos et al., 2022; Suglia et al., 2020), or novel tasks (Chung et al., 2022).

Although models trained on realistic simulations can transfer learned behaviours to real-world environments (Octo Model Team, 2023; Open X-Embodiment Collaboration, 2024), they remain sensitive to distributional shifts in visual inputs (Li et al., 2024), an issue that persists even when training data includes perturbations (Pumacay et al., 2024). Furthermore, while models can adapt to relocated targets, they struggle with mid-trajectory linguistic shifts, such as swapping directions from "left" to "right" (Anwar et al., 2024). Our work extends these findings by examining model behaviour under extreme instruction perturbations, providing insights into how models handle challenging and unconventional scenarios.

3 Experimental Setup

Evaluation Data We use VIMA-BENCH to compare model performance across various skills, tasks, and levels of systematicity, as it is best suited for evaluating the role instructions play in generalising from multimodal prompts.[3] Specifically, we assess the compositional generalisation capabilities at four distinct levels of systematicity (Hupkes et al., 2020; Pantazopoulos et al., 2022): object pose sensitivity (L1), combinatorial generalisation (L2), novel objects (L3), and novel tasks (L4). See Appendix B for environment and evaluation details.

Models We compare four model architectures: encoding visual representations with either object-centric or image-patches; and conditioning prompts on the state through either cross-attention or concatenation (Ma et al., 2024). All models are trained on multimodal instructions with interleaved visual and linguistic features. Multimodal instructions are encoded through a frozen pretrained T5 language model (Raffel et al., 2020), where encoded visual features are injected into the embedding space of the language model (Driess et al., 2023; Ma et al., 2024; Tsimpoukelli et al., 2021). Visual features are implicitly encoded through embedding image frames per observation—more adaptable, more efficient, and outperforming explicit symbolic representations (Gadre et al., 2022; Song et al., 2024). For each observation, the model predicts an action defining a linear movement between two end-effector poses in $\mathbf{SE}(3)$—each representing position and rotation in 3D space. See Appendix A for further training and implementation details.

---

[3]Jiang et al. (2023) did not release a reproducible benchmark; Appendix C.1 details how we remedied this issue.

	L1	L2	L3	L4
(a) Trained on Original; Evaluated on Original				
Cross-Attn + Obj-Centric	**79.3**	**78.8**	<u>72.3</u>	<u>48.6</u>
Cross-Attn + Patches	63.0	62.0	44.9	13.9
Concatenate + Obj-Centric	**79.2**	**78.8**	77.1	49.2
Concatenate + Patches	68.0	66.3	52.9	23.4
(b) Trained on Original; Evaluated on Paraphrases				
Cross-Attn + Obj-Centric	**78.6**	**77.6**	69.8	47.1
Cross-Attn + Patches	61.1	58.5	45.3	16.8
Concatenate + Obj-Centric	<u>71.5</u>	<u>72.2</u>	<u>62.7</u>	<u>43.0</u>
Concatenate + Patches	61.3	57.0	46.0	20.5
(c) Trained on Paraphrases; Evaluated on Original				
Cross-Attn + Obj-Centric	**82.7**	**81.8**	77.4	48.0
Cross-Attn + Patches	63.9	63.0	49.5	20.4
Concatenate + Obj-Centric	<u>80.4</u>	<u>78.2</u>	<u>74.8</u>	**49.0**
Concatenate + Patches	67.1	62.8	52.0	19.8
(d) Trained on Paraphrases; Evaluated on Paraphrases				
Cross-Attn + Obj-Centric	**77.4**	77.5	70.8	48.6
Cross-Attn + Patches	62.2	61.0	45.7	16.1
Concatenate + Obj-Centric	<u>68.8</u>	67.2	59.6	46.0
Concatenate + Patches	67.2	<u>67.8</u>	<u>60.5</u>	<u>46.9</u>

Table 1: Average success rate per level for each model when trained or evaluated on either original or paraphrased multimodal instructions.

4 The Evaluation Framework

We systematically perturb model inputs at test time to investigate the importance of visual and linguistic information in multimodal prompts. This approach helps us understand how input characteristics contribute to a model's task comprehension. Full per-task results are reported in Appendix F.

4.1 Substitutivity in Instructions

We explore how resilient models with multimodal prompts are to substitutivity (Hupkes et al., 2020) by comparing performance on paraphrased instructions: a meaning-preserving operation. We expect robust models to perform similarly to these *plausible* inputs.[4] We also replace visual referents with textual descriptors to assess how models map visuals to the language embedding space.[5]

Baseline Table 1a shows model performance for each combination of prompt-conditioning method and visual encoder when trained and evaluated using the default instructions from VIMA-BENCH.

[4] Appendix E.2 details how paraphrases were created.
[5] We focus on single-object referents, excluding scene or frame referents. See Appendix D.5 for implementation details.

The best-performing approach uses cross-attention to condition prompts on object-centric observations, outperforming image patches. Performance for each model is similar at L1–2 but worsens at L3–4, indicating their inability to generalise to new objects and tasks.

Evaluating on Paraphrases Table 1b shows that models trained on original instructions are predominantly robust to substitutivity in instructions; however, models do exhibit a small performance loss with cross-attention affected less than those using concatenation. This robustness to paraphrased instructions likely stems from using T5 (Raffel et al., 2020) as the frozen pretrained language model (Tsimpoukelli et al., 2021); Raffel et al. (2020) demonstrate that T5 exhibits strong performance on GLUE/SuperGLUE (Wang et al., 2019a,b). The lack of syntactic or lexical diversity in the VIMA-BENCH inputs, suggests that the models might overfit to the surface form rather than learning to generalise to new sentences.

Training on Paraphrases Table 1c and 1d show that training on linguistically-diverse instructions can improve performance for models that use cross-attention to condition prompts or use object-centric visual features. However, performance worsens when evaluated on paraphrased instructions. Taken together, this suggests that training on diverse instructions helps models better connect the semantics of a multimodal instruction over its surface form, aiding in generalisation to novel scenes. However, poor performance on L3 shows that models struggle more with unseen objects. Furthermore, using image patches and concatenation performs better when trained and evaluated on diverse instructions over any training/evaluation condition. This suggests that these architectures are more resilient to unseen objects and unseen tasks allowing for better generalisation in more complex settings.

Replacing Visual Referents with Descriptors Table 2 shows object-centric models perform comparably when replacing objects with natural language descriptors, suggesting that models have learned to map visual features within the language model's embedding space (Driess et al., 2023; Tsimpoukelli et al., 2021). Furthermore, cross-attention outperforms concatenation, indicating it better preserves relationships between natural language descriptors and visual referents. Additionally, both models that use image patches perform

	L1	L2	L3	L4
With Visual Referents*				
Cross-Attn + Obj-Centric	<u>87.3</u>	**86.4**	<u>75.8</u>	**49.2**
Cross-Attn + Patches	78.3	77.5	54.6	17.8
Concatenate + Obj-Centric	**88.9**	**86.4**	**81.2**	48.5
Concatenate + Patches	79.9	75.1	55.0	15.2
Replace Visual Referents with Descriptors*				
Cross-Attn + Obj-Centric	**87.9**	**87.2**	**73.3**	**49.0**
Cross-Attn + Patches	46.8	44.7	38.2	25.5
Concatenate + Obj-Centric	<u>79.4</u>	<u>78.1</u>	<u>70.0</u>	<u>38.5</u>
Concatenate + Patches	56.4	50.6	52.0	25.8

Table 2: Average success rate per level when visual referents are replaced with textual descriptors during evaluation only. Models trained on paraphrased multimodal instructions—using visual referents. **Not all tasks included; see Appendix D.5 for details.*

notably worse on L1–3. When using patches, all visuals provided to the model in the prompt—be it a single object or a frame—are encoded into a fixed number of patches, whereas object-centric methods encode one object per token. Due to the cardinality of this mapping, the former is a more difficult task than the latter.

4.2 Perturbations of Instruction Syntax

We introduce two methods to distort language within a multimodal prompt: *Gobbledygook Words* and *Gobbledygook Tokens*. As shown in Figure 2, each method removes information from the language modality differently without affecting the visual referents. *Gobbledygook Tokens* preserves the tokenised sequence length, while *Gobbledygook Words* maintains the word count but increases the tokenised sequence length (see Appendix D.4 for implementation details). As *Gobbledygook* perturbations are unrealistic, we expect performance to plummet to near-random chance.[6] Furthermore, while the *Gobbledygook* perturbations removes signal from the linguistic channel, it does not remove text from the instruction. While irrelevant to the task, they are still provided to, and considered by, the language model. To investigate the contribution of each modality further, we compare their individual impact on the overall model performance.

***Gobbledygook* Perturbations**   Table 3 shows that *Gobbledygook* perturbations degrade performance across architectures, but not to random chance, implying that models rely on other cues to infer tasks

---

[6]We derive random chance in Appendix D.1.

Figure 2: Illustration of language perturbations **challenging model sensitivity to language content in multimodal instructions**: *Gobbledygook Words* (random characters, increased token length) and *Gobbledygook Tokens* (random words, same sequence length).

	L1	L2	L3	L4
Without Gobbledygook*				
Cross-Attn + Obj-Centric	**82.7**	**81.8**	**77.4**	48.0
Cross-Attn + Patches	63.9	63.0	49.5	20.4
Concatenate + Obj-Centric	<u>80.4</u>	<u>78.2</u>	<u>74.8</u>	**49.0**
Concatenate + Patches	67.1	62.8	52.0	19.8
Gobbledygook Tokens				
Cross-Attn + Obj-Centric	**56.7**	<u>54.5</u>	<u>36.6</u>	<u>22.9</u>
Cross-Attn + Patches	45.2	45.9	34.0	15.1
Concatenate + Obj-Centric	**56.7**	**55.3**	**45.8**	**26.4**
Concatenate + Patches	45.9	44.3	32.9	20.0
Gobbledygook Words				
Cross-Attn + Obj-Centric	**50.8**	**51.8**	**39.9**	**33.8**
Cross-Attn + Patches	<u>46.7</u>	<u>48.4</u>	33.9	18.6
Concatenate + Obj-Centric	44.8	44.5	<u>35.4</u>	<u>23.9</u>
Concatenate + Patches	44.3	42.7	31.0	19.0

Table 3: Average success per level after applying each *Gobbledygook* perturbation to the original multimodal instructions, showing all models outperforming random chance. Models trained on multimodal paraphrased instructions. **Copied from Table 1c.*

despite nonsensical instructions.

When applying *Gobbledygook Tokens*, object-centric features outperform image-patches, regardless of the prompt-conditioning method used. This implies that object-centric features provide a stronger signal for models to infer the desired task without explicit direction. While we would expect similar performance drops across both perturbation methods, object-centric models exhibit poorer performance with *Gobbledygook Words* compared to *Gobbledygook Tokens*. With *Gobbledygook Words*, conditioning with cross-attention helps models uncover the task at lower levels, but cross-attention with patches struggles more with novel tasks and objects, possibly indicating overfitting. This problem might arise because the decoder uses absolute

	L1	L2	L3	L4
No Tokens Masked*				
Cross-Attn + Obj-Centric	**82.7**	**81.8**	**77.4**	48.0
Cross-Attn + Patches	63.9	63.0	49.5	20.4
Concatenate + Obj-Centric	<u>80.4</u>	<u>78.2</u>	<u>74.8</u>	**49.0**
Concatenate + Patches	67.1	62.8	52.0	19.8
Mask Language Tokens				
Cross-Attn + Obj-Centric	<u>36.3</u>	<u>35.1</u>	19.1	14.8
Cross-Attn + Patches	26.3	26.5	20.7	11.2
Concatenate + Obj-Centric	**39.0**	**39.0**	**28.8**	**25.9**
Concatenate + Patches	30.2	29.0	<u>24.5</u>	16.2
Mask Visual Referents				
Cross-Attn + Obj-Centric	63.6	62.6	**56.4**	**47.9**
Cross-Attn + Patches	<u>64.8</u>	<u>63.0</u>	49.6	20.6
Concatenate + Obj-Centric	59.8	58.9	53.2	<u>47.8</u>
Concatenate + Patches	**67.1**	**63.7**	<u>52.8</u>	23.0

Table 4: Average success rate per level after **masking tokens from one modality** within the multimodal instruction. Models trained on paraphrased instructions and evaluated on original instructions. *Copied from Table 1c.*

	L1	L2	L3	L4
(a) Mistakes Allowed*				
Cross-Attn + Obj-Centric	**82.7**	**81.8**	**77.4**	48.0
Cross-Attn + Patches	63.9	63.0	49.5	20.4
Concatenate + Obj-Centric	<u>80.4</u>	<u>78.2</u>	<u>74.8</u>	**49.0**
Concatenate + Patches	67.1	62.8	52.0	19.8
(b) No Mistakes Allowed				
Cross-Attn + Obj-Centric	<u>70.3</u>	<u>69.7</u>	**67.9**	**46.8**
Cross-Attn + Patches	58.2	57.3	44.2	15.9
Concatenate + Obj-Centric	**72.2**	**71.4**	<u>65.7</u>	<u>45.5</u>
Concatenate + Patches	61.2	57.6	46.0	12.9

Table 5: Average success rate per level when models must solve tasks either **with or without mistakes permitted**. Models trained on paraphrased instructions and evaluated on original instructions. *Copied from Table 1c.*

positional embeddings, which are known to poorly extrapolate to longer sequences (Press et al., 2022; Sun et al., 2022).

Comparing Modalities We investigate whether models rely equally on both modalities by masking one modalities at test time—a perturbation that should significantly decrease performance. Table 4 shows that when masking one modality, performance across all models and levels is above random chance, indicating that models continue to determine the task. Notably, performance suffers more when masking language tokens than when the visual referents are masked. While this indicates that models may rely more heavily on the language content, we would expect that applying *Gobbledygook* perturbations to lead to a comparable drop in performance as masking out the language tokens entirely. Since this is not the case, we instead hypothesise that we can attribute the observed differences to the nature of autoregressive modelling and the order in which modalities are arranged in the instructions. Specifically, as all instructions begin with language tokens, models may struggle with sequences that do not start this way.

4.3 Are Models Relying on Heuristics?

When provided with incomplete instructions, humans often combine available information with heuristics to act rationally in the face of uncertainty (Gigerenzer and Goldstein, 1996; Simon, 1955). Similarly, models may rely on heuristics—combining *any* available information with prior knowledge and world understanding—to infer appropriate actions and complete tasks. Furthermore, when given the opportunity, models may attempt to recover from mistakes through trial and error.

Models Try to Recover from Mistakes Table 5b shows object-centric representations outperform models encoding visuals with image-patches, showing that these models are better at solving the tasks without any errors. However, the performance across all levels and models is lower compared to the more lenient time limit (Table 5a), indicating that given additional time, models explore alternative actions, often successfully. While beneficial, extended time may lead to misleading conclusions when evaluating model performance under unreasonable conditions or nonsensical instructions, as models simply have more time to perform suboptimal action sequences that eventually lead to success. We attribute this behaviour to recovery demonstrations from VIMA-BENCH (see discussion in Appendix B.3.1).

Models Act Without Instructions Table 6 reveals that models continue to perform tasks when instructions are entirely removed, which suggests that models learn to rely on heuristics from observations. Concatenation with object-centric visual features exhibits worse performance, indicating a higher sensitivity to the presence of an instruction, which is a desirable characteristic. Additionally,

	L1	L2	L3	L4
Mask Instructions; Mistakes Allowed				
Cross-Attn + Obj-Centric	**52.4**	**50.5**	**40.6**	**33.0**
Cross-Attn + Patches	26.5	25.8	20.2	11.9
Concatenate + Obj-Centric	30.9	30.5	22.1	14.1
Concatenate + Patches	30.4	29.4	25.0	15.6
Mask Instructions; No Mistakes Allowed				
Cross-Attn + Obj-Centric	**45.2**	**43.9**	**33.1**	**27.4**
Cross-Attn + Patches	19.5	18.7	14.5	7.6
Concatenate + Obj-Centric	7.0	7.2	3.5	2.2
Concatenate + Patches	22.5	22.2	18.2	8.8

Table 6: Average success rate per level with **instructions entirely masked** at test-time. Models trained on paraphrased instructions and evaluated with original instructions before masking. Performing above random chance indicates that the model is using other information solve each task.

cross-attention with object-centric features outperforms concatenation, despite both models using the same visual encoding method. Model performance across all levels is greater when they can recover from errors, suggesting that models will persistently attempt to solve the task if uninterrupted. This behaviour raises important safety concerns: models are acting without clear direction, and yet somehow find the right answer. Furthermore, this difference highlights the effects of how instructions are conditioned on observations, especially when those instructions are masked.

4.4 Task Complexity

As each architecture can infer the correct task without instruction, it implies that they rely on cues solely from the observations, as that is the only other source of input into the model. We test this in two ways: 1) by introducing distractors with the *Distracting* difficulty level, or 2) by increasing task difficulty with the *Extreme* difficulty level. Distractors are objects similar to the target objects in either texture or shape and "task difficulty increases" are specific to each task. The *Extreme* level assesses a model reliance on object affordances when reasoning about actions (Lohmann et al., 2020). These new difficulty levels are *plausible*: agents should be able to disregard unnecessary details and focus on task-critical objects or aspects. Figure 3 provides an example, with further details in Appendix E.1.

Table 7 presents results on our novel evaluation set. Models using patches likely perform poorly due to their inability to represent objects in complex

	L1	L2	L3	L4
(a) Distracting				
Cross-Attn + Obj-Centric	53.8	52.4	46.6	34.8
Cross-Attn + Patches	27.9	27.4	18.2	3.8
Concatenate + Obj-Centric	**60.2**	**59.8**	**53.3**	**39.0**
Concatenate + Patches	29.2	27.0	18.4	5.1
(b) Extreme				
Cross-Attn + Obj-Centric	**53.1**	**53.5**	**55.5**	**36.6**
Cross-Attn + Patches	13.0	12.5	9.7	9.2
Concatenate + Obj-Centric	22.5	23.0	23.2	10.5
Concatenate + Patches	16.2	15.0	12.1	12.1
(c) Extremely Distracting				
Cross-Attn + Obj-Centric	**30.2**	**30.7**	**33.0**	**31.8**
Cross-Attn + Patches	4.5	3.8	2.1	2.9
Concatenate + Obj-Centric	14.6	14.3	10.8	8.5
Concatenate + Patches	5.7	5.2	2.8	4.1
(d) Distracting; Mask Instructions				
Cross-Attn + Obj-Centric	**33.0**	**32.2**	**21.0**	**21.9**
Cross-Attn + Patches	6.7	7.3	2.4	2.6
Concatenate + Obj-Centric	5.9	4.5	1.2	0.6
Concatenate + Patches	6.5	6.7	3.4	1.6
(e) Extreme; Mask Instructions				
Cross-Attn + Obj-Centric	**15.2**	**15.5**	**17.4**	**11.2**
Cross-Attn + Patches	5.7	4.6	2.7	4.6
Concatenate + Obj-Centric	4.5	3.3	2.0	2.2
Concatenate + Patches	3.9	4.1	2.7	7.4
(f) Extremely Distracting; Mask Instructions				
Cross-Attn + Obj-Centric	**10.1**	**8.7**	**8.7**	**10.9**
Cross-Attn + Patches	2.9	2.8	0.4	3.6
Concatenate + Obj-Centric	4.3	4.3	2.0	1.4
Concatenate + Patches	1.7	1.6	0.4	1.2

Table 7: Average success rates **across difficulty levels**. Models trained on paraphrased instructions and evaluated with original instructions *without any mistakes*.

scenes, a known limitation of Transformer-based vision encoders (Darcet et al., 2024; Pantazopoulos et al., 2024). Recent work has proposed several solutions to favour suitable object-centric representation learning (Locatello et al., 2020). While increasing the resolution per patch or image might improve performance (Karamcheti et al., 2024; Liu et al., 2024), it can increase the number of tokens in the decoder, potentially introducing new issues such as increased computational complexity (Lin et al., 2022) or inference time (Firoozi et al., 2024). The *Extreme* difficulty level, which changes expected affordances of objects (e.g., using non-container objects to place objects on) impacts patch-based models more significantly than object-centric models. This indicates that patch-based models are less robust when objects are used in unexpected ways,
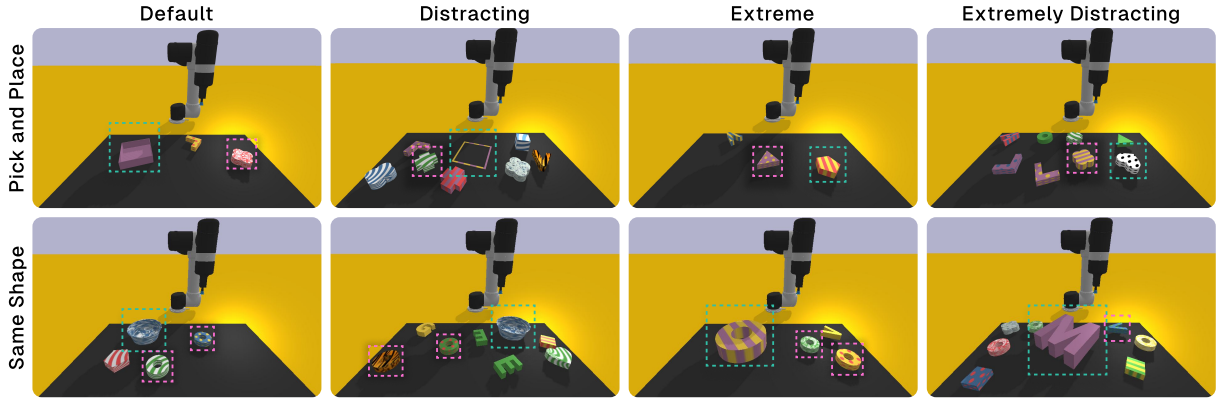
Figure 3: Difficulty level comparisons to default (first column). *Distracting* add visual clutter; *Extreme* changes parameters, complexity, and affordances; and, *Extremely Distracting* combines both. Top row: T1 (*"pick and place into the container"*). Bottom row: T15 (*"place all objects with the same shape as the container into it"*). For illustration purposes, we denote target containers with a green dashed box and target objects with pink dashed box.
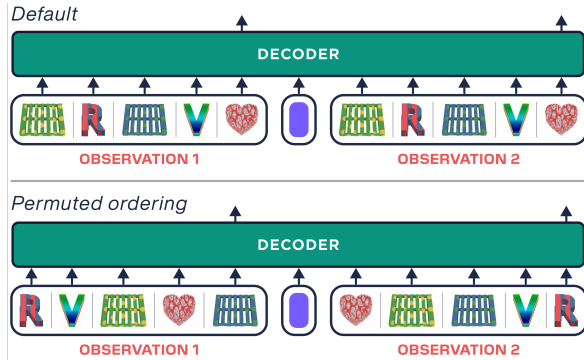
Figure 4: Illustration comparing default and permuted object tokens per observation. In the default ordering (top), tokens in each observation follow the same pattern: the container object first, the target object second, and then any distractor objects. The permuted ordering (bottom) randomises the order differently for each observation in the same sequence.

	L1	L2	L3	L4
Permute Object Tokens; Mistakes Allowed				
Cross-Attn + Obj-Centric	**40.9**	39.1	33.3	11.8
Concatenate + Obj-Centric	40.6	**40.6**	**36.3**	**14.5**
Permute Object Tokens; No Mistakes Allowed				
Cross-Attn + Obj-Centric	24.9	24.6	20.7	5.9
Concatenate + Obj-Centric	**27.6**	**27.8**	**24.8**	**8.2**
Permute Object Tokens; Distracting				
Cross-Attn + Obj-Centric	**14.5**	**14.3**	**12.0**	1.2
Concatenate + Obj-Centric	13.3	12.5	**12.0**	**1.4**
Permute Object Tokens; Extreme				
Cross-Attn + Obj-Centric	**12.0**	**12.7**	**10.8**	6.1
Concatenate + Obj-Centric	7.7	7.3	7.2	**7.1**
Train + Eval with Permutation; Mistakes Allowed				
Cross-Attn + Obj-Centric	59.7	42.1	38.1	**14.4**
Concatenate + Obj-Centric	**70.6**	**49.9**	**44.7**	**14.5**
Train + Eval with Permutation; No Mistakes Allowed				
Cross-Attn + Obj-Centric	50.3	34.1	30.1	**10.0**
Concatenate + Obj-Centric	**58.4**	**41.0**	**34.5**	8.1

Table 8: Average success rate per level when **evaluated with permuted object tokens.** All models are trained with paraphrased instructions and evaluated with original instructions.

while object-centric models adapt better to these changes. At the *Extremely Distracting* difficulty level, patch-based models struggle substantially, indicating their inability to handle both altered object affordances and excessive visual clutter. This decline highlights limitations of patch-based models in complex, yet plausible, scenarios.

As task complexity increases, we expect the model to be increasingly reliant on an instruction to be able to solve the task without error. Tables 7d–f show that with masked instructions, all models except *Cross-Attn + Obj-Centric* plummet, though not to random chance. This suggests that instructions are crucial in more complex settings, there remains some chance that the model may success-

fully solve the task.[7] As *Cross-Attn + Obj-Centric* outperforms all other models without instruction, it suggests that it is using heuristics from the environment to determine and solve the task.

---

[7]See Appendix F.2 for additional analysis into why average performance is above random chance.

4.5 Order Permutations

Object-centric models outperform others, but how they succeed without instruction remains unclear, possibly due to cues from observation encoding. We explore whether permuting the order of object tokens when provided in the model's input affects model performance (see Figure 4 for example permutations). We assume that Transformer-based models using object-centric tokens should be invariant to order permutations (Carion et al., 2020).

Instead, as shown in Table 8, we note that permuting the order of object-centric tokens causes performance on the default difficulty level to half. Exploring how well models perform without the opportunity to recover from mistakes halves this result further. This indicates that when they do not rely on spurious correlations, models try to recover from mistakes until an episode terminates. Further proof of this is that performance degrades as the environment becomes more complex: both with more objects present (*Distracting*) and when various affordances are not as expected (*Extreme*).

Similar to findings from Carion et al. (2020), Transformer-based models are vulnerable to order permutations. When trained on these permutations, model performance improves, however, it is not at the same level as Table 1, suggesting that a considerable proportion of model performance stems from learned spurious correlations.

5 Conclusion

We define an evaluation framework for Embodied AI grounded in the generalisation framework from (Hupkes et al., 2023). Specifically, we assess generalisation across important axes by means of specific multimodal input perturbations including paraphrases, replacing visual referents with descriptors, and manipulating the instruction syntax as well as entire input modalities. We instantiate this evaluation framework in VIMA-BENCH to assess the robustness of state-of-the-art models.

Overall, our findings indicate that while substitutivity can lead to performance gains, language perturbations do not impact performance as expected. To further explore this effect, we evaluate whether models rely on heuristics to complete tasks by removing individual modalities. We show that models perform tasks even without instructions by relying on spurious correlations within observations, as learned during training. We further prove this effect by showing that performance decreases when

the number of objects in an environment increases, and agents can no longer randomly perform the correct sequence of actions.

Taken together, our findings suggest that it is important to define evaluation frameworks like ours that can assess generalisation across multiple axes in order to have a more reliable characterisation of the overall model performance. In future work, we aim to apply this evaluation framework systematically to other benchmarks as well to discover important architectural insights that will guide the next generation of Embodied AI models.

Limitations & Ethical Considerations

Limited in Embodied AI This study aims to provide Embodied AI researchers with an experimental evaluation framework for studying generalisation capabilities of robot policies via an extensive set of multimodal input perturbations. We have instantiated this framework using VIMA-BENCH. VIMA-BENCH was created to evaluate robot manipulation tasks in a controlled setting with a focus on compositional generalisation skills. To date, many proposed embodied AI tasks require several skills, such as navigation and manipulation. We focus on manipulation skills as they remove an extra degree of complexity found in navigation tasks that require more sophisticated skills (e.g., SLAM). Further, tabletop manipulation allows us to focus on problems in grounding language instructions in the real world to assess visual grounding capabilities.

The architectures used in this work are also used in more realistic benchmarks (e.g., Open X-Embodiment Collaboration, 2024). Therefore, this provides the possibility to study architectures used for embodied AI tasks under very strict conditions without being influenced by differences in robotic platforms and embodiments.

The main contribution of our paper is to assess to what extent this is true and to shed light on the weaknesses of current Transformer-based action policies. Additionally, we believe that our framework is generic enough to be applied to other datasets considering that it analyses model performance using core concepts of systematic generalisation (Hupkes et al., 2023).

Choice of Perturbations on Visual Observations In this work, we focus primarily on perturbations that directly affect how models make decisions. However, a possible avenue for future work would be to explore how robust models are to other fac-

tors such as camera choice and background colours (Pumacay et al., 2024). In robotic manipulation tasks, the camera's distance from the robot is often constant (Octo Model Team, 2023; Shridhar et al., 2022; Zeng et al., 2021). Changing the camera's position relative to the robot after training would introduce confounds and increase downstream difficulties, unless trained to do so (e.g., Grauman et al., 2024; Pumacay et al., 2024). When deploying models, it is crucial to test them under varying light levels and background colours. Reducing light levels can impede the model's ability to perceive objects. Therefore, using ground-truth segmentation masks in low-light conditions is ecologically invalid; requiring a new model to extract segmentation masks at risk of introducing new confounds and potential issues like sensitivity to light or camera limitations.

Safety Concerns with Embodied AI The aim of Embodied AI is to build artificial agents that can collaborate and enhance the human experience via either offering companionship (Deng et al., 2019; Strohmann et al., 2023) or performing tasks (Duan et al., 2022; Takeda et al., 2019). As explained by Duan et al. (2022), the latter is tested via simulations which attempt to create ecologically valid frameworks to evaluate agent performance before deployment in a real-world setting. Through this lens, the findings shown in this paper are particularly worrisome, as the shortcomings that we describe indicate issues with the evaluation process itself. This could mean that embodied agents previously evaluated as successful in their generalisation capabilities may fail outside of a simulated environment, increasing the chance to harm humans.

While our framework explains how to thoroughly and systematically assess the training and evaluation of an embodied agent, it is important to note that while our exploration is extensive, there are still aspects that fall outside of the scope of this paper. Our future work aims to apply our framework to a wider array of environments. This will allow us to provide the research community with a more systematic evaluation approach aimed at pinpointing edge cases and limitations of Embodied AI systems, paving the way to a more robust solution for Sim2Real transfer.

Acknowledgements

References

Arjun Akula, Spandana Gella, Aishwarya Padmakumar, Mahdi Namazifar, Mohit Bansal, Jesse Thomason, and Dilek Hakkani-Tur. 2022. ALFRED-L: Investigating the Role of Language for Action Learning in Interactive Visual Environments. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9369–9378, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. 2024. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, volume 2 of *ASPLOS '24*, pages 929–947, New York, NY, USA. Association for Computing Machinery.

Abrar Anwar, Rohan Gupta, and Jesse Thomason. 2024. Contrast Sets for Evaluating Language-Guided Robot Policies. *Preprint*, arXiv:2406.13636.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. 2023. RT-1: Robotics Transformer for Real-World Control at Scale. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, volume 12346, pages 213–229. Springer International Publishing, Cham.

Yixin Chen, Qing Li, Deqian Kong, Yik Lun Kei, Song-Chun Zhu, Tao Gao, Yixin Zhu, and Siyuan Huang. 2021. YouRefIt: Embodied Reference Understanding with Language and Gesture. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1365–1375, Montreal, QC, Canada. IEEE.

Javier Chiyah-Garcia, Alessandro Suglia, and Arash Eshghi. 2024. Repairs in a Block World: A New Benchmark for Handling User Corrections with Multi-Modal Language Models. *Preprint*, arXiv:2409.14247.

Javier Chiyah-Garcia, Alessandro Suglia, Arash Eshghi, and Helen Hastie. 2023. 'What are you referring to?' Evaluating the Ability of Multi-Modal Dialogue Models to Process Clarificational Exchanges. In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 175–182, Prague, Czechia. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. *Preprint*, arXiv:2210.11416.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. 2024. Vision Transformers Need Registers. In *The Twelfth International Conference on Learning Representations*.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Embodied Question Answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2135–213509.

Eric Deng, Bilge Mutlu, Maja J Mataric, et al. 2019. Embodiment in socially interactive robots. *Foundations and Trends® in Robotics*, 7(4):251–356.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 2023. PaLM-E: An Embodied Multimodal Language Model. *Preprint*, arXiv:2303.03378.

Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. 2022. A Survey of Embodied AI: From Simulators to Research Tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244.

Yan Duan, Marcin Andrychowicz, Bradly Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. 2017. One-Shot Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

William Falcon and The PyTorch Lightning Team. 2024. PyTorch Lightning. Zenodo.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. 2024. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*.

Samir Yitzhak Gadre, Kiana Ehsani, Shuran Song, and Roozbeh Mottaghi. 2022. Continuous Scene Representations for Embodied AI. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14829–14839, New Orleans, LA, USA. IEEE.

Qiaozi Gao, Govind Thattai, Xiaofeng Gao, Suhaila Shakiah, Shreyas Pansare, Vasu Sharma, Gaurav Sukhatme, Hangjie Shi, Bofei Yang, Desheng Zheng, Lucy Hu, Karthika Arumugam, Shui Hu, Matthew Wen, Dinakar Guthy, Cadence Chung, Rohan Khanna, Osman Ipek, Leslie Ball, Kate Bland, Heather Rocker, Yadunandana Rao, Michael Johnston, Reza Ghanadan, Arindam Mandal, Dilek Hakkani Tur, and Prem Natarajan. 2023. Alexa Arena: A User-Centric Interactive Platform for Embodied AI. *Preprint*, arXiv:2303.01586.

G. Gigerenzer and D. G. Goldstein. 1996. Reasoning the fast and frugal way: Models of bounded rationality. *Psychological Review*, 103(4):650–669.

Ran Gong, Jiangyong Huang, Yizhou Zhao, Haoran Geng, Xiaofeng Gao, Qingyang Wu, Wensi Ai, Ziheng Zhou, Demetri Terzopoulos, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. 2023. ARNOLD: A Benchmark for Language-Grounded Task Learning With Continuous States in Realistic 3D Scenes. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20426–20438.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J. Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. 2024. Ego-Exo4D: Understanding Skilled Human Activity from First- and Third-Person Perspectives. *Preprint*, arXiv:2311.18259.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality Decomposed: How do Neural Networks Generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Dieuwke Hupkes, Mario Giulianelli, Verna Dankers, Mikel Artetxe, Yanai Elazar, Tiago Pimentel, Christos Christodoulopoulos, Karim Lasri, Naomi Saphra, Arabella Sinclair, Dennis Ulmer, Florian Schottmann, Khuyagbaatar Batsuren, Kaiser Sun, Koustuv Sinha, Leila Khalatbari, Maria Ryskina, Rita Frieske, Ryan Cotterell, and Zhijing Jin. 2023. A taxonomy and review of generalization research in NLP. *Nature Machine Intelligence*, 5(10):1161–1174.

Md Mofijul Islam, Reza Mirzaiee, Alexi Gladstone, Haley Green, and Tariq Iqbal. 2022. CAESAR: An Embodied Simulator for Generating Multimodal Referring Expression Datasets. In *Advances in Neural Information Processing Systems*, volume 35, pages 21001–21015.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. 2023. VIMA: General Robot Manipulation with Multimodal Prompts. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 14975–15022. PMLR.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. 2024. Prismatic VLMs: Investigating the Design Space of Visually-Conditioned Language Models. *Preprint*, arxiv:2402.07865.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. 2017. AI2-THOR: An Interactive 3D Environment for Visual AI. *Preprint*, arXiv:1712.05474.

John E. Laird, Kevin Gluck, John Anderson, Kenneth D. Forbus, Odest Chadwicke Jenkins, Christian Lebiere, Dario Salvucci, Matthias Scheutz, Andrea Thomaz, Greg Trafton, Robert E. Wray, Shiwali Mohan, and James R. Kirk. 2017. Interactive Task Learning. *IEEE Intelligent Systems*, 32(4):6–21.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. 2017. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253.

Jiachen Li, Qiaozi Gao, Michael Johnston, Xiaofeng Gao, Xuehai He, Suhaila Shakiah, Hangjie Shi, Reza Ghanadan, and William Yang Wang. 2023. Mastering Robot Manipulation with Multimodal Prompts through Pretraining and Multi-task Finetuning. *Preprint*, arXiv:2310.09676.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. 2024. Evaluating Real-World Robot Manipulation Policies in Simulation. *Preprint*, arXiv:2405.05941.

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2022. A survey of transformers. *AI Open*, 3:111–132.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved Baselines with Visual Instruction Tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. In *Thirty-Seventh Conference on Neural Information Processing Systems*.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. 2020. Object-centric learning with slot attention. *Advances in neural information processing systems*, 33:11525–11538.

Martin Lohmann, Jordi Salvador, Aniruddha Kembhavi, and Roozbeh Mottaghi. 2020. Learning About Objects by Learning to Interact with Them. In *Advances in Neural Information Processing Systems*, volume 33, pages 3930–3941. Curran Associates, Inc.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. 2024. A Survey on Vision-Language-Action Models for Embodied AI. *Preprint*, arXiv:2405.14093.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. 2023. Octo: An Open-Source Generalist Robot Policy.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I. Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J. Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R. Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, and Zipeng Lin. 2024. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *Preprint*, arXiv:2310.08864.

Georgios Pantazopoulos, Malvina Nikandrou, Amit Parekh, Bhathiya Hemanthage, Arash Eshghi, Ioannis Konstas, Verena Rieser, Oliver Lemon, and Alessandro Suglia. 2023. Multitask Multimodal Prompted Training for Interactive Embodied Task Completion. In *Proceedings of the 2023 Conference*

on *Empirical Methods in Natural Language Processing*, pages 768–789, Singapore. Association for Computational Linguistics.

Georgios Pantazopoulos, Alessandro Suglia, and Arash Eshghi. 2022. Combine to Describe: Evaluating Compositional Generalization in Image Captioning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 115–131, Dublin, Ireland. Association for Computational Linguistics.

Georgios Pantazopoulos, Alessandro Suglia, Oliver Lemon, and Arash Eshghi. 2024. Lost in Space: Probing Fine-grained Spatial Understanding in Vision and Language Resamplers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 540–549, Mexico City, Mexico. Association for Computational Linguistics.

Ofir Press, Noah A Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. In *International Conference on Learning Representations*.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. 2024. THE COLOSSEUM: A Benchmark for Evaluating Generalization for Robotic Manipulation. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. 2022. A Generalist Agent. *Transactions on Machine Learning Research*.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2022. CLIPort: What and Where Pathways for Robotic Manipulation. In *Proceedings of the 5th Conference on Robot Learning*, pages 894–906. PMLR.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA. IEEE.

Herbert A. Simon. 1955. A Behavioral Model of Rational Choice. *The Quarterly Journal of Economics*, 69(1):99–118.

Yaoxian Song, Penglei Sun, Haoyu Liu, Zhixu Li, Wei Song, Yanghua Xiao, and Xiaofang Zhou. 2024. Scene-Driven Multimodal Knowledge Graph Construction for Embodied AI. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–14.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. 2023. Open-World Object Manipulation using Pre-Trained Vision-Language Models. In *Proceedings of The 7th Conference on Robot Learning*, pages 3397–3417. PMLR.

Timo Strohmann, Dominik Siemon, Bijan Khosrawi-Rad, and Susanne Robra-Bissantz. 2023. Toward a design theory for virtual companionship. *Human–Computer Interaction*, 38(3-4):194–234.

Alessandro Suglia, Ioannis Konstas, Andrea Vanzo, Emanuele Bastianelli, Desmond Elliott, Stella Frank, and Oliver Lemon. 2020. CompGuessWhat?!: A Multi-task Evaluation Framework for Grounded Language Learning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7625–7641, Online. Association for Computational Linguistics.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A Length-Extrapolatable Transformer. *Preprint*, arXiv:2212.10554.

Mizuki Takeda, Yasuhisa Hirata, Yueh-Hsuan Weng, Takahiro Katayama, Yasuhide Mizuta, and Atsushi Koujina. 2019. Accountable system design architecture for embodied ai: a focus on physical human support robots. *Advanced Robotics*, 33(23):1248–1263.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-Dialog Navigation. In *Proceedings of the Conference on Robot Learning*, pages 394–406. PMLR.

Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal Few-Shot Learning with Frozen Language Models. In *Advances in Neural Information Processing Systems*, volume 34, pages 200–212. Curran Associates, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *International Conference on Learning Representations*.

Terry Winograd. 1972. Understanding Natural Language. *Cognitive Psychology*, 3(1):1–191.

Omry Yadan. 2019. Hydra - A framework for elegantly configuring complex applications. GitHub.

Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. 2021. Transporter Networks: Rearranging the Visual World for Robotic Manipulation. In *Proceedings of the 2020 Conference on Robot Learning*, pages 726–747. PMLR. ISSN: 2640-3498.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2024. Multimodal Chain-of-Thought Reasoning in Language Models. *Transactions on Machine Learning Research*.

Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. 2023. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation.

Kaizhi Zheng, Xiaotong Chen, Odest Jenkins, and Xin Eric Wang. 2022. VLMbench: A Compositional Benchmark for Vision-and-Language Manipulation. In *Thirty-Sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wang Zhu, Ishika Singh, Yuan Huang, Robin Jia, and Jesse Thomason. 2023. Does VLN Pretraining Work with Nonsensical or Irrelevant Instructions? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Hyperparameter	Value
Pretrained Language Model	t5-base (Raffel et al., 2020)
Optimizer	AdamW (Loshchilov and Hutter, 2019)
Dropout	0.1
Weight Decay	0
Gradient Clip Threshold	1.0
Maximum Learning Rate	1e-4
Minimum Learning Rate	1e-7
Warmup steps	7K (896K examples)
Cosine Annealing steps	All remaining steps
Training epochs	10
Total examples seen	6 099 200
Examples per optimizer step	128

Table A.1: Hyperparameters using during model training for each model.

A   Training Details

A.1   Policy Definition

In the environment, models must learn a non-Markovian policy $\pi : \mathcal{P} \times \mathcal{H} \to \mathcal{A}$, which is essential for completing tasks that rely on previous observations (such as tasks 5 and 16). The policy π maps a multimodal instruction $\boldsymbol{p} \in \mathcal{P}$ and a history trajectory of observations and actions $h_t \in \mathcal{H}$ up to some discrete time step t to the two-pose action primitive $a_t = (\mathcal{T}_{\text{start}}, \mathcal{T}_{\text{end}}) \in \mathcal{A}$.

A multimodal instruction $\boldsymbol{p}$ is an ordered sequence $(x_1, \ldots, x_l)$ of length l, where each element x_i can either be a word w_i or a visual representation of an object or frame of a scene v_i. Observations provided to the model are denoted as $o_t \in \Omega$, where t represents the time step of the observation in the sequence.

Each action a_t defines a linear movement between two end effector poses—where the robot arm moves linearly from the start pose $\mathcal{T}_{\text{start}}$ to the end pose $\mathcal{T}_{\text{end}}$ before retracting. Each pose is defined in the special Euclidean group $\textbf{SE}(3)$ and represented as the state vector $(x, y, z, qw, qx, qy, qz)$, where x, y, z are Cartesian coordinates and qw, qx, qy, qz are quaternion components representing the orientation of the end effector.

The history trajectory h_t consists of pairs of past observations and actions up to time step t, with the final element being the observation at time step t. Formally, each history trajectory is structured as $h_t = (o_0, a_0, o_1, \ldots, a_{t-1}, o_t)$. Consequently, the history trajectory space for time step t can be defined as $\mathcal{H} = (\Omega \times \mathcal{A})^t \times \Omega$.

Training objective   Similar to Jiang et al. (2023), the model is trained through behaviour cloning of expert demonstrations (Duan et al., 2017) that minimises a loss function for a trajectory of T actions given by Equation (1):

$$L(\theta) = \frac{1}{T} \sum_{t=0}^{T} \log \pi_\theta(a_t | \boldsymbol{p}, h_t) \qquad (1)$$

Notably, the loss function was modified to prevent the model from being influenced by the trajectory length (Pantazopoulos et al., 2023).

A.2   Implementation Details

To allow for a fair comparison, all model code uses the code provided from Jiang et al. (2023). Various alterations were made to capture metrics and improve performance, however all architectures are identical. Hyperparameters per component follow that stated in Appendix C in Jiang et al. (2023).

Following Brohan et al. (2023) and Jiang et al. (2023), each coordinate of the pose is predicted separately into one-of-n bins. We follow Jiang et al. (2023), where each coordinate per pose is discretised into 50 bins, with the exception of the y-position which is discretised into 100 bins. For each action dimension, the bin width is uniform across the total action space of the environment.

A.3   Training Hyperparameters

To control for possible confounding variables across all models, we use the same training hyperparameters from Appendix D in Jiang et al. (2023) and from the various GitHub issues. We report a comprehensive table of hyperparameters in Table A.1. Across all models that were trained, these hyperparameters were kept constant and no hyperparameter sweeps were performed. All models were trained for 10 epochs and we used the checkpoint created at the end of epoch 10.

Computation Budget   All models were trained using four NVIDIA A100 40GB GPUs, with each run taking approximately 10 hours. Each evaluation run on the environment took approximately 2 hours and did not require the use of any GPUs. Therefore, the total computational budget for this work is 480 GPU hours.

Pretrained Language Model   Following Jiang et al. (2023) and Octo Model Team (2023), we also use the pretrained encoder from t5-base (Raffel et al., 2020) as the pretrained language model that

encodes multimodal instructions. Additionally, following Jiang et al. (2023) and Tsimpoukelli et al. (2021), we unfreeze the last two layers of the T5 encoder during training.

Learning Rate Schedule While our training process is similar to Jiang et al. (2023), preliminary experiments showed that using a cosine annealing learning rate schedule that reduced the learning rate to the end of the 10th epoch performed better than annealing to 17K steps and training the model at 10^{-7} for 5 epochs.

A.4 Training Components from Scratch

Following Jiang et al. (2023), the instruction encoder was the only pretrained component—using `t5-base` (Raffel et al., 2020); *all other components* were trained from scratch.

Segmentation Masks We used the ground-truth segmentation masks during training and evaluation over a trained object detector model because there is minimal performance difference between using a ground truth predictor and one that was trained for the task (Jiang et al., 2023; Octo Model Team, 2023). As a result, this allows us to control for possible confounding variables from propagated errors.

B Environment Details

In this section, we further outline details of VIMA-BENCH from Jiang et al. (2023). Built on top of the Ravens simulator (Zeng et al., 2021), VIMA-BENCH contains 17 tabletop object manipulation tasks to assess the capabilities learned by VLMs through a four-level protocol that evaluates their systematic generalisation capabilities. All models are trained using behavioural cloning from 50K expert demonstrations for each of 13 tasks, with 4 tasks held out for zero-shot evaluation.

B.1 Skills Models Must Learn to Perform

One of the benefits of VIMA-BENCH is that models must learn skills either in isolation or in combination with other skills, which is a desirable capability of intelligent systems (Lake et al., 2017).

1. **Simple Object Manipulation.** Picking up objects from a name or a visual representation, and placing them in specific locations and positions.
2. **Visual Goal Completion.** Manipulating objects to match the scene in the provided frame.

3. **Visual Memory**. After performing actions, remember the previous state of the workspace and perform an action given information from that time.
4. **Visual Reasoning**. Only performing actions on objects that have the same colours/shapes as in the instruction.
5. **One-Shot Imitation**. Imitate the actions necessary to make the workspace look like a given sequence of frames.
6. **Novel Concept Grounding**. The prompt contains unfamiliar words like *"dax"* which are explained through visual referents and used within an instruction similar to multimodal in-context learning (Zhang et al., 2024).

B.2 Different Levels of Generalisation

VIMA-BENCH uses tiers of generalisation levels to enable more precise assessment of a model's capabilities in the environment by testing its adaptability conditions unseen during training that are either object or instruction specific, as described below:

Placement Generalisation (L1) Object poses—starting positions and orientation—are novel. Failure at this level indicates that model learning is not invariant to object poses, and therefore indicates the model is unable to generalise beyond how objects are positioned in training data.

Combinatorial Generalisation (L2) Object shape and texture combinations are novel (e.g., the model has seen either red objects and squares during training, but never a red square). Failure indicates an inability learn and/or combine object-specific information, therefore unable to perform systematicity within the visual scenes.

Novel Object Generalisation (L3) Objects shapes and textures are novel (e.g., the model has never seen blue objects or triangles during training). Failure at this level indicates difficulty in abstracting object-specific information beyond the training corpus.

Novel Task Generalisation (L4) Tasks (including instructions and success criteria) have never been seen. Failure at this level indicates an inability to perform compositional generalisation to combine skills/movements to solve novel tasks.

Task	1	2	3	4	5	6	7	9	11	12	15	16	17
Minimum	1	1	1	1	1	1	1	1	2	1	2	2	3
Maximum	2	3	2	4	7	2	3	3	2	8	4	4	4

Table B.1: Minimum and maximum actions taken to solve each task across all episodes within the training data. Missing tasks (8, 10, 13, 14) do not appear in the training data as they are only seen when evaluating unseen tasks (L4).

B.3 Dataset Preparation for Training

We parse all 664 976 instances across the 13 tasks used for training—as provided by Jiang et al. (2023)—each containing an action trajectory created by a scripted oracle. We create a validation set using stratified sampling such that a total of 50 000 instances across all the tasks are held out.[8] Each instance is prepared for training in advance by tokenizing any natural language and preparing visual features for the model. We release all code used to prepare the dataset as well as the examples for each split, both before and after preprocessing (see Appendix C for more).

B.3.1 Error Recovery Is Not Emergent Behaviour

We analysed the expert trajectories used to train the model from the VIMA-BENCH dataset to determine whether models are only shown the most efficient solution. Table B.1 shows the minimum and maximum number of actions shown to models to solve each task from the given expert trajectories. The minimum number of moves required per task is dependent on the number of objects and parameters for a given episode. They are not identical for all episodes. We found multiple observation-action pairs in several examples, showing that VIMA-BENCH contains expert trajectories that are not always optimal, thereby suggesting that recovering from mistakes is not an emergent behaviour of the models.

C Reproducibility

We are deeply committed to reproducibility in ML research. To this end, we provide a fully reproducible training and evaluation framework at https://github.com/amitkparekh/CoGeLoT.

License   VIMA-BENCH from Jiang et al. (2023), including model code, pre-trained checkpoint, and the VIMA-BENCH environment are licensed under MIT. All artefacts produced from this work will also be released under the MIT license.

Codebase   We are providing our entire codebase—the full, unabridged version we used throughout development, training, and evaluation. This includes implementations for every perturbation, including the *Gobbledygook* perturbations, to encourage use in other evaluation settings and benchmarks.

Training Data   We are releasing all training data, including the exact training/validation splits used, using the process outlined in Appendix B.3. Our codebase includes the methodology for generating these from the original VIMA-BENCH dataset, which did not include pre-defined splits. Additionally, we provide additional datasets with paraphrased multimodal instructions, along with the commands used to create them. For all datasets splits and variations, we provide the pre-processed instances—stripped of unnecessary metadata and with tokenised instructions with T5—that we used to accelerate model training. All datasets are hosted on our Hugging Face repository[9], and we recommend using them with our provided framework.

Model Checkpoints   We provide every model checkpoint used in our evaluation, including checkpoints from earlier training epochs, to facilitate further interpretability experiments and explorations. Table B.1 provides a list of unique IDs for each trained model, along with the architecture used. These IDs can be used to source model checkpoints from our Hugging Face repository[10], or using our provided framework. As mentioned in Appendix A.3, we only evaluate models after completing all 10 training epochs. However, we provide checkpoints created at the end of each epoch to support future work.

Reproducibility   We trained our models using PyTorch (Ansel et al., 2024) and Lightning (Falcon and The PyTorch Lightning Team, 2024), and tracked all dependencies with PDM[11]. We are providing all components, including a Docker image, to facilitate replication. Our experiments were managed using Hydra configuration files (Yadan, 2019),

---

[8]Authors state that they held out 50 000 examples for validation on their GitHub: https://github.com/vimalabs/VIMA/issues/8#issuecomment-1491255242.

[9]https://huggingface.co/datasets/amitkparekh/vima

[10]https://huggingface.co/amitkparekh/cogelot

[11]https://pdm-project.org/

Instruction-style	Instruction Modalities	Prompt-conditioning	Vision Encoder	Shuffled Objects?	Model ID
Original	Text + Visual	Cross-Attention	Object-Centric	False	8lkml12g
Original	Text + Visual	Cross-Attention	Object-Centric	True	ftwoyjb1
Original	Text + Visual	Cross-Attention	Image-Patches	N/A	ln4nrqhg
Original	Text + Visual	Concatenate	Object-Centric	False	bhuja4vo
Original	Text + Visual	Concatenate	Object-Centric	True	wn9jc5l8
Original	Text + Visual	Concatenate	Image-Patches	N/A	efxugme9
Paraphrases	Text + Visual	Cross-Attention	Object-Centric	False	2df3mwfn
Paraphrases	Text + Visual	Cross-Attention	Object-Centric	True	0nsnkaer
Paraphrases	Text + Visual	Cross-Attention	Image-Patches	N/A	ah5btw8w
Paraphrases	Text + Visual	Concatenate	Object-Centric	False	fs5v61mz
Paraphrases	Text + Visual	Concatenate	Object-Centric	True	xb3yttg9
Paraphrases	Text + Visual	Concatenate	Image-Patches	N/A	zby6xk27

Table B.1: Unique ID for each model checkpoint to aid with reproducibility and the conditions they were trained on.

and we are sharing all configurations, commands, hyperparameters, and seeds used. Our codebase is designed to automatically download the required datasets and models from our Hugging Face repositories when run with the provided configurations and commands, mirroring our exact training and evaluation process.

C.1 Discrepancies in Reported Results

Jiang et al. (2023) only provided the code for the model and the dataset did not contain a train-test split. After creating a working codebase, we were unable to reproduce the results reported by Jiang et al. (2023) using the provided model checkpoint. We spent several weeks trying to reproduce the results, including consulting the original authors on their experimental setup, but were unsuccessful in doing so. Table C.2 contains the reported results from Jiang et al. (2023) and our results when running the evaluation on their provided checkpoint. For this comparison, *no new models were trained*. Note that the provided checkpoint uses cross-attention to condition prompts and object-centric visual features. Across all tasks/generalisation levels (with the exception of T3), task success is significantly lower than what was reported. Possible reasons for this difference include:

- Pure randomness as only 200 episodes are sampled per task, and the exact episodes are not compared.
- There may be a different checkpoint provided compared to the paper.
- Possible misunderstandings during re-implementation.

D Evaluation Details

D.1 Estimating Random Chance

The model predicts actions by mapping embedded action tokens to the action space, which consists of 14 coordinates across two $\mathbf{SE}(3)$ poses. Each pose has seven coordinates that predict a discrete bin. There are 50 discrete bins for each axis, except for the y-position which has 100.

To correctly predict a movement, the model must accurately predict 14 coordinates. Assuming each axis is predicted independently, and that the likelihood of choosing each discrete bin per coordinate is equal, the probability of randomly predicting the correct action is $1/(50 \times 12 + 100 \times 2) = 1/800 = 0.125\%$. Assuming each predicted action is i.i.d., for a task requiring t time steps, the probability that a model will randomly succeed is 0.00125^t.

D.2 Sample Size for Computing Task Performance

Jiang et al. (2023) claimed to run each task in the environment for 100 episodes.[12] However, we assume there is some inconsistency in the statement as the reported success rates consist of multiples of "0.5". Furthermore, due to inconsistencies in the environment, the model will not view the same instantiation of each 200 episodes. As a result, we assume that running 200 samples is large enough to fall under the law of large numbers. Li et al. (2023)

---

[12] While not reported in the final manuscript, it was mentioned on their public GitHub repository: https://github.com/vimalabs/VIMA/issues/16#issuecomment-1622973970.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Reported in Jiang et al. (2023)																		
L1	100.0	100.0	99.5	100.0	56.5	100.0	100.0	—	18.0	—	77.0	93.0	—	—	97.0	76.5	43.0	81.6
L2	100.0	100.0	99.5	100.0	54.5	100.0	100.0	—	17.5	—	77.0	93.0	—	—	98.5	75.0	45.0	81.5
L3	99.0	100.0	100.0	97.0	54.5	100.0	99.0	—	17.5	—	90.5	—	—	—	97.5	46.0	43.5	70.4
L4	—	—	—	—	—	—	—	100.0	—	0.0	—	—	0.0	94.5	—	—	—	48.6
From the Provided Checkpoint																		
L1	93.0	93.5	99.5	85.0	49.5	93.5	95.5	—	14.5	—	90.5	96.0	—	—	5.0	43.5	3.0	66.3
L2	92.0	93.0	100.0	89.5	55.0	91.5	91.0	—	16.0	—	84.0	95.5	—	—	7.0	40.5	0.5	65.8
L3	91.5	94.5	99.5	83.0	51.5	87.0	90.5	—	20.0	—	93.5	—	—	—	6.0	35.5	2.0	62.9
L4	—	—	—	—	—	—	—	80.0	—	2.0	—	—	0.0	4.5	—	—	—	21.6

Table C.2: Comparing the average success rate per task as reported by Jiang et al. (2023) with our results obtained from running the checkpoint provided in the environment. Each task was run for 200 samples.

also sampled 200 episodes for each task during evaluation on VIMA-BENCH.

D.3 When Does an Evaluation Episode End?

During the online evaluation, the episode ends when one of two conditions are met:

1. the model has successfully completed the instruction with the previous action it took; or,
2. the model has not successfully completed the instruction within a maximum of 10 actions.

A maximum length of 10 actions is longer than the default length used by Jiang et al. (2023).

D.4 *Gobbledygook* Perturbations

We outline how *Gobbledygook Words* and *Gobbledygook Tokens* manipulate multimodal instructions to remove all linguistic information without altering the positions of any visual referents.

Gobbledygook Words Let $w_i = (c_1, c_2, \ldots, c_j)$ represent a word with j characters, where each character is from a set $\mathbb{A}$ containing all uppercase and lowercase alphabetical English characters. Given a multimodal prompt p of multiple words, we transform the sequence by: first replacing each character per word with a random choice from $\mathbb{A}$, then randomly swap the positions of words within the sequence without changing the position of any visual representations within the sequence.

Gobbledygook Tokens This method transforms the multimodal prompt by randomising each subword unit after tokenizing the instruction with any other token from the vocabulary such that the number of sub-word units is the same as the original

	# Words	# Tokens
Original Instruction	12.9 ± 7.6	20.2 ± 13.6
Gobbledygook Tokens	15.2 ± 9.3	20.2 ± 13.6
Gobbledygook Words	12.9 ± 7.6	49.7 ± 27.8

Table D.1: Average length of instructions (with standard deviation), both before and after transforming through a language perturbation method. A single word is defined as sequences of alphanumeric characters delimited by a whitespace character. Tokens are defined as the number of IDs returned from the tokenizer.

instruction. See Figure 2 for an example where an instruction perturbed with *Gobbledygook Tokens* does not contain any information in the language modality pertaining to the original task.

Controlling for sequence lengths To avoid introducing additional difficulty into the tasks, we ensure that the length of the instruction is identical to before perturbing for either natural language words or the tokenised form. Table D.1 further verifies this as the number of words in an instruction does not change for *Gobbledygook Words*, and the number of tokens does not change for *Gobbledygook Tokens*. It also allows for checking whether or not the length of the instruction in natural language has any impact on model performance.

As illustrated in Figure 2, *Gobbledygook Words* ensures that the number of characters and "words" within the multimodal prompt—and the number of words between each visual placeholder—does not change. However, the average length of the prompt after tokenizing has increased because T5 uses a SentencePiece tokenizer that was trained on natural language text (Raffel et al., 2020).

19408

1. IPy oXV {dragged_obj_1} xzcl TzN {base_obj}.

2. trP EZDn {dragged_obj_1} nGY Dtx {base_obj}.

3. cOfl Piw {dragged_obj_1} GGF KgN {base_obj}.

4. URs ljzF {dragged_obj_1} pJF WWY {base_obj}.
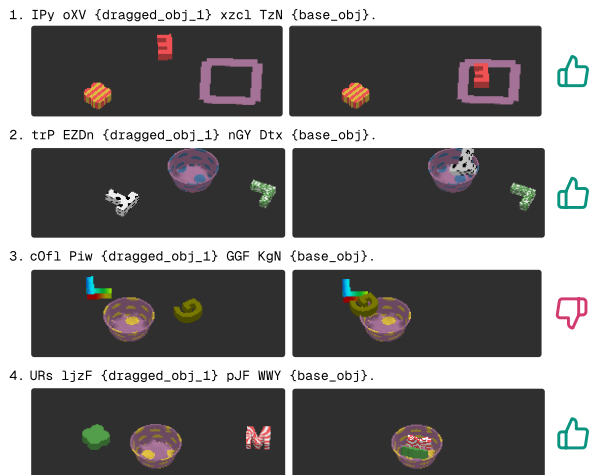
Figure D.1: In-environment observations seen by the model, showing task performance when using *Gobbledygook Words*. Instructions given to the model are shown on top of the images, with the images themselves showing different iterations of either success (see 1, 2, and 4) or failure (see 3).
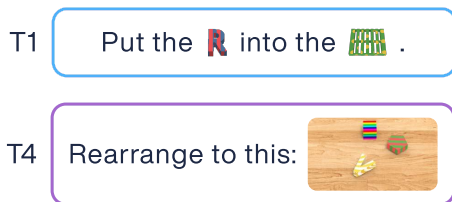
Figure D.2: Example instruction for T1 (pick and place) and T4 (rearrange to this scene).

In-environment examples after applying *Gobbledygook Words* Figure D.1 contains some examples where the model still succeeds in performing the task, even when provided with perturbed language from *Gobbledygook Words*. From Figure D.1, Examples 1 and 2 both show that the model followed through on incomprehensible instructions and successfully performed the tasks of: identifying the task to perform with the stated object from a choice of two, picking it up, and putting it into a destination. Example 4 indicates interesting behaviour as the model continued to place all objects into the container to end the episode.[13] Such a failure is indicated in Example 3, where the model picked the object and placed it onto the receptacle in a way that resulted in a scenario it could not recover from, having chosen the wrong object to place and by balancing it on the edge of the container.

---

[13]We outline the termination conditions for a given episode in Appendix D.3.

D.5 Which Visual Referents Can Be Substituted as Text?

There are two types of visual referents that appear in VIMA-BENCH: ones that refer to a single object, and ones that represent an object *within a scene*. For example, as shown in Figure D.2, T1 directly refers to an object whereas T4 directly includes a frame of a scene. As a result, it does not make sense to convert tasks that include frames or scenes in their instruction as the textual description can refer to more than necessary. In total, 9 of the 17 tasks (across all 4 generalisation levels) use instructions that do not use frames.

E Extensions to VIMA-Bench

In this work, we propose multiple extensions to VIMA-BENCH. In this section, we provide further analysis and details for each.

E.1 Increasing Difficulty Across All Tasks
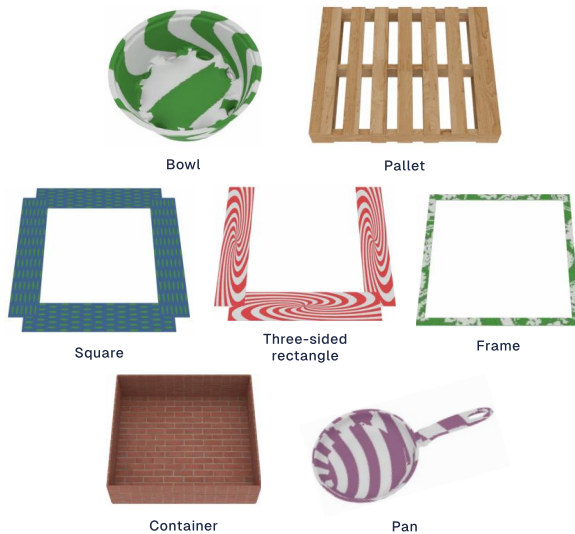
Figure E.1: Objects within VIMA-BENCH that are often regarded as "containers"; i.e., other objects are always placed within these.

Table E.1 outlines the changes made for each difficulty level for each task. The *Distracting* difficulty level focuses on drastically increasing the number of distractors in the scene to try and confuse the model, whereas the *Extreme* difficulty level alters the parameters of the task to check whether a model is over-reliant on the parameters seen during training. Additionally, a subset of objects in VIMA-BENCH is always used as "containers" (Figure E.1): objects are always put into/onto them across all tasks. Therefore, as part of the *Extreme*

difficulty, the container/destination object is just any other acceptable object (within the generalisation level constraints) that is not one of these.

E.2 Paraphrasing Multimodal Instructions

We created paraphrases by manually inspecting the instructions and using meta-templates to construct variations. Notably, we were careful to avoid introducing ambiguity that could introduce any misunderstanding into the semantic meaning of the instruction. As a result, only the natural language words are altered; any novel words (as in T6–8) remained unchanged. The observations seen, the actions the model must perform, and the instances for each train-valid-test split are unchanged. We provide examples of some paraphrased alternatives of the original instruction in Table E.2. All meta-templates used for each task are included within the provided source code.

	Description	Distracting	Extreme
T1	Put specified objects into specified container.	Distractors: 1 → 6	Containers are now one of the draggable objects instead of the designated container shapes
T2	Place objects with specified texture *from the given frame* into container with specified colour.	Distractors in frame: 1 → 3. Distractors in workspace: 1 → 3	Containers are now one of the draggable objects instead of the designated container shapes
T3	Rotate the specified object by the given number of degrees.	Distractors: 1 → 8	Possible Angles of rotation: From [30, 60, 90, 120, 150] to [20, 40, 60, 80, 100, 120, 140, 160]
T4	Look at the objects within the frame and move the objects in the workspace to those positions. Distractors in the workspace may need to be moved out the way. Not all objects in the workspace are visible in the frame.	Distractors in workspace: 2 → 3	Distractors in workspace will ALWAYS be in the way (therefore the model must move them out the way to complete the task)
T5	Perform T4, and then put all the objects back to the start	Distractors in workspace: 2 → 3	Distractors in workspace will ALWAYS be in the way (therefore the model must move them out the way to complete the task)
T6	Compare the size or texture saturation of objects and make adjustments to the specified object(s) accordingly.	Distractors: 1 → 3	All container shapes are replaced with other shapes. Adjective word choices are now: "xachup", "feplicat", "gazip", or "duchat".
T7	Apply novel words to two objects (one is a container class), and put one object into the container.	Distractors: 1 → 3	All container shapes are replaced with other shapes. Noun word choices are now:
T8	Combination of T6 and T7	Combination of T6 and T7.	All container shapes are replaced with other shapes.
T9	Determine the degrees to rotate an object from three before/after demonstrations (i.e., 3-shot demonstration to learn the task)	Total number of objects: 3 → 8	Possible Angles of rotation: [30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330] → [20, 40, 60, 80, 100, 120, 140, 160]
T10	Follow motions for specific objects from demonstrations of frames	Distractors in workspace: 1 → 3. Distractors in frames: 1 → 3	Possible motion points: 5 → 10
T11	Stack objects with the order illustrated in given frames.	Distractors in workspace: 1 → 3	Objects in workspace: 3 → 5
T12	Sweep the objects into the region **without exceeding** the boundary	Objects in the scene 1–5 → 6–10	Sweepable objects are now any dragged object
T13	Sweep the objects into a region **without touching** the constraint.	Objects in the scene 1–5 → 6–10	Sweepable objects are now any dragged object
T14	Pick all objects in the workspace with the same texture as the container object specified in the prompt, into it.	Distractors: 1 → 5	All container shapes are replaced with other shapes.
T15	Put all objects in the workspace with the same top-down profile goal container into it.	Distractors: 1 → 5	All container shapes are replaced with other shapes.
T16	Put the target object into the container, and then put one of its old neighbours into the same container	Distractors: 1 → 3	Density grid of objects: 3×3 → 4×4
T17	Pick and place the object into different containers in order then restore to the initial container.	Distractors: 0 → 4	All containers can be different types of shapes

Table E.1: Descriptions of each task, number of distractors added to increase difficulty, and description of the extreme difficulty for each.

Task	Original	Alternative
1	Put the `blue` `spiral` object in {scene} into the wooden object.	From the {scene} stack the `blue` `spiral` object on the wooden thing.
2	Put the {dragged_texture} object in {scene} into the {base_texture} object.	Move objects in the {scene} so that the {dragged_texture} item is on one {base_texture} item.
3	Rotate the {dragged_obj} {angle_in_degree} degrees.	Turn the {dragged_obj} precisely {angle_in_degree} degrees.
4	Rearrange to this {scene}.	Rearrange things into this setup {scene}.
5	Rearrange objects to this setup {scene} and then restore.	Rearrange objects into this configuration {scene} and put it back.
6	{demo_blicker_obj_1} is kobar than {demo_blicker_obj_2}. {demo_blicker_obj_3} is kobar than {demo_blicker_obj_4}. Put the kobar {dragged_obj} into the {base_obj}.	{object1} {object3} and {object5} are all kobar than objects {object2} {object4} and {object6} respectively. move the kobar {dragged_obj} inside of the {base_obj}.
7	This is a blinket {dragged_obj}. This is a zup {base_obj}. Put a zup into a blinket.	This is a blinket {object2}. this is a zup {object1}. drop the zup inside of the blinket.
11	Stack objects in this order: {frame1} {frame2} {frame3}.	Move objects like this: {frame1} {frame2} {frame3}.
16	First put {object1} into {object2} then put the object that was previously at its {direction} into the same {object2}.	Set {object1} in {object2} then place the item that was at its {direction} before you placed it into the same place.
17	Put {object1} into {object2}. Finally restore it into its original container.	Set {object1} within {object2} then restore it to its original place.

Table E.2: Examples of how each original instruction was converted into an alternative paraphrase using the meta-templates.

F Further Experimental Results

F.1 Per-Task Results

We report the per-task results for each table reported in the main paper. Table F.1 contains a mapping from each table in the paper to the one with the per-task results. Some tasks only exist for certain generalisation levels and therefore are left blank for other levels.

Per-Level	Per-Task
Table 1	Table F.2 and Table F.3
Table 2	Table F.4
Table 3	Table F.5
Table 4	Table F.6
Table 5	Table F.7
Table 6	Table F.8
Table 7	Table F.9 and Table F.10
Table 8	Table F.11, Table F.12, Table F.13, and Table F.14

Table F.1: Mapping of per-task results for each table listed in the main paper.

F.2 Exploring Task Success at Higher Difficulty Levels and Masked Instructions

Table F.10 shows that model performance drops to 0 for most tasks without instructions, as expected. However, T1 (pick-and-place), T2 (pick-and-place from a frame), and particularly T12 (object sweeping) can still be performed. T12 shows the best performance, followed by T1 and T2, with T12's performance remaining significantly higher than T1 at increased difficulty levels for all models except *Cross-Attn + Obj-Centric*.

T12 is unique in VIMA-BENCH as the only training task requiring sweeping objects into some boundary. Without instructions, the model has a 50/50 chance of choosing the correct object type to sweep. Therefore, the model has likely overfit to perform a sweeping action when using a spatula, as it's the only task with this specific end-effector. This explains T12's higher performance across difficulty levels and reinforces the claim that without instructions, models rely on spurious correlations learned during training, such as associating the spatula with sweeping, rather than true task understanding.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Trained and Evaluated on Original Instructions																		
Cross-Attn + Obj-Centric																		
L1	100.0	99.5	99.5	97.0	8.5	100.0	100.0	—	19.0	—	91.5	96.0	—	—	96.5	49.5	73.5	79.3
L2	99.5	99.5	100.0	98.0	9.5	99.0	99.5	—	18.0	—	95.0	96.5	—	—	92.0	47.5	71.0	78.8
L3	100.0	99.0	100.0	99.0	10.0	98.5	99.5	—	14.0	—	90.5	—	—	—	93.0	43.0	21.5	72.3
L4	—	—	—	—	—	—	—	96.5	—	0.5	—	—	0.0	97.5	—	—	—	48.6
Cross-Attn + Patches																		
L1	91.5	75.0	97.5	12.0	1.0	76.5	95.0	—	9.0	—	90.5	93.0	—	—	79.5	95.5	2.5	63.0
L2	94.0	73.5	96.5	9.5	2.5	78.5	92.0	—	14.0	—	87.5	91.5	—	—	71.5	94.5	0.0	62.0
L3	57.0	70.0	68.0	9.0	0.5	72.5	57.5	—	11.5	—	85.0	—	—	—	62.0	44.0	2.0	44.9
L4	—	—	—	—	—	—	—	25.5	—	1.0	—	—	0.0	29.0	—	—	—	13.9
Concatenate + Obj-Centric																		
L1	100.0	100.0	99.5	97.0	19.0	100.0	100.0	—	13.5	—	88.5	95.0	—	—	96.0	45.5	75.5	79.2
L2	99.5	100.0	99.5	99.0	19.0	100.0	100.0	—	16.0	—	91.0	95.5	—	—	92.0	39.0	74.5	78.8
L3	98.0	97.0	100.0	99.0	21.0	92.0	96.5	—	18.5	—	95.0	—	—	—	96.5	43.0	68.5	77.1
L4	—	—	—	—	—	—	—	97.0	—	2.5	—	—	0.0	97.5	—	—	—	49.2
Concatenate + Patches																		
L1	96.0	84.5	97.5	13.0	2.5	87.0	95.0	—	42.5	—	96.0	96.5	—	—	75.0	94.0	4.0	68.0
L2	92.5	73.5	97.5	17.0	3.5	93.5	91.0	—	31.0	—	95.5	88.0	—	—	75.0	96.5	7.0	66.3
L3	71.5	66.5	91.0	12.5	3.5	93.0	58.0	—	30.5	—	87.0	—	—	—	58.5	61.0	2.0	52.9
L4	—	—	—	—	—	—	—	44.0	—	11.0	—	—	0.0	38.5	—	—	—	23.4
Trained on Original Instructions; Evaluated on Paraphrases																		
Cross-Attn + Obj-Centric																		
L1	99.5	100.0	99.0	86.5	57.0	100.0	100.0	—	15.0	—	60.5	94.0	—	—	99.5	47.0	64.0	78.6
L2	97.5	100.0	99.5	85.5	52.5	100.0	100.0	—	14.5	—	59.0	96.5	—	—	97.5	49.5	57.0	77.6
L3	92.0	96.5	99.5	87.0	58.5	99.0	99.0	—	12.0	—	51.0	—	—	—	97.0	39.5	6.5	69.8
L4	—	—	—	—	—	—	—	90.5	—	0.0	—	—	0.0	98.0	—	—	—	47.1
Cross-Attn + Patches																		
L1	88.5	72.5	96.0	11.5	0.5	66.5	95.0	—	13.5	—	91.5	92.5	—	—	76.5	86.5	3.0	61.1
L2	81.5	52.0	93.5	8.0	2.0	66.0	93.5	—	12.5	—	94.5	89.5	—	—	68.5	93.5	6.0	58.5
L3	57.0	64.0	81.0	8.5	2.0	64.0	65.0	—	14.5	—	90.0	—	—	—	51.0	45.5	1.0	45.3
L4	—	—	—	—	—	—	—	27.5	—	1.5	—	—	0.0	38.0	—	—	—	16.8
Concatenate + Obj-Centric																		
L1	100.0	100.0	96.5	73.5	4.0	100.0	100.0	—	16.5	—	80.0	89.5	—	—	88.5	42.0	38.5	71.5
L2	99.5	99.0	96.5	79.0	10.0	99.5	100.0	—	18.0	—	79.0	94.5	—	—	83.5	46.5	33.0	72.2
L3	87.0	81.5	97.0	76.0	6.0	85.0	94.5	—	17.5	—	71.0	—	—	—	79.0	47.5	10.0	62.7
L4	—	—	—	—	—	—	—	90.5	—	0.5	—	—	0.5	80.5	—	—	—	43.0
Concatenate + Patches																		
L1	95.0	80.5	47.5	11.5	3.0	76.0	93.5	—	34.0	—	91.5	94.0	—	—	78.5	85.5	7.0	61.3
L2	90.0	66.5	42.5	12.5	4.0	74.5	89.5	—	28.5	—	86.5	89.0	—	—	77.0	77.5	3.0	57.0
L3	70.5	66.5	45.0	13.0	1.0	69.0	59.0	—	33.0	—	89.0	—	—	—	47.0	56.0	3.0	46.0
L4	—	—	—	—	—	—	—	29.0	—	17.0	—	—	0.0	36.0	—	—	—	20.5

Table F.2: Per-task average success rate when evaluating performance on either **original instructions or paraphrases** during inference, corresponding to Table 1a and Table 1b respectively. All models are trained **on original instructions.**

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Trained on Paraphrases; Evaluated on the Original Instructions																	
Cross-Attn + Obj-Centric																	
L1	99.5	100.0	99.5	98.5	62.5	100.0	100.0	—	11.5	—	92.0	97.5	—	—	99.0	43.5	72.0	82.7
L2	99.0	100.0	99.5	98.0	55.5	100.0	100.0	—	13.5	—	91.5	92.5	—	—	97.0	48.0	69.0	81.8
L3	99.0	99.0	99.5	99.0	68.5	99.0	99.0	—	15.5	—	93.0	—	—	—	99.0	48.5	10.0	77.4
L4	—	—	—	—	—	—	—	93.0	—	0.5	—	—	0.0	98.5	—	—	—	48.0
Cross-Attn + Patches																	
L1	92.0	77.0	96.0	12.5	0.5	83.0	97.0	—	16.5	—	93.0	93.0	—	—	74.5	92.0	3.5	63.9
L2	90.0	66.5	97.0	9.5	1.0	93.5	94.5	—	13.0	—	93.0	87.5	—	—	79.0	91.0	3.5	63.0
L3	66.5	65.0	78.0	12.5	0.5	88.0	58.5	—	10.5	—	89.5	—	—	—	60.5	61.5	2.5	49.5
L4	—	—	—	—	—	—	—	45.5	—	0.5	—	—	0.0	35.5	—	—	—	20.4
Concatenate + Obj-Centric																	
L1	100.0	100.0	99.0	99.0	18.0	100.0	100.0	—	13.0	—	93.0	98.0	—	—	96.5	51.0	77.5	80.4
L2	100.0	100.0	100.0	98.0	8.5	100.0	100.0	—	13.5	—	92.0	92.5	—	—	92.0	46.5	73.5	78.2
L3	98.0	94.0	100.0	99.5	14.0	94.5	93.0	—	12.5	—	96.5	—	—	—	98.0	42.0	56.0	74.8
L4	—	—	—	—	—	—	—	96.5	—	2.5	—	—	0.0	97.0	—	—	—	49.0
Concatenate + Patches																	
L1	97.0	81.5	98.5	13.0	1.5	94.5	96.0	—	33.0	—	89.0	92.5	—	—	73.5	98.0	4.0	67.1
L2	89.5	69.5	96.0	11.5	2.0	93.5	87.5	—	23.0	—	91.5	92.5	—	—	67.0	92.5	1.0	62.8
L3	65.0	74.5	87.5	14.0	3.0	88.5	60.5	—	29.0	—	85.5	—	—	—	50.5	65.0	1.5	52.0
L4	—	—	—	—	—	—	—	38.5	—	10.0	—	—	0.0	30.5	—	—	—	19.8
Trained and Evaluated on Paraphrases																	
Cross-Attn + Obj-Centric																	
L1	98.5	100.0	97.5	85.5	56.0	99.5	100.0	—	12.0	—	55.5	96.0	—	—	99.0	44.0	62.5	77.4
L2	98.0	99.5	99.5	90.5	63.5	99.0	100.0	—	10.5	—	55.0	92.0	—	—	98.0	45.0	57.5	77.5
L3	91.5	98.0	98.5	88.0	64.0	97.5	98.5	—	14.0	—	48.0	—	—	—	95.5	53.0	2.5	70.8
L4	—	—	—	—	—	—	—	94.0	—	2.5	—	—	0.0	98.0	—	—	—	48.6
Cross-Attn + Patches																	
L1	92.0	66.5	97.5	8.0	0.5	73.0	95.5	—	15.0	—	91.5	92.0	—	—	82.0	92.0	3.5	62.2
L2	92.5	53.5	96.0	13.5	0.5	72.5	94.0	—	17.5	—	93.0	90.0	—	—	75.0	93.5	1.5	61.0
L3	63.5	50.0	80.0	1.5	0.5	69.0	62.0	—	16.0	—	87.5	—	—	—	64.5	50.5	3.0	45.7
L4	—	—	—	—	—	—	—	24.5	—	3.5	—	—	0.0	36.5	—	—	—	16.1
Concatenate + Obj-Centric																	
L1	100.0	99.5	99.5	56.5	2.5	100.0	100.0	—	15.5	—	69.0	95.5	—	—	94.0	45.5	17.5	68.8
L2	100.0	97.5	99.5	54.5	7.0	100.0	99.5	—	15.5	—	60.0	94.0	—	—	89.5	43.5	13.0	67.2
L3	88.5	84.0	100.0	52.5	5.0	94.5	94.0	—	14.5	—	54.0	—	—	—	86.0	39.0	3.5	59.6
L4	—	—	—	—	—	—	—	97.5	—	2.0	—	—	0.0	84.5	—	—	—	46.0
Concatenate + Patches																	
L1	100.0	99.0	99.5	53.5	5.5	99.5	100.0	—	11.0	—	61.0	94.5	—	—	92.5	42.5	15.5	67.2
L2	99.5	99.5	100.0	62.0	6.5	100.0	100.0	—	14.0	—	60.5	95.0	—	—	90.0	43.0	12.0	67.8
L3	93.0	79.5	99.5	59.5	4.0	95.5	94.5	—	13.0	—	57.5	—	—	—	88.5	38.0	4.0	60.5
L4	—	—	—	—	—	—	—	97.0	—	4.0	—	—	0.0	86.5	—	—	—	46.9

Table F.3: Per-task average success rate when evaluating performance on either **original instructions or paraphrases** during inference, corresponding to Table 1c and Table 1d respectively. All models are trained **on paraphrased instructions.**

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
With Visual Referents*																		
Cross-Attn + Obj-Centric																		
L1	99.5	—	99.5	—	—	—	100.0	—	—	—	—	97.5	—	—	99.0	43.5	72.0	87.3
L2	99.0	—	99.5	—	—	—	100.0	—	—	—	—	92.5	—	—	97.0	48.0	69.0	86.4
L3	99.0	—	99.5	—	—	—	99.0	—	—	—	—	—	—	—	99.0	48.5	10.0	75.8
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	98.5	—	—	—	49.2
Cross-attention + Patches																		
L1	92.0	—	96.0	—	—	—	97.0	—	—	—	—	93.0	—	—	74.5	92.0	3.5	78.3
L2	90.0	—	97.0	—	—	—	94.5	—	—	—	—	87.5	—	—	79.0	91.0	3.5	77.5
L3	66.5	—	78.0	—	—	—	58.5	—	—	—	—	—	—	—	60.5	61.5	2.5	54.6
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	35.5	—	—	—	17.8
Concatenate + Obj-Centric																		
L1	100.0	—	99.0	—	—	—	100.0	—	—	—	—	98.0	—	—	96.5	51.0	77.5	88.9
L2	100.0	—	100.0	—	—	—	100.0	—	—	—	—	92.5	—	—	92.0	46.5	73.5	86.4
L3	98.0	—	100.0	—	—	—	93.0	—	—	—	—	—	—	—	98.0	42.0	56.0	81.2
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	97.0	—	—	—	48.5
Concatenate + Patches																		
L1	97.0	—	98.5	—	—	—	96.0	—	—	—	—	92.5	—	—	73.5	98.0	4.0	79.9
L2	89.5	—	96.0	—	—	—	87.5	—	—	—	—	92.5	—	—	67.0	92.5	1.0	75.1
L3	65.0	—	87.5	—	—	—	60.5	—	—	—	—	—	—	—	50.5	65.0	1.5	55.0
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	30.5	—	—	—	15.2
Replace Visual Referents with Descriptors*																		
Cross-Attn + Obj-Centric																		
L1	100.0	—	100.0	—	—	—	100.0	—	—	—	—	97.5	—	—	97.5	47.5	72.5	87.9
L2	100.0	—	99.0	—	—	—	99.5	—	—	—	—	94.5	—	—	98.5	47.0	72.0	87.2
L3	99.0	—	99.5	—	—	—	96.5	—	—	—	—	—	—	—	96.5	48.5	0.0	73.3
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	98.0	—	—	—	49.0
Cross-attention + Patches																		
L1	69.5	—	41.5	—	—	—	64.0	—	—	—	—	72.5	—	—	44.0	33.0	3.0	46.8
L2	61.5	—	31.0	—	—	—	64.5	—	—	—	—	81.0	—	—	38.5	35.5	1.0	44.7
L3	56.0	—	42.5	—	—	—	50.0	—	—	—	—	—	—	—	42.0	37.0	1.5	38.2
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	51.0	—	—	—	25.5
Concatenate + Obj-Centric																		
L1	100.0	—	99.5	—	—	—	100.0	—	—	—	—	97.0	—	—	98.5	44.0	17.0	79.4
L2	99.0	—	100.0	—	—	—	99.5	—	—	—	—	93.0	—	—	92.5	47.0	16.0	78.1
L3	94.5	—	99.5	—	—	—	94.5	—	—	—	—	—	—	—	89.0	41.0	1.5	70.0
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	77.0	—	—	—	38.5
Concatenate + Patches																		
L1	82.5	—	92.5	—	—	—	61.5	—	—	—	—	85.5	—	—	25.5	45.0	2.5	56.4
L2	65.5	—	80.5	—	—	—	58.5	—	—	—	—	81.0	—	—	29.0	39.0	0.5	50.6
L3	72.0	—	88.0	—	—	—	55.5	—	—	—	—	—	—	—	55.5	41.0	0.0	52.0
L4	—	—	—	—	—	—	—	—	—	—	—	—	0.0	51.5	—	—	—	25.8

Table F.4: Per-task average success rate when evaluating performance either with **visual referents (top) and when visual referents are replaced with descriptors (bottom),** corresponding to Table 2. All **models are trained on paraphrases**. As mentioned in Appendix D.5, not all instructions contain visual referents that can be directly substituted for language. For ease of comparison, only tasks with instructions that support substitutions are included in the top section, with the average for the level calculated with only these tasks.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Gobbledygook Tokens																		
Cross-Attn + Obj-Centric																		
L1	89.5	74.5	5.5	73.5	8.5	92.5	90.0	—	1.0	—	92.5	94.0	—	—	29.0	19.5	67.5	56.7
L2	89.5	67.5	5.0	71.5	12.0	86.5	89.5	—	0.5	—	92.0	93.5	—	—	25.0	19.0	56.5	54.5
L3	63.0	48.0	7.5	75.0	11.5	59.5	59.5	—	0.0	—	93.5	—	—	—	10.5	11.0	0.5	36.6
L4	—	—	—	—	—	—	—	72.5	—	0.0	—	—	0.0	19.0	—	—	—	22.9
Cross-Attn + Patches																		
L1	88.5	32.5	23.0	7.0	0.5	53.5	90.0	—	13.0	—	92.0	88.5	—	—	66.5	28.0	4.5	45.2
L2	87.5	33.0	24.5	8.5	0.5	53.5	92.5	—	12.0	—	94.0	91.5	—	—	67.0	30.0	2.0	45.9
L3	63.0	43.5	17.5	7.0	0.5	48.5	61.5	—	13.5	—	88.0	—	—	—	49.5	14.5	1.5	34.0
L4	—	—	—	—	—	—	—	22.5	—	1.5	—	—	0.0	36.5	—	—	—	15.1
Concatenate + Obj-Centric																		
L1	100.0	97.0	15.0	81.5	9.5	100.0	99.5	—	0.5	—	93.0	95.0	—	—	17.5	21.5	7.5	56.7
L2	99.0	95.0	9.5	86.0	5.5	99.0	99.5	—	1.0	—	94.0	96.5	—	—	12.5	16.0	5.5	55.3
L3	87.0	68.0	10.0	93.5	7.0	86.0	78.0	—	0.5	—	90.5	—	—	—	15.5	11.5	2.0	45.8
L4	—	—	—	—	—	—	—	89.5	—	4.5	—	—	0.0	11.5	—	—	—	26.4
Concatenate + Patches																		
L1	89.5	56.5	16.0	11.5	0.5	52.0	85.0	—	11.5	—	76.0	92.0	—	—	66.5	35.5	4.0	45.9
L2	89.5	46.5	15.0	7.5	2.0	53.0	85.0	—	12.0	—	78.5	87.0	—	—	64.0	33.0	3.0	44.3
L3	66.5	53.0	21.0	8.5	1.5	47.5	45.5	—	11.0	—	80.5	—	—	—	37.5	21.0	1.0	32.9
L4	—	—	—	—	—	—	—	25.5	—	8.0	—	—	0.0	46.5	—	—	—	20.0
Gobbledygook Words																		
Cross-Attn + Obj-Centric																		
L1	95.0	98.5	9.5	17.0	4.5	98.0	99.5	—	0.5	—	56.0	90.5	—	—	58.5	17.0	16.5	50.8
L2	95.5	97.0	7.0	23.0	2.5	99.0	100.0	—	1.5	—	52.5	93.0	—	—	64.5	25.0	13.5	51.8
L3	74.5	86.5	6.0	23.5	3.0	93.5	87.5	—	0.0	—	56.5	—	—	—	37.0	11.0	0.0	39.9
L4	—	—	—	—	—	—	—	85.5	—	0.0	—	—	0.0	49.5	—	—	—	33.8
Cross-Attn + Patches																		
L1	95.5	47.5	21.5	8.0	0.5	50.5	92.5	—	11.0	—	89.0	87.5	—	—	72.5	28.0	3.5	46.7
L2	91.5	55.5	21.5	6.5	1.0	63.0	93.0	—	12.5	—	91.5	91.5	—	—	75.0	24.5	2.5	48.4
L3	61.0	52.0	13.5	7.0	0.5	52.5	57.0	—	12.5	—	81.5	—	—	—	47.5	22.0	0.0	33.9
L4	—	—	—	—	—	—	—	26.5	—	3.5	—	—	0.0	44.5	—	—	—	18.6
Concatenate + Obj-Centric																		
L1	99.5	99.0	15.5	10.0	0.0	100.0	100.0	—	0.5	—	46.0	90.5	—	—	4.0	13.0	5.0	44.8
L2	99.5	94.5	17.5	11.0	1.0	98.5	98.5	—	1.5	—	47.0	87.5	—	—	5.0	12.0	4.5	44.5
L3	84.0	74.5	19.5	10.0	0.0	88.5	84.0	—	2.5	—	51.5	—	—	—	7.0	3.0	0.0	35.4
L4	—	—	—	—	—	—	—	87.0	—	3.0	—	—	0.0	5.5	—	—	—	23.9
Concatenate + Patches																		
L1	91.0	60.5	15.0	7.0	1.0	48.5	86.0	—	10.0	—	70.5	91.0	—	—	69.0	23.5	2.5	44.3
L2	87.0	50.5	22.0	9.5	0.5	51.5	82.0	—	5.5	—	69.5	90.5	—	—	63.0	22.0	1.5	42.7
L3	60.5	51.0	17.0	6.0	1.5	46.0	56.5	—	6.0	—	70.0	—	—	—	40.5	16.0	0.5	31.0
L4	—	—	—	—	—	—	—	21.0	—	9.0	—	—	0.0	46.0	—	—	—	19.0

Table F.5: Per-task average success rate when evaluating performance with either *Gobbledygook Tokens* **(top) and** *Gobbledygook Words* **(bottom).** All models are trained **on paraphrased instructions**. This table corresponds to Table 3.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Mask Language Tokens																		
Cross-Attn + Obj-Centric																		
L1	74.5	49.0	0.5	2.0	0.0	76.0	71.0	—	0.0	—	23.5	96.0	—	—	20.5	15.5	44.0	36.3
L2	72.5	53.0	0.0	4.0	0.0	66.0	68.0	—	0.0	—	25.5	94.0	—	—	20.0	17.5	35.5	35.1
L3	45.0	39.5	0.5	3.5	0.0	51.0	48.0	—	0.0	—	30.0	—	—	—	6.0	5.5	0.0	19.1
L4	—	—	—	—	—	—	—	43.0	—	0.5	—	—	0.0	15.5	—	—	—	14.8
Cross-Attn + Patches																		
L1	58.5	35.5	8.0	0.0	0.0	46.5	56.0	—	6.0	—	18.0	87.0	—	—	6.5	17.0	3.5	26.3
L2	58.0	24.0	10.0	0.0	0.0	54.0	60.0	—	6.0	—	17.0	83.5	—	—	6.0	23.5	2.5	26.5
L3	50.5	23.5	14.0	0.0	0.0	43.5	50.0	—	11.5	—	21.5	—	—	—	21.0	11.0	1.5	20.7
L4	—	—	—	—	—	—	—	19.5	—	2.5	—	—	0.0	23.0	—	—	—	11.2
Concatenate + Obj-Centric																		
L1	97.0	95.5	8.0	0.0	0.0	99.5	99.0	—	0.0	—	5.0	92.5	—	—	2.5	7.0	0.5	39.0
L2	96.0	96.0	6.0	0.0	0.0	99.0	98.5	—	1.5	—	5.0	91.5	—	—	1.5	11.0	1.5	39.0
L3	80.0	79.0	5.0	0.0	0.0	87.0	79.5	—	1.5	—	3.0	—	—	—	1.0	9.0	0.0	28.8
L4	—	—	—	—	—	—	—	94.5	—	4.0	—	—	0.0	5.0	—	—	—	25.9
Concatenate + Patches																		
L1	71.5	58.5	7.0	0.0	0.0	56.0	60.5	—	1.5	—	9.0	78.5	—	—	33.0	17.5	0.0	30.2
L2	66.5	60.0	5.0	0.0	0.0	49.0	61.5	—	1.0	—	11.0	77.5	—	—	28.5	16.0	1.0	29.0
L3	60.0	55.5	8.5	0.0	0.0	57.0	57.0	—	1.0	—	9.0	—	—	—	37.0	9.5	0.0	24.5
L4	—	—	—	—	—	—	—	23.0	—	3.0	—	—	0.0	39.0	—	—	—	16.2
Mask Visual Referents																		
Cross-Attn + Obj-Centric																		
L1	100.0	100.0	100.0	0.0	0.0	100.0	100.0	—	13.5	—	6.5	95.0	—	—	94.5	48.5	69.0	63.6
L2	100.0	99.5	100.0	0.0	0.0	98.0	99.5	—	12.0	—	6.0	94.0	—	—	88.0	49.5	67.0	62.6
L3	99.0	100.0	100.0	0.0	0.0	98.5	99.0	—	9.0	—	4.0	—	—	—	98.0	50.0	19.5	56.4
L4	—	—	—	—	—	—	—	92.0	—	0.5	—	—	0.0	99.0	—	—	—	47.9
Cross-Attn + Patches																		
L1	96.0	82.0	98.5	10.0	0.5	89.0	95.5	—	16.0	—	91.5	93.5	—	—	72.5	94.5	2.5	64.8
L2	92.0	64.5	97.5	12.5	1.5	92.5	95.0	—	11.5	—	89.5	93.5	—	—	70.0	94.0	5.0	63.0
L3	64.0	70.0	82.0	9.5	1.5	83.5	58.5	—	14.5	—	90.0	—	—	—	63.0	57.0	2.0	49.6
L4	—	—	—	—	—	—	—	41.5	—	5.5	—	—	0.0	35.5	—	—	—	20.6
Concatenate + Obj-Centric																		
L1	100.0	100.0	99.5	0.0	0.0	100.0	100.0	—	15.5	—	1.0	98.5	—	—	96.5	47.0	19.0	59.8
L2	100.0	100.0	99.5	0.0	0.0	100.0	100.0	—	15.5	—	0.5	97.0	—	—	91.0	43.5	19.0	58.9
L3	94.5	92.5	100.0	0.0	0.0	95.5	94.5	—	14.5	—	3.0	—	—	—	94.0	47.0	2.5	53.2
L4	—	—	—	—	—	—	—	95.5	—	2.5	—	—	0.0	93.0	—	—	—	47.8
Concatenate + Patches																		
L1	97.5	86.0	98.5	13.5	5.5	88.5	94.0	—	35.0	—	90.0	92.5	—	—	71.5	97.0	3.0	67.1
L2	89.5	67.0	95.5	15.5	2.0	93.5	90.0	—	29.0	—	89.0	90.5	—	—	71.5	92.5	2.0	63.7
L3	69.0	73.5	89.0	15.5	2.0	85.5	63.5	—	30.5	—	86.5	—	—	—	52.5	65.5	0.5	52.8
L4	—	—	—	—	—	—	—	44.0	—	10.0	—	—	0.0	38.0	—	—	—	23.0

Table F.6: Per-task average success rate when **evaluating performance after masking language tokens (top) or visual referents (bottom)** within each multimodal instruction. All models are trained **on paraphrased instructions**. This table corresponds to Table 4.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Cross-Attn + Obj-Centric																		
L1	99.0	99.0	99.0	83.5	0.0	97.5	98.0	—	11.5	—	92.0	97.5	—	—	96.0	41.5	0.0	70.3
L2	97.5	98.0	99.5	78.0	0.0	98.0	99.0	—	13.5	—	91.0	91.5	—	—	94.5	46.0	0.0	69.7
L3	98.0	97.0	99.5	77.5	0.0	97.5	95.5	—	15.5	—	92.5	—	—	—	94.5	47.5	0.0	67.9
L4	—	—	—	—	—	—	—	92.0	—	0.0	—	—	0.0	95.0	—	—	—	46.8
Cross-Attn + Patches																		
L1	88.5	66.0	92.0	8.0	0.0	79.0	96.5	—	10.5	—	93.0	75.5	—	—	58.5	89.5	0.0	58.2
L2	87.5	56.0	92.0	6.5	0.5	87.0	91.5	—	4.5	—	92.0	82.0	—	—	58.5	87.0	0.0	57.3
L3	61.0	59.5	69.0	7.0	0.0	84.5	50.5	—	5.0	—	87.5	—	—	—	46.5	60.5	0.0	44.2
L4	—	—	—	—	—	—	—	38.0	—	0.0	—	—	0.0	25.5	—	—	—	15.9
Concatenate + Obj-Centric																		
L1	99.5	99.5	98.5	98.0	6.5	99.5	99.0	—	13.0	—	90.0	92.0	—	—	95.5	47.5	0.0	72.2
L2	99.0	100.0	99.5	97.5	4.5	98.5	99.5	—	13.5	—	90.5	91.0	—	—	89.5	45.0	0.0	71.4
L3	91.5	90.0	100.0	99.5	9.5	72.0	84.0	—	12.5	—	93.0	—	—	—	94.0	42.0	0.0	65.7
L4	—	—	—	—	—	—	—	90.5	—	0.0	—	—	0.0	91.5	—	—	—	45.5
Concatenate + Patches																		
L1	94.0	72.5	97.5	6.5	1.5	92.5	90.5	—	26.0	—	87.0	74.0	—	—	59.0	95.0	0.0	61.2
L2	83.5	62.5	90.5	9.5	2.0	91.5	82.5	—	14.0	—	90.0	79.0	—	—	52.0	91.5	0.0	57.6
L3	57.5	68.5	81.5	11.0	1.5	86.5	49.5	—	18.0	—	81.5	—	—	—	32.0	64.0	0.0	46.0
L4	—	—	—	—	—	—	—	34.0	—	2.5	—	—	0.0	15.0	—	—	—	12.9

Table F.7: Per-task average success rate when evaluating performance **without allowing models to recover from mistakes.** This table corresponds to Table 5. All models are trained **on paraphrased instructions**.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Mask Instructions; Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	99.5	99.0	22.0	0.0	0.0	96.0	100.0	—	2.0	—	30.5	97.5	—	—	73.5	23.0	38.0	52.4
L2	99.0	98.5	15.5	0.0	0.0	96.5	100.0	—	1.0	—	26.0	93.5	—	—	67.0	29.0	30.0	50.5
L3	97.0	92.5	17.5	0.0	0.0	84.0	94.5	—	1.5	—	30.0	—	—	—	48.5	22.0	0.0	40.6
L4	—	—	—	—	—	—	—	84.0	—	0.0	—	—	0.0	48.0	—	—	—	33.0
Cross-Attn + Patches																		
L1	62.0	24.0	11.5	0.0	0.0	46.5	57.0	—	8.0	—	13.5	89.0	—	—	7.5	19.5	5.5	26.5
L2	54.0	26.5	15.0	0.0	0.0	53.0	53.0	—	6.5	—	17.0	86.0	—	—	6.0	16.5	2.5	25.8
L3	46.0	30.0	12.5	0.0	0.0	44.5	50.0	—	5.5	—	18.0	—	—	—	19.0	14.5	2.0	20.2
L4	—	—	—	—	—	—	—	22.5	—	1.0	—	—	0.0	24.0	—	—	—	11.9
Concatenate + Obj-Centric																		
L1	73.0	71.5	8.0	0.0	0.0	71.5	75.0	—	1.0	—	1.5	88.5	—	—	1.0	11.0	0.0	30.9
L2	67.5	68.5	11.0	0.0	0.0	72.0	76.5	—	1.0	—	1.5	85.5	—	—	1.0	11.0	0.5	30.5
L3	58.0	63.5	9.0	0.0	0.0	59.5	60.0	—	1.5	—	3.0	—	—	—	4.0	6.5	0.0	22.1
L4	—	—	—	—	—	—	—	53.0	—	0.0	—	—	0.0	3.5	—	—	—	14.1
Concatenate + Patches																		
L1	70.0	65.0	4.0	0.0	0.0	56.0	62.5	—	1.5	—	13.5	74.5	—	—	28.5	17.5	2.0	30.4
L2	71.5	59.5	8.5	0.0	0.0	51.0	61.0	—	0.5	—	6.5	80.5	—	—	27.5	14.0	1.5	29.4
L3	62.5	53.0	7.0	0.0	0.0	53.0	55.5	—	0.0	—	16.0	—	—	—	38.0	15.0	0.0	25.0
L4	—	—	—	—	—	—	—	25.0	—	4.0	—	—	0.0	33.5	—	—	—	15.6
Mask Instructions; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	99.0	97.0	10.0	0.0	0.0	83.0	99.5	—	1.5	—	22.0	95.0	—	—	61.0	19.5	0.0	45.2
L2	98.0	95.5	8.5	0.0	0.0	86.5	98.0	—	0.5	—	13.0	89.5	—	—	55.5	26.0	0.0	43.9
L3	96.0	78.0	6.0	0.0	0.0	62.0	86.5	—	1.0	—	18.0	—	—	—	32.0	18.0	0.0	33.1
L4	—	—	—	—	—	—	—	76.0	—	0.0	—	—	0.0	33.5	—	—	—	27.4
Cross-Attn + Patches																		
L1	52.0	13.0	2.5	0.0	0.0	37.0	45.0	—	1.5	—	4.5	79.5	—	—	3.0	16.0	0.0	19.5
L2	44.0	15.5	4.5	0.0	0.0	41.0	41.0	—	1.0	—	5.0	75.5	—	—	1.0	14.0	0.0	18.7
L3	37.5	17.5	3.5	0.0	0.0	35.5	44.5	—	1.0	—	5.0	—	—	—	15.0	14.0	0.0	14.5
L4	—	—	—	—	—	—	—	12.5	—	0.0	—	—	0.0	18.0	—	—	—	7.6
Concatenate + Obj-Centric																		
L1	12.5	14.5	0.0	0.0	0.0	12.0	15.0	—	0.0	—	0.0	33.0	—	—	0.0	4.5	0.0	7.0
L2	16.5	15.5	0.0	0.0	0.0	10.0	15.5	—	0.0	—	0.5	32.5	—	—	0.5	2.5	0.0	7.2
L3	10.0	11.0	0.0	0.0	0.0	10.0	10.0	—	0.0	—	1.0	—	—	—	0.0	0.0	0.0	3.5
L4	—	—	—	—	—	—	—	9.0	—	0.0	—	—	0.0	0.0	—	—	—	2.2
Concatenate + Patches																		
L1	56.5	50.0	0.5	0.0	0.0	45.0	46.0	—	0.0	—	6.0	63.0	—	—	11.5	13.5	0.0	22.5
L2	59.5	46.0	2.0	0.0	0.0	36.5	51.5	—	0.0	—	1.5	71.0	—	—	11.0	10.0	0.0	22.2
L3	52.0	44.0	0.5	0.0	0.0	40.5	42.5	—	0.0	—	5.0	—	—	—	24.5	9.5	0.0	18.2
L4	—	—	—	—	—	—	—	19.0	—	0.5	—	—	0.0	15.5	—	—	—	8.8

Table F.8: Per-task average success rate when evaluating performance with **entirely masked instructions**. This compares models' ability to recover from mistakes (top) versus acting without making mistakes (bottom). All models are trained **on paraphrased instructions**. This table corresponds to Table 6.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Distracting																		
Cross-Attn + Obj-Centric																		
L1	65.5	84.5	99.0	0.5	0.0	87.0	73.5	—	4.5	—	88.0	74.5	—	—	78.0	44.5	0.0	53.8
L2	66.5	86.0	97.5	1.0	0.0	84.0	72.5	—	2.5	—	86.5	70.0	—	—	69.0	46.0	0.0	52.4
L3	59.0	70.0	98.0	1.5	0.0	83.0	70.5	—	5.0	—	83.5	—	—	—	44.5	44.5	0.0	46.6
L4	—	—	—	—	—	—	—	79.5	—	0.0	—	—	0.0	59.5	—	—	—	34.8
Cross-Attn + Patches																		
L1	22.5	25.0	19.5	0.0	0.0	16.0	22.5	—	1.0	—	87.5	56.0	—	—	25.0	87.5	0.0	27.9
L2	20.5	15.0	21.0	0.0	0.0	15.0	22.0	—	0.5	—	88.0	57.0	—	—	26.5	91.0	0.0	27.4
L3	11.0	20.0	12.5	0.0	0.0	12.5	9.5	—	0.5	—	84.5	—	—	—	9.0	59.5	0.0	18.2
L4	—	—	—	—	—	—	—	12.0	—	1.0	—	—	0.0	2.0	—	—	—	3.8
Concatenate + Obj-Centric																		
L1	98.5	99.5	99.0	1.5	0.0	96.0	99.0	—	3.5	—	86.0	76.5	—	—	79.0	44.5	0.0	60.2
L2	99.5	98.0	99.5	0.0	0.0	98.0	99.5	—	4.5	—	85.0	73.5	—	—	73.0	47.5	0.0	59.8
L3	95.0	83.0	100.0	0.0	0.0	80.0	86.5	—	4.0	—	84.5	—	—	—	64.0	42.5	0.0	53.3
L4	—	—	—	—	—	—	—	87.0	—	0.0	—	—	0.0	69.0	—	—	—	39.0
Concatenate + Patches																		
L1	35.5	30.0	18.0	0.0	0.0	18.0	21.5	—	7.5	—	84.0	55.0	—	—	12.5	98.0	0.0	29.2
L2	27.0	20.0	21.5	0.0	0.0	15.5	14.5	—	2.5	—	84.5	61.0	—	—	12.5	91.5	0.0	27.0
L3	13.0	20.5	13.5	0.0	0.0	16.0	10.0	—	2.5	—	81.0	—	—	—	2.5	62.0	0.0	18.4
L4	—	—	—	—	—	—	—	14.5	—	3.0	—	—	0.0	3.0	—	—	—	5.1
Extreme																		
Cross-Attn + Obj-Centric																		
L1	97.5	97.5	74.5	96.0	5.5	98.5	97.0	—	12.0	—	2.0	33.0	—	—	77.0	0.0	0.0	53.1
L2	98.0	96.5	80.0	95.0	7.0	93.5	98.0	—	13.0	—	0.5	34.0	—	—	79.5	0.0	0.0	53.5
L3	100.0	99.5	79.5	98.5	5.0	95.0	99.5	—	16.0	—	0.5	—	—	—	72.5	0.0	0.0	55.5
L4	—	—	—	—	—	—	—	70.5	—	0.5	—	—	0.0	75.5	—	—	—	36.6
Cross-Attn + Patches																		
L1	19.0	7.0	52.0	8.5	1.0	13.0	9.0	—	9.0	—	0.5	33.5	—	—	16.5	0.0	0.0	13.0
L2	15.0	7.5	49.0	10.0	0.5	12.0	8.5	—	5.0	—	2.0	33.5	—	—	20.0	0.0	0.0	12.5
L3	11.0	7.0	46.5	6.0	0.5	10.5	8.0	—	8.0	—	2.5	—	—	—	16.5	0.0	0.0	9.7
L4	—	—	—	—	—	—	—	3.0	—	11.0	—	—	0.5	22.5	—	—	—	9.2
Concatenate + Obj-Centric																		
L1	25.0	4.0	77.0	99.0	4.5	0.5	2.5	—	15.5	—	1.5	32.5	—	—	30.5	0.0	0.0	22.5
L2	22.0	1.5	71.5	99.0	7.0	0.5	3.0	—	13.5	—	4.0	36.5	—	—	40.0	0.0	0.0	23.0
L3	30.5	4.0	77.0	100.0	7.0	1.0	2.0	—	15.5	—	3.0	—	—	—	38.0	0.0	0.0	23.2
L4	—	—	—	—	—	—	—	0.5	—	8.5	—	—	1.5	31.5	—	—	—	10.5
Concatenate + Patches																		
L1	12.5	13.5	74.5	7.0	2.5	11.5	15.5	—	22.5	—	3.5	28.5	—	—	19.5	0.0	0.0	16.2
L2	13.5	11.0	66.5	6.5	0.5	19.5	7.5	—	18.0	—	5.0	29.0	—	—	18.5	0.0	0.0	15.0
L3	15.0	13.0	51.0	7.0	1.0	15.0	5.0	—	16.5	—	3.5	—	—	—	18.0	0.0	0.0	12.1
L4	—	—	—	—	—	—	—	4.5	—	18.0	—	—	0.0	26.0	—	—	—	12.1
Extremely Distracting																		
Cross-Attn + Obj-Centric																		
L1	48.0	71.5	78.5	0.0	0.0	58.0	41.0	—	3.0	—	2.0	33.0	—	—	57.0	0.0	0.0	30.2
L2	43.5	68.5	80.5	0.0	0.0	60.0	47.0	—	4.5	—	0.5	36.0	—	—	58.5	0.0	0.0	30.7
L3	49.0	77.5	74.0	0.0	0.0	67.5	50.0	—	4.0	—	0.5	—	—	—	73.0	0.0	0.0	33.0
L4	—	—	—	—	—	—	—	58.0	—	0.0	—	—	0.5	68.5	—	—	—	31.8
Cross-Attn + Patches																		
L1	3.5	1.0	10.0	0.0	0.0	4.0	1.5	—	1.5	—	0.5	32.0	—	—	4.0	0.0	0.0	4.5
L2	1.5	0.0	9.5	0.0	0.0	2.0	1.0	—	0.0	—	1.5	32.5	—	—	2.0	0.0	0.0	3.8
L3	2.0	4.0	8.5	0.0	0.0	5.0	2.0	—	1.0	—	0.0	—	—	—	3.0	0.0	0.0	2.1
L4	—	—	—	—	—	—	—	2.5	—	8.5	—	—	0.0	0.5	—	—	—	2.9
Concatenate + Obj-Centric																		
L1	27.5	1.0	75.0	0.0	0.0	0.5	4.0	—	6.5	—	4.0	36.0	—	—	35.5	0.0	0.0	14.6
L2	26.5	0.5	75.0	0.0	0.0	1.5	3.5	—	6.5	—	0.5	41.0	—	—	31.0	0.0	0.0	14.3
L3	19.5	0.5	68.5	0.0	0.0	2.5	1.5	—	7.5	—	1.5	—	—	—	28.0	0.0	0.0	10.8
L4	—	—	—	—	—	—	—	0.5	—	0.0	—	—	0.0	33.5	—	—	—	8.5
Concatenate + Patches																		
L1	3.5	5.0	11.0	0.0	0.0	4.5	3.0	—	9.5	—	2.0	32.0	—	—	3.0	0.0	0.0	5.7
L2	4.0	4.0	14.0	0.0	0.0	2.5	4.0	—	3.0	—	1.5	31.5	—	—	2.5	0.0	0.0	5.2
L3	3.5	7.0	9.0	0.0	0.0	4.5	1.0	—	5.5	—	3.0	—	—	—	0.0	0.0	0.0	2.8
L4	—	—	—	—	—	—	—	3.5	—	12.5	—	—	0.0	0.5	—	—	—	4.1

Table F.9: Per-task average success rate when evaluating performance **across each difficulty level, without making mistakes.** This corresponds to the top of Table 7. All models were trained **on paraphrased instructions**.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Distracting																		
Cross-Attn + Obj-Centric																		
L1	69.0	63.5	10.5	0.0	0.0	67.5	70.5	—	0.5	—	10.5	69.0	—	—	47.0	21.5	0.0	33.0
L2	67.5	74.0	8.0	0.0	0.0	67.5	71.0	—	0.0	—	10.0	65.5	—	—	38.0	17.0	0.0	32.2
L3	46.0	55.0	5.0	0.0	0.0	50.5	53.0	—	1.0	—	8.5	—	—	—	20.5	13.0	0.0	21.0
L4	—	—	—	—	—	—	—	62.0	—	0.0	—	—	0.0	25.5	—	—	—	21.9
Cross-Attn + Patches																		
L1	3.5	2.0	0.5	0.0	0.0	8.0	2.5	—	0.0	—	2.5	56.0	—	—	0.0	12.5	0.0	6.7
L2	2.5	4.5	0.0	0.0	0.0	4.0	3.5	—	0.0	—	3.0	62.0	—	—	0.0	16.0	0.0	7.3
L3	4.0	4.5	0.0	0.0	0.0	5.5	2.0	—	0.0	—	0.5	—	—	—	0.0	12.5	0.0	2.4
L4	—	—	—	—	—	—	—	7.5	—	3.0	—	—	0.0	0.0	—	—	—	2.6
Concatenate + Obj-Centric																		
L1	3.5	12.0	0.0	0.0	0.0	9.0	1.5	—	0.0	—	1.5	42.5	—	—	0.0	6.5	0.0	5.9
L2	3.5	4.0	0.0	0.0	0.0	4.0	4.5	—	0.0	—	1.0	40.0	—	—	0.0	2.0	0.0	4.5
L3	2.5	6.5	0.0	0.0	0.0	2.5	2.0	—	0.0	—	0.5	—	—	—	0.0	0.0	0.0	1.2
L4	—	—	—	—	—	—	—	2.0	—	0.0	—	—	0.0	0.5	—	—	—	0.6
Concatenate + Patches																		
L1	2.0	13.0	0.0	0.0	0.0	4.5	5.5	—	0.0	—	1.0	49.5	—	—	1.0	8.5	0.0	6.5
L2	3.5	12.5	0.0	0.0	0.0	4.5	3.0	—	0.0	—	3.5	48.0	—	—	0.5	11.5	0.0	6.7
L3	6.5	17.0	0.0	0.0	0.0	6.0	3.5	—	0.0	—	2.0	—	—	—	1.0	5.0	0.0	3.4
L4	—	—	—	—	—	—	—	6.5	—	0.0	—	—	0.0	0.0	—	—	—	1.6
Extreme																		
Cross-Attn + Obj-Centric																		
L1	61.0	14.5	10.5	0.0	0.0	2.5	30.5	—	0.5	—	0.5	39.5	—	—	37.5	0.0	0.0	15.2
L2	69.0	14.5	6.5	0.0	0.0	5.0	26.5	—	1.0	—	0.0	36.0	—	—	43.0	0.0	0.0	15.5
L3	77.0	31.0	6.0	0.0	0.0	9.0	41.0	—	0.5	—	2.0	—	—	—	42.5	0.0	0.0	17.4
L4	—	—	—	—	—	—	—	10.0	—	0.0	—	—	0.0	35.0	—	—	—	11.2
Cross-Attn + Patches																		
L1	4.0	4.0	3.5	0.0	0.0	6.0	6.0	—	1.5	—	1.0	37.5	—	—	10.5	0.0	0.0	5.7
L2	3.5	2.5	3.5	0.0	0.0	5.0	2.5	—	1.5	—	0.5	34.0	—	—	7.0	0.0	0.0	4.6
L3	4.5	1.0	7.5	0.0	0.0	4.5	2.5	—	4.0	—	0.0	—	—	—	8.0	0.0	0.0	2.7
L4	—	—	—	—	—	—	—	2.5	—	8.0	—	—	0.0	8.0	—	—	—	4.6
Concatenate + Obj-Centric																		
L1	8.5	6.0	1.0	0.0	0.0	8.0	5.0	—	0.5	—	0.5	29.0	—	—	0.5	0.0	0.0	4.5
L2	2.5	3.5	0.0	0.0	0.0	7.0	7.5	—	0.0	—	0.5	21.5	—	—	1.0	0.0	0.0	3.3
L3	6.0	4.0	0.0	0.0	0.0	8.5	4.5	—	0.0	—	0.5	—	—	—	0.0	0.0	0.0	2.0
L4	—	—	—	—	—	—	—	3.5	—	5.0	—	—	0.5	0.0	—	—	—	2.2
Concatenate + Patches																		
L1	4.5	3.0	1.5	0.0	0.0	4.0	5.0	—	0.0	—	1.5	20.5	—	—	11.0	0.0	0.0	3.9
L2	7.0	5.5	0.5	0.0	0.0	6.0	6.0	—	0.0	—	1.0	18.5	—	—	9.0	0.0	0.0	4.1
L3	8.0	2.0	0.0	0.0	0.0	5.5	4.5	—	0.0	—	0.0	—	—	—	12.0	0.0	0.0	2.7
L4	—	—	—	—	—	—	—	2.5	—	5.0	—	—	0.0	22.0	—	—	—	7.4
Extremely Distracting																		
Cross-Attn + Obj-Centric																		
L1	22.5	3.0	9.0	0.0	0.0	8.5	17.5	—	1.0	—	2.5	36.5	—	—	31.0	0.0	0.0	10.1
L2	14.5	6.0	7.0	0.0	0.0	6.5	14.0	—	0.5	—	1.0	35.0	—	—	29.0	0.0	0.0	8.7
L3	23.0	10.5	7.5	0.0	0.0	6.0	17.5	—	0.5	—	2.0	—	—	—	37.0	0.0	0.0	8.7
L4	—	—	—	—	—	—	—	9.5	—	0.0	—	—	0.0	34.0	—	—	—	10.9
Cross-Attn + Patches																		
L1	1.0	2.5	0.0	0.0	0.0	1.5	1.0	—	0.0	—	0.0	32.0	—	—	0.0	0.0	0.0	2.9
L2	1.0	1.0	0.0	0.0	0.0	1.0	0.5	—	0.0	—	0.5	32.5	—	—	0.5	0.0	0.0	2.8
L3	1.5	0.5	0.0	0.0	0.0	0.0	2.0	—	0.0	—	0.5	—	—	—	0.0	0.0	0.0	0.4
L4	—	—	—	—	—	—	—	2.0	—	12.0	—	—	0.5	0.0	—	—	—	3.6
Concatenate + Obj-Centric																		
L1	9.5	5.0	0.5	0.0	0.0	6.0	6.5	—	0.5	—	0.0	28.0	—	—	0.0	0.0	0.0	4.3
L2	3.5	5.0	0.0	0.0	0.0	6.5	7.5	—	0.0	—	0.0	32.5	—	—	0.5	0.0	0.0	4.3
L3	3.0	7.5	0.0	0.0	0.0	7.0	5.0	—	0.0	—	1.0	—	—	—	0.5	0.0	0.0	2.0
L4	—	—	—	—	—	—	—	3.0	—	2.5	—	—	0.0	0.0	—	—	—	1.4
Concatenate + Patches																		
L1	1.5	2.0	0.0	0.0	0.0	1.5	0.0	—	0.0	—	0.0	16.5	—	—	0.5	0.0	0.0	1.7
L2	1.0	1.0	0.0	0.0	0.0	4.0	1.0	—	0.0	—	0.0	13.5	—	—	0.0	0.0	0.0	1.6
L3	0.5	0.5	0.0	0.0	0.0	1.0	1.0	—	0.0	—	0.5	—	—	—	1.0	0.0	0.0	0.4
L4	—	—	—	—	—	—	—	2.5	—	2.0	—	—	0.5	0.0	—	—	—	1.2

Table F.10: Per-task average success rate when evaluating performance **across each difficulty level** when the **instruction is entirely masked**. The model must perform **without making mistakes**. This corresponds to the bottom of Table 7. All models were trained **on paraphrased instructions**.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Permute Object Order; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	60.5	58.5	56.0	9.5	0.5	56.0	63.0	—	14.5	—	92.0	55.0	—	—	33.5	28.5	4.5	40.9
L2	60.0	51.5	61.5	12.0	0.0	53.5	54.0	—	8.5	—	91.0	49.5	—	—	34.5	28.5	3.5	39.1
L3	42.5	38.5	58.5	11.0	1.0	47.5	44.5	—	12.0	—	91.0	—	—	—	27.5	25.5	0.0	33.3
L4	—	—	—	—	—	—	—	20.5	—	0.0	—	—	0.0	26.5	—	—	—	11.8
Concatenate + Obj-Centric																		
L1	59.5	61.0	50.5	19.5	3.5	63.0	60.0	—	8.5	—	91.5	42.0	—	—	30.5	35.0	3.0	40.6
L2	65.5	56.5	56.5	13.0	1.0	56.0	58.5	—	10.5	—	96.5	42.5	—	—	32.0	35.0	4.5	40.6
L3	53.5	49.0	59.0	17.0	3.0	42.5	44.0	—	8.5	—	93.5	—	—	—	29.5	35.5	0.5	36.3
L4	—	—	—	—	—	—	—	21.5	—	1.0	—	—	0.0	35.5	—	—	—	14.5
Permute Object Order; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	22.5	19.0	44.5	6.5	0.0	23.5	20.5	—	5.0	—	91.0	41.5	—	—	24.5	25.5	0.0	24.9
L2	19.5	19.5	49.0	8.5	0.0	23.5	20.5	—	1.5	—	91.0	40.5	—	—	20.0	26.0	0.0	24.6
L3	14.5	13.0	47.5	6.0	0.0	17.5	17.0	—	5.0	—	90.5	—	—	—	13.0	24.5	0.0	20.7
L4	—	—	—	—	—	—	—	7.5	—	0.0	—	—	0.0	16.0	—	—	—	5.9
Concatenate + Obj-Centric																		
L1	21.0	27.0	40.5	19.0	2.0	41.0	30.0	—	2.0	—	89.5	39.0	—	—	18.0	29.5	0.0	27.6
L2	20.5	28.0	49.5	12.0	0.5	34.0	27.5	—	6.5	—	94.0	39.5	—	—	17.0	32.0	0.0	27.8
L3	17.0	20.5	52.5	16.5	2.0	24.0	16.5	—	6.0	—	90.5	—	—	—	17.0	35.5	0.0	24.8
L4	—	—	—	—	—	—	—	13.0	—	0.0	—	—	0.0	20.0	—	—	—	8.2

Table F.11: Per-task average success rate when evaluating performance for **object-centric models with a permuted object order** per observation during inference. This table compares a models' ability to recover from mistakes (top) versus acting without making mistakes (bottom). All models are trained **on paraphrased instructions**.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Permute Object Order; Distracting; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	12.5	26.0	17.0	0.0	0.0	13.0	13.0	—	0.5	—	89.5	46.5	—	—	5.0	29.0	0.0	19.4
L2	11.0	23.5	19.5	0.5	0.0	10.5	9.5	—	1.5	—	88.5	46.0	—	—	4.5	25.5	0.0	18.5
L3	7.5	10.5	19.0	0.5	0.0	10.0	9.0	—	1.0	—	87.0	—	—	—	6.5	33.0	0.0	15.3
L4	—	—	—	—	—	—	—	8.0	—	0.0	—	—	0.0	2.5	—	—	—	2.6
Concatenate + Obj-Centric																		
L1	6.0	24.5	11.5	0.0	0.0	21.0	6.5	—	1.0	—	91.0	20.5	—	—	1.5	35.0	0.0	16.8
L2	5.5	25.5	12.5	0.0	0.0	12.0	9.0	—	1.5	—	86.0	21.5	—	—	2.0	31.5	0.0	15.9
L3	6.5	18.0	12.0	0.0	0.0	10.5	8.0	—	4.5	—	87.5	—	—	—	1.5	32.0	0.0	15.0
L4	—	—	—	—	—	—	—	10.0	—	0.0	—	—	0.0	2.5	—	—	—	3.1
Permute Object Order; Extreme; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	37.0	33.5	48.0	12.0	1.5	24.0	33.5	—	15.0	—	1.0	31.0	—	—	34.0	0.0	0.5	20.8
L2	37.0	37.0	49.0	12.5	1.0	23.0	33.0	—	8.5	—	1.5	29.0	—	—	31.0	0.0	1.5	20.3
L3	38.0	34.0	49.5	15.0	1.5	25.5	34.0	—	11.0	—	0.5	—	—	—	25.0	0.0	0.0	19.5
L4	—	—	—	—	—	—	—	10.5	—	5.5	—	—	0.5	31.5	—	—	—	12.0
Concatenate + Obj-Centric																		
L1	37.0	23.0	40.0	14.0	2.5	18.5	25.0	—	10.5	—	5.0	17.0	—	—	38.5	0.0	1.5	17.9
L2	30.5	22.5	41.0	14.0	2.0	18.5	30.0	—	10.0	—	3.5	15.5	—	—	27.0	0.0	0.0	16.5
L3	36.0	22.0	40.5	12.0	2.0	21.5	22.0	—	9.0	—	6.0	—	—	—	29.0	0.5	0.0	16.7
L4	—	—	—	—	—	—	—	12.0	—	20.0	—	—	2.0	31.0	—	—	—	16.2
Permute Object Order; Extremely Distracting; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	4.0	11.0	19.5	0.0	0.0	8.5	4.0	—	2.0	—	0.5	25.0	—	—	3.0	0.0	0.0	6.0
L2	5.0	12.5	13.5	0.0	0.0	9.0	7.5	—	1.0	—	2.0	36.5	—	—	4.5	0.0	0.5	7.1
L3	7.0	13.0	21.5	0.0	0.0	7.0	6.0	—	1.5	—	1.5	—	—	—	4.0	0.0	0.0	5.1
L4	—	—	—	—	—	—	—	8.0	—	0.0	—	—	1.0	4.0	—	—	—	3.2
Concatenate + Obj-Centric																		
L1	3.5	10.0	8.5	0.0	0.0	3.5	5.0	—	3.0	—	4.5	13.5	—	—	1.5	0.5	0.0	4.1
L2	2.5	8.5	9.5	0.0	0.0	7.0	6.0	—	0.0	—	4.0	16.5	—	—	3.0	1.5	0.0	4.5
L3	3.5	6.5	13.0	0.0	0.0	4.5	6.0	—	0.0	—	6.0	—	—	—	3.0	0.0	0.0	3.5
L4	—	—	—	—	—	—	—	8.0	—	1.0	—	—	0.0	2.0	—	—	—	2.8

Table F.12: Per-task average success rate when evaluating performance for models with **permuted object order per observation across each difficulty level**. All models are trained **on paraphrased instructions** and **can recover from mistakes** during inference. This table corresponds to the middle section of Table 8, and also provides per-task results for the *Extremely Distracting* difficulty level.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Default Difficulty; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	84.5	94.0	16.0	90.5	10.0	60.5	82.5	—	10.5	—	93.0	91.5	—	—	92.5	48.0	2.5	59.7
L2	31.5	41.0	16.0	73.0	7.0	68.0	24.5	—	5.5	—	93.0	86.0	—	—	53.0	46.5	2.0	42.1
L3	42.5	55.0	19.0	46.5	4.0	54.0	42.0	—	8.0	—	93.0	—	—	—	49.5	42.5	1.0	38.1
L4	—	—	—	—	—	—	—	23.5	—	0.0	—	—	0.0	34.0	—	—	—	14.4
Concatenate + Obj-Centric																		
L1	96.0	99.0	46.5	93.0	22.5	63.0	89.0	—	78.0	—	95.0	87.5	—	—	97.5	44.5	6.0	70.6
L2	35.5	44.5	52.0	88.0	16.5	54.0	20.5	—	39.5	—	94.0	87.5	—	—	65.0	47.5	4.0	49.9
L3	37.0	63.0	49.0	76.5	12.0	39.5	27.0	—	60.5	—	94.5	—	—	—	32.0	45.0	0.0	44.7
L4	—	—	—	—	—	—	—	29.0	—	6.0	—	—	0.0	23.0	—	—	—	14.5
Distracting Difficulty; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	31.0	71.0	1.0	7.5	0.0	20.5	38.0	—	4.0	—	92.5	82.0	—	—	92.5	45.0	0.0	37.3
L2	1.0	11.5	0.5	4.5	0.0	6.5	1.0	—	1.5	—	93.5	75.5	—	—	27.0	49.0	0.0	20.9
L3	7.5	31.0	0.5	3.5	0.0	13.5	8.0	—	3.5	—	88.0	—	—	—	13.0	44.5	0.0	17.8
L4	—	—	—	—	—	—	—	8.5	—	0.0	—	—	0.0	2.5	—	—	—	2.8
Concatenate + Obj-Centric																		
L1	42.0	93.0	6.5	0.0	0.0	26.5	34.5	—	27.5	—	93.0	73.0	—	—	71.0	46.5	0.0	39.5
L2	0.5	15.0	7.0	0.0	0.0	15.5	1.5	—	6.5	—	91.5	81.0	—	—	20.0	40.5	0.0	21.5
L3	7.5	27.0	7.0	2.0	0.0	8.0	6.0	—	15.0	—	93.5	—	—	—	2.0	46.0	0.0	17.8
L4	—	—	—	—	—	—	—	11.5	—	1.0	—	—	0.0	2.0	—	—	—	3.6
Extreme Difficulty; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	32.0	41.5	17.5	81.5	9.5	17.5	39.5	—	11.5	—	5.0	52.5	—	—	22.5	0.0	2.0	25.6
L2	5.5	22.5	9.5	74.0	5.5	20.0	3.5	—	6.5	—	2.0	56.0	—	—	15.5	0.0	2.5	17.2
L3	20.5	26.5	19.0	45.0	6.5	20.5	22.0	—	9.5	—	2.0	—	—	—	21.0	0.5	1.5	16.2
L4	—	—	—	—	—	—	—	9.5	—	1.0	—	—	0.5	24.0	—	—	—	8.8
Concatenate + Obj-Centric																		
L1	17.5	47.5	31.0	93.0	17.5	29.0	22.0	—	73.0	—	9.5	51.0	—	—	21.5	0.5	2.5	32.0
L2	21.5	42.0	37.0	93.0	15.5	31.0	11.5	—	38.5	—	9.0	45.5	—	—	27.0	0.5	2.5	28.8
L3	28.5	49.5	37.0	73.0	13.0	24.5	22.0	—	56.5	—	7.0	—	—	—	33.5	0.0	0.0	28.7
L4	—	—	—	—	—	—	—	14.5	—	18.5	—	—	0.0	19.5	—	—	—	13.1
Extremely Distracting Difficulty; Can Recover From Mistakes																		
Cross-Attn + Obj-Centric																		
L1	5.0	18.5	0.5	0.0	0.0	9.5	12.5	—	3.0	—	2.0	48.0	—	—	9.5	1.0	0.0	8.4
L2	1.5	10.5	0.5	0.0	0.0	5.5	0.0	—	0.5	—	1.5	46.0	—	—	2.0	0.0	0.0	5.2
L3	7.0	10.0	1.5	0.0	0.0	10.5	6.5	—	1.0	—	1.0	—	—	—	2.5	0.0	0.0	3.3
L4	—	—	—	—	—	—	—	7.5	—	0.0	—	—	1.0	0.5	—	—	—	2.2
Concatenate + Obj-Centric																		
L1	3.0	35.0	5.5	0.0	0.0	15.0	4.5	—	29.0	—	11.0	50.5	—	—	1.0	0.0	0.0	11.9
L2	5.0	30.5	3.5	0.0	0.0	12.0	2.5	—	7.0	—	9.0	49.0	—	—	2.5	0.0	0.5	9.3
L3	4.0	28.0	7.0	0.0	0.0	6.5	6.0	—	12.0	—	7.5	—	—	—	3.5	0.5	0.0	6.2
L4	—	—	—	—	—	—	—	7.5	—	5.0	—	—	0.0	0.5	—	—	—	3.2

Table F.13: Per-task average success rate when evaluating performance for models *trained* **with a randomised object order per observation** for each difficulty level. Models **can recover from mistakes** and are trained **with paraphrased instructions**.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T15	T16	T17	Avg.
Permute Object Order; Default Difficulty; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	69.0	85.0	5.0	90.0	3.5	37.5	71.5	—	5.5	—	92.0	85.5	—	—	63.5	46.0	0.0	50.3
L2	18.5	31.0	5.0	73.0	2.0	38.5	16.5	—	3.0	—	92.0	75.0	—	—	43.0	46.0	0.0	34.1
L3	32.5	44.0	10.5	46.5	2.0	31.0	27.0	—	3.5	—	92.0	—	—	—	29.5	42.5	0.0	30.1
L4	—	—	—	—	—	—	—	15.5	—	0.0	—	—	0.0	24.5	—	—	—	10.0
Concatenate + Obj-Centric																		
L1	73.0	96.0	24.0	92.5	11.0	39.5	75.5	—	70.5	—	92.5	72.5	—	—	68.5	44.0	0.0	58.4
L2	24.0	39.5	30.0	87.5	7.5	39.5	14.5	—	26.5	—	91.5	75.5	—	—	51.5	46.0	0.0	41.0
L3	20.0	42.0	28.5	75.5	8.0	19.0	14.0	—	46.5	—	93.5	—	—	—	22.0	45.0	0.0	34.5
L4	—	—	—	—	—	—	—	18.5	—	0.0	—	—	0.0	14.0	—	—	—	8.1
Permute Object Order; Distracting Difficulty; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	17.5	63.5	0.5	4.0	0.0	10.5	25.0	—	2.0	—	90.5	68.5	—	—	75.0	42.5	0.0	30.7
L2	0.0	8.5	0.0	2.5	0.0	6.0	0.0	—	1.0	—	92.0	63.5	—	—	24.0	47.0	0.0	18.8
L3	4.0	22.0	0.0	3.0	0.0	9.0	4.0	—	2.0	—	87.5	—	—	—	7.5	44.5	0.0	15.3
L4	—	—	—	—	—	—	—	3.5	—	0.0	—	—	0.0	2.5	—	—	—	1.5
Concatenate + Obj-Centric																		
L1	33.5	85.0	3.0	0.0	0.0	13.5	22.5	—	25.0	—	82.5	61.5	—	—	61.5	45.5	0.0	33.3
L2	0.0	14.0	5.5	0.0	0.0	8.0	0.0	—	3.5	—	87.5	62.5	—	—	15.5	39.5	0.0	18.2
L3	4.5	21.5	5.0	0.0	0.0	4.5	4.5	—	11.5	—	91.0	—	—	—	1.5	46.0	0.0	15.8
L4	—	—	—	—	—	—	—	7.5	—	0.0	—	—	0.0	1.0	—	—	—	2.1
Permute Object Order; Extreme Difficulty; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	2.5	17.0	7.5	81.5	3.5	4.0	3.5	—	10.0	—	4.5	34.5	—	—	13.5	0.0	0.0	14.0
L2	0.0	4.0	1.5	73.5	2.5	5.5	0.0	—	3.0	—	1.5	42.5	—	—	12.0	0.0	0.0	11.2
L3	3.5	10.0	4.5	44.5	3.5	4.0	3.5	—	5.0	—	0.5	—	—	—	16.5	0.5	0.0	8.0
L4	—	—	—	—	—	—	—	2.5	—	0.0	—	—	0.5	15.0	—	—	—	4.5
Concatenate + Obj-Centric																		
L1	8.5	36.0	14.0	93.0	9.0	9.5	12.0	—	68.5	—	4.0	39.5	—	—	16.5	0.5	0.0	23.9
L2	0.0	18.0	18.0	93.0	7.5	6.5	0.0	—	22.0	—	3.0	38.0	—	—	17.0	0.5	0.0	17.2
L3	4.0	28.0	18.5	72.0	8.5	6.0	7.0	—	45.0	—	1.5	—	—	—	19.0	0.0	0.0	17.5
L4	—	—	—	—	—	—	—	7.0	—	5.5	—	—	0.0	11.0	—	—	—	5.9
Permute Object Order; Extremely Distracting Difficulty; No Mistakes Allowed																		
Cross-Attn + Obj-Centric																		
L1	0.0	10.5	0.0	0.0	0.0	1.0	1.5	—	1.5	—	2.0	33.0	—	—	8.5	0.5	0.0	4.5
L2	0.0	4.0	0.0	0.0	0.0	1.5	0.0	—	0.5	—	0.5	34.5	—	—	2.0	0.0	0.0	3.3
L3	1.5	3.5	0.5	0.0	0.0	2.0	0.5	—	0.0	—	0.5	—	—	—	1.5	0.0	0.0	0.8
L4	—	—	—	—	—	—	—	1.0	—	0.0	—	—	1.0	0.5	—	—	—	0.6
Concatenate + Obj-Centric																		
L1	0.0	27.0	3.0	0.0	0.0	5.0	3.0	—	26.5	—	3.0	36.0	—	—	1.0	0.0	0.0	8.0
L2	0.0	14.5	2.0	0.0	0.0	3.0	0.0	—	3.0	—	4.5	38.5	—	—	2.0	0.0	0.0	5.2
L3	1.5	15.0	3.0	0.0	0.0	1.0	0.5	—	10.0	—	2.0	—	—	—	3.0	0.5	0.0	3.0
L4	—	—	—	—	—	—	—	1.5	—	0.0	—	—	0.0	0.5	—	—	—	0.5

Table F.14: Per-task average success rate for models *trained* **with a randomised object order per observation** for each difficulty level, and then **evaluated with a permuted object order**. Models are trained **with paraphrased instructions** and are **not allowed to make mistakes** during evaluation.