

LLaRA: SUPERCHARGING ROBOT LEARNING DATA FOR VISION-LANGUAGE POLICY

Anonymous authors
Paper under double-blind review

ABSTRACT

LLMs with visual inputs, *i.e.*, Vision Language Models (VLMs), have the capacity to process state information as visual-textual prompts and respond with policy decisions in text. We propose LLaRA: Large Language and Robotics Assistant, a framework that formulates robot action policy as conversations and provides improved action outputs when trained with auxiliary data that complements policy learning. We first introduce an automated pipeline to generate *conversation-style instruction tuning* data from existing behavior cloning data. Then we enrich the dataset in a self-supervised fashion by formulating six auxiliary tasks. A VLM finetuned with the resulting collection of datasets can generate meaningful robot action policy decisions. Our experiments across multiple simulated and real-world environments demonstrate the state-of-the-art performance of the proposed LLaRA framework. The code, datasets, and pretrained models will be made publicly available.

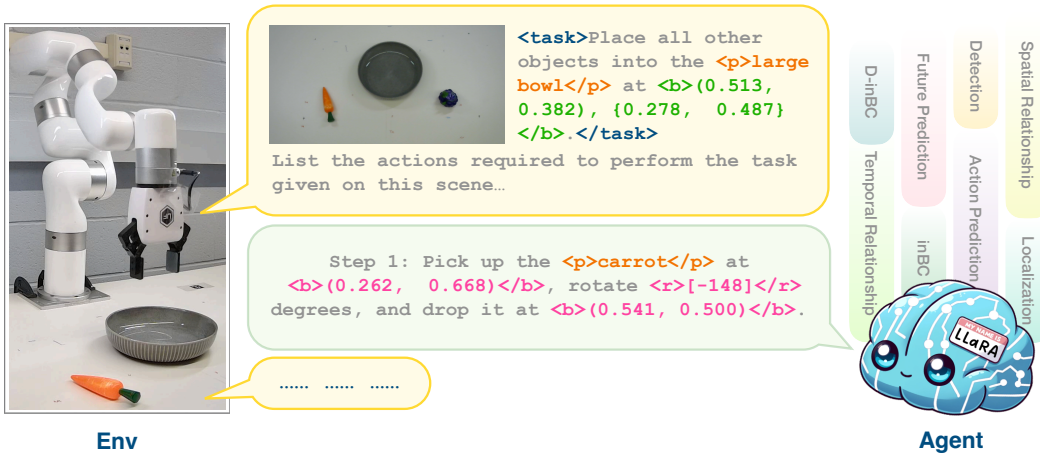


Figure 1: A real-world demonstration of LLaRA solving an unseen task. In the training stage, expert trajectories are converted into instruction-tuning data with image coordinates. Additionally, auxiliary data is generated to enhance the policy in a self-supervised manner. Using this data, we finetune a pretrained Vision Language Model (VLM) as a robot policy. At inference, the VLM is queried to generate actions in natural language.

1 INTRODUCTION

Large Language Models (LLMs) such as GPT-4 (OpenAI, 2023), Llama (Touvron et al., 2023; AI, 2024), and Gemini (Anil et al., 2023) exhibit unprecedented abilities across diverse language tasks. Such LLMs are often trained together with visual data by taking advantage of pretrained vision encoders, forming powerful Vision Language Models (VLMs). Effective and efficient training of such VLMs significantly depends on the style and format of the vision-language data used for training, leading to the exploration of various *instruction tuning* (Taori et al., 2023; Chiang et al., 2023; Liu et al., 2023a; 2024b) strategies. The resulting VLMs exhibit strong vision-language skills, but not without limitations such as spatial awareness (Chen et al., 2023; Ranasinghe et al., 2024b)

054 or niche-domain understanding. The latter has led to domain-specific visual instruction tuning (Li
055 et al., 2023; Kuckreja et al., 2024; Hong et al., 2023).

056 Several robot learning approaches also leverage pretrained LLMs / VLMs (Zeng et al., 2022; Brohan
057 et al., 2023c; Driess et al., 2023; Vemprala et al., 2024; Kim et al., 2024; Niu et al., 2024) with
058 recent work focusing on low-level robot actions (Brohan et al., 2023b;a; Padalkar et al., 2023).
059 The strong performance of such approaches can be attributed to the extensive world knowledge
060 (e.g., understanding of physics, human common-sense) (Yu et al., 2023; Zhao et al., 2024) and
061 powerful reasoning abilities (Creswell & Shanahan, 2022; Liu et al., 2023b) of underlying language
062 models. However, studies on curating and using conversation-style data for instruction tuning in
063 robotics have been rather limited (Zhen et al., 2024).

064 Motivated by the promising attributes of VLMs, we explore a formulation of VLM-based robot
065 action policy in this paper. Our goal is to adapt a VLM as a robot action policy that can handle diverse
066 visuomotor control challenges, a process we call *Visuomotor Instruction Tuning*. More specifically,
067 we transform a typical behavior cloning (BC) dataset into an instruction dataset, *Instruct-BC (inBC)*,
068 which is then used to finetune a VLM. Given a state described in visual-textual modalities, our
069 VLM is trained to generate suitable actions as text. Such a formulation based on conversation-style
070 instruction-response data enables us to instruction-tune and convert a VLM into a robot action policy
071 effortlessly. However, we also note that the effectiveness of an instruction-tuned VLM depends
072 heavily on the quality of data formulation (Liu et al., 2023a; Ranasinghe et al., 2024b; Kuckreja
073 et al., 2024), which is non-trivial to automate (or, scale) for new domains (Li et al., 2023; Kuckreja
074 et al., 2024; Hong et al., 2023; Chen et al., 2023; Ranasinghe et al., 2024b). Therefore, to strengthen
075 (or, *supercharge*) such data, we construct auxiliary datasets that complement policy learning from
076 the same BC dataset, in a self-supervised fashion without any extra action label.

077 Our overall framework termed **LLaRA (Large Language and Robotics Assistant)** generates visuo-
078 motor instruction data that can efficiently and effectively finetune a VLM into a robot action policy.
079 It is motivated by LLaVA (Liu et al., 2023a) which is designed primarily for vision tasks. Similarly,
080 LLaRA offers the entire framework of data generation, model formulation, and training pipeline for
081 VLMs, now specialized for robot learning. Our key contributions are:

- 082 1. Formulating robot manipulation tasks into instruction-response pairs described by natural
083 language, which enables successful instruction tuning of a VLM as a policy.
- 084 2. Aligning robot actions to image coordinates for VLMs, making the transfer of a pretrained
085 VLM to the robotics domain easier, particularly when the training data is limited.
- 086 3. Identifying and generating auxiliary instruction data that further enhances robot policy
087 learning in a self-supervised manner.

088
089 We conduct extensive experiments on the proposed framework to establish the effectiveness of both
090 our automated data generation pipeline and instruction-tuned VLM in solving robotics tasks.

092 2 RELATED WORK

093
094 **Instruction data generation.** Finetuning LLMs / VLMs with instruction data has shown a lot of
095 potential for vision tasks (Taori et al., 2023; Chiang et al., 2023; Liu et al., 2023a; 2024b). While
096 effective construction of such datasets with high quality and quantity is beyond trivial (Liu et al.,
097 2023a), especially for specialized visual domains (Li et al., 2023; Bazi et al., 2024; Kuckreja et al.,
098 2024; Thawkar et al., 2023) such as robotics. Yet these modified VLMs suffer in the robotics do-
099 mains (Ranasinghe et al., 2024a).

100 **Spatial reasoning in VLMs.** Several recent works investigate how VLMs can be modified for
101 spatial awareness within images (Zhang et al., 2023; Zhao et al., 2023; Zang et al., 2023; Peng et al.,
102 2023; Chen et al., 2023; Ranasinghe et al., 2024b; You et al., 2023; Cai et al., 2024; Shtedritski et al.,
103 2023), which is a critical ability of a visuomotor policy. One line of work explores prompting using
104 textual or specialized tokens to encode locations, followed by instruction tuning on localization-
105 specific datasets (Peng et al., 2023; Chen et al., 2023; Ranasinghe et al., 2024b; You et al., 2023). In
106 contrast, our framework extends beyond localization to directly predict robot actions. An alternate
107 line of work explores visual prompting but without any focus on robot learning (Shtedritski et al.,
2023; Cai et al., 2024). An extension of such ideas to robotics is explored in PIVOT (Nasiriany

108 et al., 2024). These ideas are complementary to LLaRA, where additional visuomotor instruction
109 tuning directly generates actions optimal for robotics tasks.

110 **Robot learning via sequence models, LLMs, and VLMs.** The use of sequence models in robot
111 learning has a longstanding tradition (Zheng et al., 2022; Shang et al., 2022a;b). Recent foundation
112 LLMs / VLMs provide a source of common-sense data for robot policy training (Qian et al., 2024;
113 Ingelhart et al., 2024; Wu et al., 2023b; Yoneda et al., 2023). More recent work such as Robotics
114 Transformer (Brohan et al., 2023b;a; Padalkar et al., 2023), Gato (Reed et al., 2022), GR-1 (Wu
115 et al., 2023a) and Octo (Octo Model Team et al., 2024) learn generalist robot policies with multi-
116 modal sequence models, taking advantage of the flexibility of encoding and decoding images/actions
117 via token representations similar to VLMs.

118 PALM-E (Driess et al., 2023) utilizes a pretrained VLM for embodied reasoning and planning. How-
119 ever, it assumes access to additional policies capable of performing low-level skills from a limited
120 vocabulary, with the VLM generating natural language that conditions these low-level commands.
121 Our method requires neither a language-conditioned low-level controller, which can be expensive
122 to learn for each environment, nor a lot of training data to achieve good performance. In addition,
123 because we use 2D image coordinates to present position, one can easily transfer our model to an-
124 other embodiment/domain with limited training samples. SpatialVLM (Chen et al., 2024) focuses
125 on enhancing the spatial reasoning ability of the VLM, resulting in a VLM that serves as a spatial re-
126 lationship predictor rather than a robot policy. To embed SpatialVLM into a policy, chain-of-thought
127 spatial reasoning is required. Our method explicitly transforms a pretrained VLM into a robot policy
128 that directly generates robot actions, resulting in a simpler framework and no need for an additional
129 policy. In addition, SpatialVLM studies only the spatial relationships between objects. Our method
130 also learns temporal environment dynamics through action prediction and future prediction datasets,
131 making our VLM better prepared as a robot policy.

132 There are concurrent works (Kim et al., 2024; Yuan et al., 2024; Niu et al., 2024) which also utilize
133 a LLaVA style VLM as a robot policy. All such approaches employ a VLM as a robot action policy,
134 processing both visual observations and textual instructions as inputs. OpenVLA (Kim et al., 2024)
135 employs specialized tokens to represent quantized actions, a method akin to RT-2 (Brohan et al.,
136 2023a). OpenVLA also incorporates DINOv2 (Oquab et al., 2023) as an additional visual encoder
137 along with SigLIP (Zhai et al., 2023), compared to a single CLIP (Radford et al., 2021) vision en-
138 coder used in LLaVA. RoboPoint (Yuan et al., 2024) introduces a novel point-based action space and
139 a scalable data pipeline tailored for spatial affordance prediction. Actions are delineated via points
140 on an RGB image, which are subsequently translated to 3D space using depth data. This approach
141 eliminates the reliance on predefined action primitives (Liang et al., 2023; Singh et al., 2023), external
142 object detectors (Huang et al., 2023b; Liu et al., 2024a), and iterative visual prompting (Nasiriany
143 et al., 2024). Another concurrent work, LLARVA (Niu et al., 2024), differs by generating both 2D
144 visual traces in image coordinates and corresponding textual actions as outputs, with the former
145 functioning as an auxiliary task. In contrast, our method describes robot action using image coordi-
146 nates in natural language, which creates a more intuitive link between vision and action without
147 modifying the existing VLM model architecture. Moreover, all the aforementioned studies lack of
148 comprehensive investigation into the methods for generating *auxiliary datasets* from existing robot
149 data, as well as the implications of integrating such datasets. Our work LLaRA distinctively ad-
150 dresses this gap, marking a critical divergence from the concurrent studies. That is, in addition to
151 directly supervising a VLM based on robot actions, the LLaRA framework automatically generates
152 multiple auxiliary datasets with complementary training objectives from the existing robot trajec-
153 tories. Such auxiliary datasets are designed to train a VLM to capture spatial/temporal relations
154 between objects in the scene, enabling better scene understanding. We experimentally confirm that
155 such additional supervision significantly benefits robot action policy learning.

156 **Self-supervised learning in robotics.** Self-supervised learning (SSL) has long been explored in
157 robotics (Pinto & Gupta, 2016; Sermanet et al., 2018; Yarats et al., 2021; Laskin et al., 2020;
158 Schwarzer et al., 2021), with recent work showing a flavor of visual representation learning (Li
159 et al., 2022; 2024b; Wang et al., 2023). We introduce auxiliary SSL objectives (*i.e.*, pretext tasks)
160 based on instruction prompts, motivated by similar ideas in computer vision (Doersch et al., 2015;
161 Yun et al., 2022; Ranasinghe et al., 2022; Walker et al., 2021). More specifically, we create a set of
domain-specific auxiliary datasets (*e.g.*, action or future prediction), that enable LLaRA to under-
stand information that is useful for the downstream robotics tasks, when used for finetuning.

3 PRELIMINARY: VISUAL INSTRUCTION TUNING

Visual Instruction Tuning is a framework to finetune a large Vision Language Model (VLM) using task-specific instructions. The objective is to develop a general-purpose multi-modal model capable of solving diverse vision tasks described by language (Huang et al., 2023a). A typical example of Visual Instruction Tuning is LLaVA (Liu et al., 2023a), which consists of three neural networks as its model: a large language model θ_{LLM} , a visual encoder θ_V and an adapter layer θ_{MLP} (see Fig. 2).

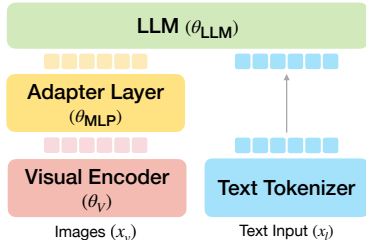


Figure 2: **LLaVA overview.** A Large Language Model (LLM) is connected to the visual domain with suitable encoder and adaptor neural networks.

Consider a conversation data sample that contains a single image x_v and user instruction text x_l as inputs. During inference, the visual encoder θ_V and adapter layer θ_{MLP} process x_v successively to produce a set of visual tokens that can be directly concatenated with the textual tokens of x_l in language embedding space of LLM θ_{LLM} . Next, θ_{LLM} autoregressively generates new tokens conditioned on both the visual tokens from x_v and text tokens from x_l . These new tokens will be decoded into natural language as the output of the model.

A two-phase training process is executed in Liu et al. (2023a) with a next-token prediction objective over the θ_{LLM} output in each phase. First, only θ_{MLP} is trained using image-caption pairs (similar to datasets for CLIP (Radford et al., 2021) training). Then, both θ_{MLP} and θ_{LLM} are finetuned using a specialized instruction tuning dataset termed LLaVA_Instruct_150k. LLaVA_Instruct_150k contains human-style instruction-response conversations, automatically created by a powerful LLM (OpenAI, 2023) from a generic object detection dataset; see (Liu et al., 2023a) for details. In our paper, we start from a pretrained LLaVA model and only perform a single-stage finetune, updating θ_{LLM} and θ_{MLP} .

4 VISUOMOTOR INSTRUCTION TUNING

We leverage large Vision Language Models (VLMs) as a generalist to address diverse visuomotor control challenges. Specifically, we transform a typical behavior cloning dataset into an instruction tuning dataset and subsequently finetune a pretrained VLM on this tailored dataset, a process we call *Visuomotor Instruction Tuning*. The resulting LLaRA framework benefits from the broad, inherent knowledge embedded within the VLM, enabling better visuomotor task learning.

In this section, we present our LLaRA framework to *a*) convert a set of robot manipulation expert trajectories into a visuomotor instruction tuning dataset, and *b*) finetune a VLM on such a dataset as a robot policy. In the next section, we show that *c*) such VLMs can be further improved for robot control using auxiliary instruction tuning data created in a self-supervised manner.

Problem formulation. We consider a behavioral cloning (BC) setting over a Markov Decision Process (MDP), described by the tuple $(\mathcal{S}, \mathcal{A}, P)$. Here, $s_t \in \mathcal{S}$ denotes the state at timestamp t , $a_t \in \mathcal{A}$ represents the action at t , and P encapsulates the transition dynamics, expressed as $P(s_{t+1}|s_t, a_t)$. Our goal is to finetune a VLM as a robot policy π that best recovers an unknown policy π^* using a demonstration dataset containing multiple tasks $\mathcal{D} = \{(s^i, a^i)\}$ collected by π^* . The policy π is designed to be conditioned on the task description, and our empirical findings suggest adding historical actions is a beneficial condition. Consequently, the policy can be articulated as $\pi(a_t^i|s_t^i, a_h^i)$, where $h = 1, 2, \dots, t-1$, indicating that π takes into account all preceding actions.

Instruction tuning data from trajectories. Next, we turn a set of expert trajectories \mathcal{D} into a visuomotor instruction tuning dataset. We focus on the stationary robot manipulation scenario featuring a robotic manipulator fixed above a flat table. The visual observation for this setup is captured by a fixed third-person view camera positioned to observe the table surface. For each state transition, we convert the state action pair (s_t, a_t) into a single-round conversation. The current visual observation and textual task description, forming s_t , can be directly assimilated by a VLM as the user instruction. However, the numerical action a_t needs conversion into textual format to be generated by a VLM. In contrast to the approach employed by RT-2 (Brohan et al., 2023a), which utilizes special tokens to directly encode numerical action values, we adopt *2D Image Coordinates*—normalized to $[0, 1]$ —to

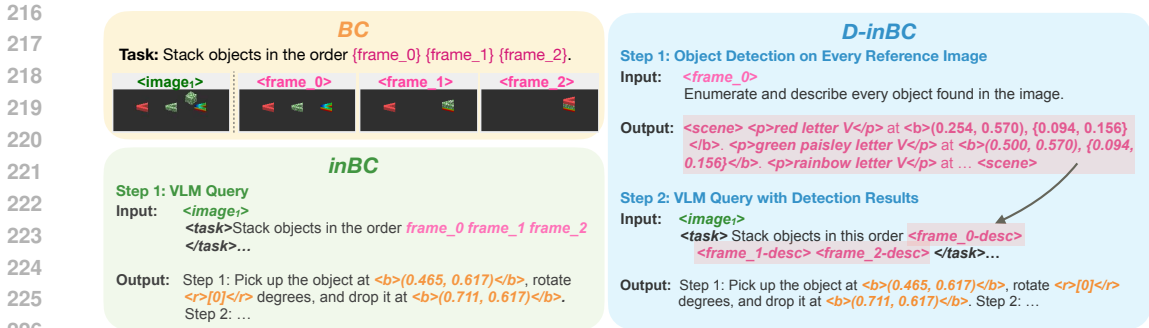


Figure 3: **Example data for visuomotor instruction tuning.** (left) *BC* is the original behavior cloning data (action omitted). *inBC* is our instruction tuning data created from *BC*. (right) *D-inBC* leverages object detection results on the additional images from the task description. Both *inBC* and *D-inBC* belong to the LLaRA family. represent the positions in actions. Additionally, any rotational component of the action, denoted as a_r (such as the angle of a joint), is expressed textually as "rotate $\langle r \rangle a_r \langle /r \rangle$ degrees". To further improve the performance, we incorporate the *history of executed actions* into the query and structure the response to predict *multiple future action steps* until the episode concludes. More details are covered in Appendix A.1.

Our approach creates an intuitive connection between visual input and robotic actions, enabling effortless transfer across various robotic embodiments. By implementing these strategies, we effectively convert a trajectory into a format comprising both image and text, which is readily processed by the VLM. We name the converted dataset **Instruct-BC** (*inBC*) and present examples in Fig. 3, Tab. 5 and Tab. 6.

Aggregate multiple images for a single-image VLM. The VLM policy we use here closely follows the design of LLaVA-1.5 (Liu et al., 2024b). The model consumes a *single* image and language instruction and generates language output. In scenarios where a single observation s_t comprises multiple images, systems such as LLaVA suffer from low performance because it was never trained on multiple images (see Tab. 11). Instead, we convert each additional image into a language description with the help of object detection on the *additional images* (e.g., images in the task description, rather than the main visual observation). The main image is still directly used as the visual input by the VLM without any changes. Compared to *inBC*, we name this method that takes additional object detection results as **Description-Instruct-BC** (*D-inBC*), and our model trained on this data is referred to with the same name as earlier. Examples are also available in Fig. 3, Tab. 5 and Tab. 6.

Inference pipeline. During inference, the prompt for the VLM is prepared using the same template as either *inBC* or *D-inBC*. As illustrated in Fig. 3, for a model trained on *inBC*, each conversation turn encapsulates the current visual observation, the task description, and the previous actions described in the text. For a model trained on *D-inBC*, the process starts with object detection to convert any *extra* images from the task description into text. The task description, now including the detection outcomes, is then combined with the current visual observation and the action history to get the complete query. Then LLaRA generates the text output that contains numerical values (e.g., 2D image coordinates) for a robot action. The extracted rotation angle can be directly mapped to the rotation of the end effector. The 2D image coordinates will be further converted to robot action space via predefined mapping, which can be estimated using visual calibration in both simulated and real-world environments. After that, the robot executes the action, and a new observation can be captured from the environment. This initiates another round of interaction, setting the stage for the subsequent action.

5 SUPERCHARGING VISUOMOTOR INSTRUCTION TUNING DATASET

Motivated by the success of LLaVA (Liu et al., 2023a) instruction tuning data, and subsequent domain-specific instruction tuning datasets (e.g., for reasoning, grounding, or referring) (Chen et al., 2023; Ranasinghe et al., 2024b; Peng et al., 2023), LLaRA creates auxiliary robotics instruction tuning datasets that enhance a VLM in a self-supervised manner.

More specifically, given an expert trajectory, we generate single-turn conversations that reveal useful information for learning a policy implicitly. In addition to visual observations, we make use of

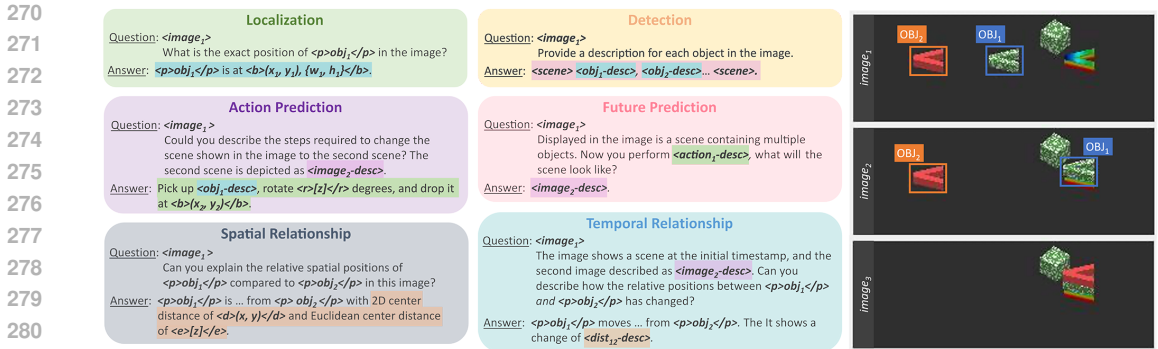


Figure 4: **Auxiliary datasets for visuomotor instruction tuning.** Given an input trajectory, we make use of expert information (e.g., object detections) to formulate conversations related to auxiliary semantics (left). Note the same template used among different answer types, highlighted with the same color. One detection demo is shown in the right figure. Please refer to Tab. 7 and Tab. 8 for the actual examples of each dataset.

object labels, locations, and geometries (e.g., rotation) together with task descriptions as expert (or oracle) information. We primarily introduce six auxiliary datasets, augmenting existing *inBC* or *D-inBC* to better finetune VLMs. In each dataset, we consider a standard instruction, which is further rephrased multiple times using GPT-4 (OpenAI, 2023) to infuse a reasonable diversity. The answers follow simple rule-based templates that format expert information as expected responses for each conversation. The following paragraphs briefly describe each auxiliary dataset. We kindly refer the readers to Fig. 4 and Appendix A.1.3 for formatting details and qualitative samples.

Localization and detection. To enhance the visual grounding ability of VLM, we first generate single-turn conversations for both localization of individual objects and higher-level detection of entire scenes. For localization, each conversation involves pinpointing an object within an observation, where the query specifies the object’s location and the response is given in the format “<p>OBJ</p> at (x, y), {w, h}”. Here, (x, y) indicates the midpoint of the object’s bounding box, while {w, h} denotes its width and height, with coordinates normalized to [0, 1] as detailed in Sec. 4. For detection, the setup extends to multiple objects, describing the scene via a list of bounding boxes.

Action and future prediction. These datasets are designed to improve the VLM’s capability to understand dynamics. In the action prediction dataset, given an initial observation, the query includes a text description of the subsequent observation, formatted similarly to the detection dataset. The VLM identifies the specific action that connects these two states. For future prediction, the roles of query and answer are reversed: the query now presents a single-step action and requests a text description of the next observation.

Spatial and temporal relationships. We further enrich the dataset with the relationships between objects. For spatial relationships, given two objects in an image, we format a conversation to address their 2D configuration (e.g., left, right, above, below) including Euclidean and axial distances, with an exemplar provided for better instruction comprehension by the VLM. For temporal relationships, the dataset examines changes in these spatial relationships over time (i.e., between two timesteps). This involves two observations—the initial in image form and the subsequent described in text—focusing on queries about movements (e.g., getting closer or further away), with responses detailing changes in Euclidean and axial distances.

By introducing these auxiliary datasets, we explicitly drive the VLM to strengthen the ability of spatial and temporal understanding, which benefits robot learning. Note that all these datasets can be automatically generated from *existing trajectories*, without introducing external task/action supervision. In Sec. 6, we empirically explore the best practices to apply these auxiliary datasets.

6 EXPERIMENTS

We conduct experiments in both simulated environments and the real world. For the simulated tasks, we tune a VLM on diverse tasks as a generalist. Then, we conduct real-world robot experiments using three protocols: *zero-shot generalization*, *finetuning*, and *joint training*.

6.1 SIMULATION EXPERIMENTS

Settings. We employ VIMA-Bench (Jiang et al., 2023), a simulated table-top robot manipulation environment to evaluate VLMs trained by our instruction tuning dataset. The environment contains 17 tasks and each task is associated with a multi-modal instruction, including text instructions and reference images that refer to objects of interest or a particular scene arrangement. The robot action space is two 2D coordinates for pick and place positions and two quaternions for rotations. We uniformly subsample the VIMA dataset (Jiang et al., 2023) to form three subsets with different sizes: VIMA-0.8k, VIMA-8k, and VIMA-80k where the number indicates the number of expert trajectories in the dataset.

Methods. We compare variants of our method with baselines that follow the recipe of RT-2 (Brohan et al., 2023a), *RT-2 Style*, and *D-RT-2 Style*. As introduced in Sec. 4, *inBC* is the dataset converted from the expert trajectories. Prefix *D* means the additional reference images in the task description are described by object detection results in text and the object detection will be performed on the reference images first during inference. Suffix *Aux* means the auxiliary datasets introduced in Sec. 5 are included in the training set, and the letter after it (e.g., *(D)*) indicates the detailed configuration of the auxiliary dataset (see Tab. 2). *Oracle* means that the groundtruth bounding box of objects is used as the object detection results only in the reference images. *RT-2 Style* is similar to *inBC* but the output of the VLM (i.e., robot actions) is a set of special tokens that can be directly mapped to quantized actions. Specifically, in VIMA, the action space is two positions in *robot* coordinates and one rotation. *D-RT-2 Style (I)* is similar to *D-RT-2 Style* but the output tokens present quantized two positions in *image* coordinates and one rotation angle, which is the quantized version of what LLaRA generates in natural language, presented by the same set of special tokens.

We train all methods on these three datasets and evaluate them with 3 levels of difficulties following the test protocol (L1 to L3). Due to missing data in the original VIMA dataset, we were not able to evaluate our approach on L4 properly. For each task, we evaluate all the methods with 20 random seeds and report the average success rates of each level.

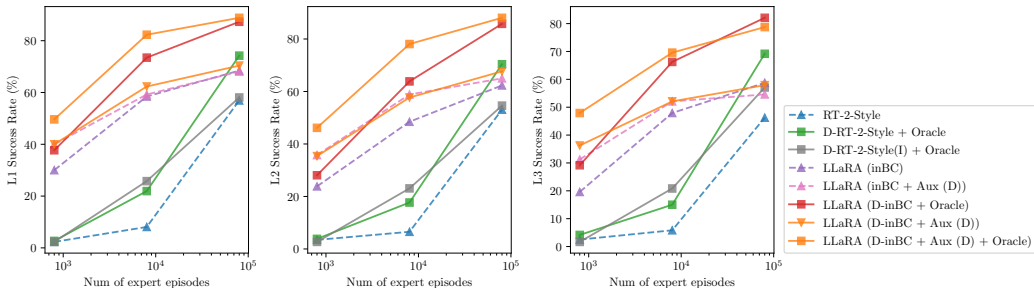


Figure 5: Performance on three VIMA subsets. x-axis is the number of expert episodes in the training set.

Effectiveness of LLaRA framework. Figure 5 illustrates the performance of selected methods across various dataset scales. All methods generally benefit from increased training data. Key observations include:

- *inBC* consistently surpasses the *RT-2 Style* baseline, and similarly, *D-inBC* outperforms *D-RT-2 Style*. This supports the effectiveness of our approach, which utilizes instruction-tuning style data as well as image coordinates. It is easier to benefit from the existing VLMs pretrained in a conversation style. Methods based on *RT-2 Style* improve when more robot supervision data is available; however, they significantly underperform compared to our methods when data is limited. Also in Appendix B.2, we confirm even with 100% data, *RT-2 Style* still underperforms our method trained on only 12% data.
- *D-inBC* generally excels over *inBC* owing to its explicit delineation of object locations. This trend is similarly observed between *D-RT-2 Style* and *RT-2 Style*.
- Auxiliary datasets prove to be highly beneficial, particularly when the size of the training set is limited. However, in scenarios like VIMA-80k where the dataset is substantial, sometimes *D-inBC* performs better and the gain from the auxiliary datasets is marginal.

We hypothesize that when training data is adequate, the benefits of the auxiliary dataset diminish, and these data may even distract the model from effectively performing the behavior cloning tasks. It is advisable to carefully regulate the amount of auxiliary data when there is an abundance of expert episodes to avoid overwhelming the primary learning objectives. By default, we randomly sample from each auxiliary dataset so that the size of each auxiliary dataset is identical to *inBC* or *D-inBC*.

Comparison to VIMA. We first clarify the difference between our method and VIMA (Jiang et al., 2023) in terms of the inputs. VIMA takes both front and top view images from the environment while ours only takes the front view. We test the most capable model released by VIMA, which is trained on 660k expert trajectories, and the results are shown in Tab. 1. Compared to VIMA (Jiang et al., 2023), our best model not only achieves better performance but also requires less input and is trained on only 12% of the expert trajectories used in VIMA. The full comparison is in Tab. 9.

Table 1: Comparison to VIMA (Jiang et al., 2023). Our best model not only achieves better performance but also requires less input and is trained on only 12% of the data used in VIMA.

| Method | Config | Data | L1 (%) | L2 (%) | L3 (%) |
|--------------|----------------------------------|------|-------------|-------------|-------------|
| VIMA | VIMA-200M + Oracle | 100% | 80.7 | 81.9 | 77.9 |
| LLaRA (Ours) | <i>D-inBC</i> + Aux (B) + Oracle | 12% | 90.0 | 88.1 | 79.2 |

Ablation on auxiliary datasets. We next study the effectiveness of auxiliary datasets. Tab. 2 shows the different combinations of the auxiliary datasets we studied. For each setting, we randomly sample the same amount of examples from each auxiliary dataset and combine them with the converted behavior cloning dataset. The ‘*’ after a ‘✓’ means the reference images that appeared only in the task description are not used to generate this dataset, which makes the dataset less diverse. We train the model on both *inBC* and *D-inBC* with different auxiliary dataset settings. On VIMA-0.8k, we control the total number of samples from the auxiliary dataset relative to the samples from the converted BC datasets and train all models for 2 epochs. Results in Fig. 6 show that in general, on VIMA-0.8k, the model performs better with more auxiliary data. In Appendix B.3, we further validate that the model with auxiliary data benefits from the data instead of extra optimization steps.

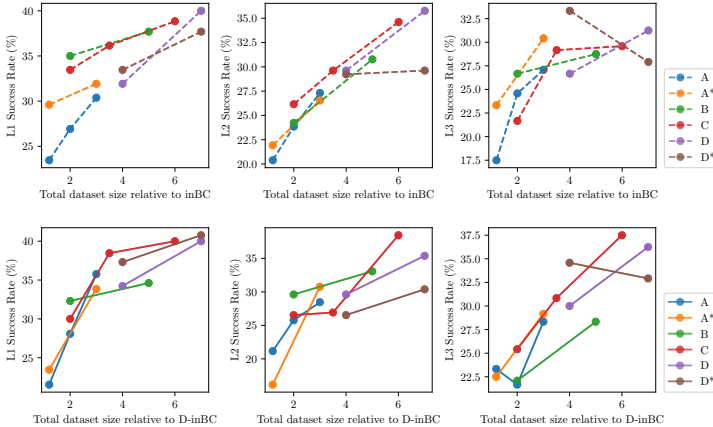


Figure 6: *inBC* (top) and *D-inBC* (bottom) with different auxiliary dataset settings. Each model is trained on VIMA-0.8k for 2 epochs. In general, the model performs better with more auxiliary data.

Table 2: Naming of different auxiliary dataset configurations. We always randomly sample the same amount of examples from each dataset. **Det.:** detection; **Loc.:** localization; **Act.:** action prediction; **Fut.:** future prediction; **Spa.:** spatial relationship; **Temp.:** temporal relationship.

| | Loc. | Det. | Act. | Fut. | Spa. | Temp. |
|----|------|------|------|------|------|-------|
| A | ✓ | ✓ | | | | |
| A* | ✓* | ✓* | | | | |
| B | ✓ | ✓ | ✓ | ✓ | | |
| C | ✓ | ✓ | ✓ | | ✓ | |
| D | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| D* | ✓* | ✓* | ✓ | ✓ | ✓* | ✓ |

A comprehensive ablation of other design choices is available in Appendix B.

6.2 REAL-WORLD ROBOT EXPERIMENTS

We further conduct zero-shot generalization, fine-tuning, and joint training experiments in a novel real-world environment, in which there is a robot arm with a gripper and a fixed RGB camera positioned above the arm to collect observations (see Fig. 7). The action space is the same as the one in VIMA. As all the objects are placed on a plain surface and the camera is fixed, similar to VIMA, a linear mapping between the image coordinates and the robot action space can be estimated by calibration. For the methods that rely on object detection *i.e.*, *D-inBC*, we employed one-shot detection using OWLv2 (Minderer et al., 2024).

We benchmark each policy on three tasks:

- **T1**: “Move the {object} into the large bowl.”
- **T2**: “Rotate the {object} by {angle} degrees.”
- **T3**: “Move the {object} on top of the tray.”

where the {object} is chosen from a set of 10 plastic toys and the {angle} in T2 is chosen between 0 to 180 degrees. In all tasks, all objects are placed randomly on the table before running an episode and they never appear in the training set. A success in T1 and T3 is if the object is placed more than 50% inside the bowl or on the tray. A success in T2 is if the object appears to be clockwise rotated by the angle by visual inspection. We run 20 randomly initialized episodes for each setting and report the average success rate. A visual reference for typical task start states and success ends are provided in Fig. 8.

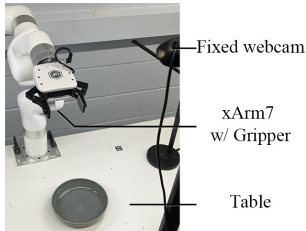


Figure 7: Our real-world robot setting.

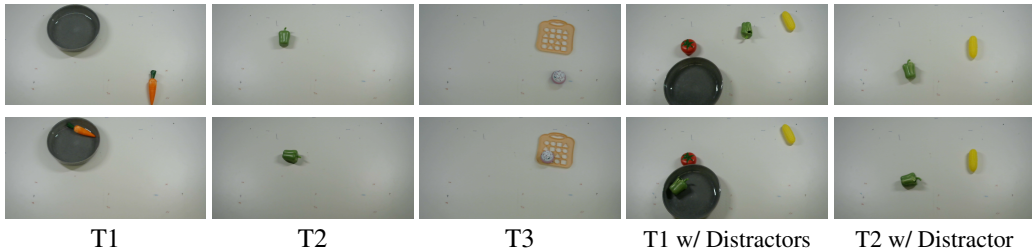


Figure 8: Visual reference for the initial image (top row) and successful end positions (bottom row) in three real-world tasks. We also show some examples that have distracting objects in the scene.

Table 3: Real-world robot experiment results. ‘Aux’ means trained on auxiliary datasets and ‘OD’ means using an object detector (OWLv2, base-16) to describe the observation before the VLM query. In **Protocol**, ‘ZS’ stands for zero-shot evaluation, ‘FT’ means the model is first trained on VIMA for 2 epochs and then finetuned on **Add. Data** for 1 epoch. ‘JT’ means the model is trained on the combination of VIMA and **Add. Data** for 2 epochs.

| Protocol | Method | Add. Data | T1 (%) | T2 (%) | T3 (%) | Avg. (%) |
|----------|------------------------------|-----------------------|-----------|------------|-----------|-------------|
| ZS | <i>RT-2 Style</i> | | 0 | 0 | 0 | 0 |
| | GPT-4o | | 20 | 45 | 30 | 31.6 |
| | <i>inBC</i> | | 40 | 50 | 20 | 36.6 |
| | LLaRA (Ours) | <i>inBC</i> + Aux (D) | 10 | 30 | 10 | 16.6 |
| | <i>D-inBC</i> + OD | 30 | 40 | 25 | 31.6 | |
| | <i>D-inBC</i> + Aux (C) + OD | 35 | 40 | 5 | 26.6 | |
| FT | <i>RT-2 Style</i> | xArm-Det | 0 | 0 | 0 | 0 |
| | <i>inBC</i> | xArm-Det | 0 | 0 | 0 | 0 |
| | LLaRA (Ours) | <i>inBC</i> | 30 | 45 | 5 | 26.6 |
| | <i>D-inBC</i> + OD | xArm-Action | 45 | 80 | 55 | 60 |
| | <i>D-inBC</i> + Aux (C) + OD | xArm-Det | 70 | 90 | 70 | 76.6 |
| | <i>D-inBC</i> + Aux (C) + OD | xArm-Action | 90 | 100 | 85 | 91.6 |
| JT | <i>inBC</i> | xArm-Action | 60 | 75 | 55 | 63.3 |
| | LLaRA (Ours) | <i>D-inBC</i> + OD | 65 | 95 | 45 | 68.3 |
| | <i>D-inBC</i> + OD | xArm-Det | 65 | 55 | 25 | 48.3 |
| | <i>D-inBC</i> + Aux (C) + OD | xArm-Action | 70 | 95 | 85 | 83.3 |
| | <i>D-inBC</i> + Aux (C) + OD | xArm-Det | 45 | 70 | 20 | 53.3 |

Zero-shot generalization. For zero-shot generalization and finetuning, we use the pretrained models from Sec. 6.1 only trained on simulated VIMA data. We also make the setting very challenging by selecting the pretrained model only using 1.2% of the VIMA training data (VIMA-8k). We benchmark the models without any further training, along with a new baseline, GPT-4o (OpenAI, 2023). The results are presented in the upper part of Tab. 3. In this setting, *inBC* achieves the best overall performance. We hypothesize that the auxiliary datasets used in *inBC* + Aux and *D-inBC* + Aux drive the model to focus more on the VIMA domain, leading to overfitting. Meanwhile, GPT-

486 4o achieves commendable performance, largely attributable to its extensive dataset and substantial
487 parameterization.

488 **Finetuning on real-world data.** In the finetuning setting, we first collect two in-domain real-
489 world datasets xArm-Det and xArm-Action (see Appendix C). Then the models from Sec. 6.1 are
490 further tuned on the new real-world datasets for 1 epoch and the evaluation results are presented
491 in the middle of Tab. 3. In general, finetuning outperforms other settings because it allows the
492 model to focus more on real-world data distribution, rather than be distracted by simulated VIMA
493 data. Auxiliary data contributes to a large boost in performance during finetuning, showing that
494 explicitly uncovering information from the existing in-domain data can benefit the model. It seems
495 *inBC* pretrained with VIMA is not trained to take advantage of such new in-domain robotics data,
496 performing worse than zero-shot. Similar to our findings in simulation experiments, LLaRA in
497 general benefits from pretrained VLM since LLaRA is trained on a dataset that has been organized
498 in a conversation style that is similar to what is used to pretrain the VLM. In contrast, *RT-2 Style*
499 requires a large amount of data so it suffers more from limited data in our real-world experiments.

500 **Joint training on both simulated and real-world data.** In the joint training setting, we combine
501 both VIMA data with xArm-Det or xArm-Action and tune a VLM jointly on both simulated and real-
502 world datasets. The full results are presented at the lower part of Tab. 3. In the joint training setting,
503 xArm-Det is equally or more beneficial than xArm-Action, highlighting the importance of detection
504 and localization information from the real-world setting. Auxiliary data proves most beneficial when
505 jointly trained with xArm-Det, achieving the highest performance across all joint training settings.
506 This observation suggests that careful exploration of auxiliary dataset usage is crucial to maximize
507 their benefits when joint training is performed.

508 Using image coordinates creates an intuitive and easy-to-learn link between visual observations and
509 robot actions. In general, LLaRA demonstrates a robust capacity for generalization. Our obser-
510 vations confirm that it takes much less effort and data to transfer LLaRA models pretrained on
511 simulated VIMA data to a real-world domain, while *RT-2 Style* suffers from the limited data.

512 **More generalization results.** We further validate the generalization ability of LLaRA on tasks with
513 distractors. There are three test settings and none of them is included in the simulated training set
514 or the dataset for fine-tuning: **T1 w/ distractor**: where we randomly put another random toy on
515 the table as a distractor; **T1 w/ complex rephrasing**: where we rephrase the task description in 10
516 different unseen ways using GPT-4; **Unseen task T4**: the task is “Weight the object using the scale
517 then place it back on the table.” which is never included in VIMA or xArm-Action. The results are
518 presented in Tab. 4.

519 Table 4: More generalization results of *D-inBC* + Aux (C). The model is tested on three unseen tasks using two
520 protocols and it shows strong performance with minimal finetuning on in-domain vision data.

| Protocol | T1 w/ distractor (%) | T1 w/ complex rephrasing (%) | T4 (%) |
|-------------------|----------------------|------------------------------|--------|
| ZS | 10 | 0 | 0 |
| FT on xArm-Action | 75 | 40 | 20 |

521
522
523
524
525 Our model demonstrates robust performance on generalized tasks, achieving a 75% success rate in
526 T1 with distractors. This suggests that our VLM retains its generalization ability even after being
527 trained on our instruction dataset. With a small amount of in-domain vision data, it can achieve
528 strong performance on previously unseen tasks.

529 We discuss all implementation details and qualitative results in Appendix C.

531 7 CONCLUSION

532
533 We present LLaRA, a framework that turns a pretrained vision language model (VLM) into a robot
534 policy using curated instruction tuning datasets. Firstly, we create a conversation-style instruction-
535 tuning data using existing behavior cloning data. Instruction tuning the VLM on this data confirms
536 the effectiveness of our framework, particularly when the training data is limited, thanks to our
537 formulation that aligns robot actions to image coordinates. Then we construct auxiliary datasets
538 using the same robot trajectory data through self-supervision, which proved to be further beneficial
539 to robot learning. Experiments across several synthetic environments and robot manipulation tasks
in the real world validate the effectiveness of our proposed LLaRA framework.

540 REPRODUCIBILITY STATEMENT

541
542 We have submitted the code for all experiments, except those specific to our robot hardware, as
543 part of the supplementary material. The code is anonymized, self-contained, and includes detailed
544 instructions to facilitate the replication of our experiments and findings. We also plan to publicly
545 release the code, data, pretrained models, and any additional resources needed for the community to
546 fully reproduce our work.

547
548 REFERENCES

- 549
550 Meta AI. Llama 3, 2024. URL <https://llama.meta.com/llama3/>. Accessed: 2024-06-
551 24.
- 552 Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Jo-
553 han Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal
554 models. *arXiv preprint arXiv:2312.11805*, 2023.
- 555 Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang
556 Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities.
557 *arXiv preprint arXiv:2308.12966*, 2023.
- 558
559 Yakoub Bazi, Laila Bashmal, Mohamad Mahmoud Al Rahhal, Riccardo Ricci, and Farid Melgani.
560 Rs-llava: A large vision-language model for joint captioning and question answering in remote
561 sensing imagery. *Remote Sensing*, 16(9):1477, 2024.
- 562
563 Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choroman-
564 ski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action
565 models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023a.
- 566
567 Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn,
568 Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian
569 Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalash-
570 nikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deek-
571 sha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl
572 Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag Sanketi,
573 Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent
574 Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and
575 Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. *Robotics science
576 and systems (RSS)*, 2023b.
- 577
578 Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho,
579 Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding
580 language in robotic affordances. In *Conference on Robot Learning (CoRL)*, pp. 287–318. PMLR,
581 2023c.
- 582
583 Mu Cai, Haotian Liu, Dennis Park, Siva Karthik Mustikovela, Gregory P. Meyer, Yuning Chai, and
584 Yong Jae Lee. Vip-llava: Making large multimodal models understand arbitrary visual prompts. In
585 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*,
586 2024.
- 587
588 Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia.
589 Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings
590 of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14455–
591 14465, 2024.
- 592
593 Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing
594 multimodal llm’s referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023.
- 595
596 Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng,
597 Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An
598 open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL <https://lmsys.org/blog/2023-03-30-vicuna/>.

- 594 Antonia Creswell and Murray Shanahan. Faithful reasoning using large language mod-
595 els. *ArXiv*, abs/2208.14271, 2022. URL [https://api.semanticscholar.org/
596 CorpusID:251929296](https://api.semanticscholar.org/CorpusID:251929296).
597
- 598 Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by
599 context prediction. In *Proceedings of the International Conference on Computer Vision (ICCV)*,
600 pp. 1422–1430, 2015.
- 601 Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter,
602 Ayzan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multi-
603 modal language model. *arXiv preprint arXiv:2303.03378*, 2023.
604
- 605 Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recog-
606 nition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition
607 (CVPR)*, pp. 770–778, 2016.
- 608 Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the
609 International Conference on Computer Vision (ICCV)*, pp. 2961–2969, 2017.
610
- 611 Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang
612 Gan. 3d-llm: Injecting the 3d world into large language models. In *Advances in Neural Informa-
613 tion Processing Systems (NeurIPS)*, 2023.
- 614 Jiaxing Huang, Jingyi Zhang, Kai Jiang, Han Qiu, and Shijian Lu. Visual instruction tuning towards
615 general-purpose multimodal model: A survey. *arXiv preprint arXiv:2312.16602*, 2023a.
616
- 617 Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer:
618 Composable 3d value maps for robotic manipulation with language models. *arXiv preprint
619 arXiv:2307.05973*, 2023b.
- 620 Nils Ingelhart, Jesper Munkeby, van Jonne Haastregt, Anastasia Varava, Michael C. Welle, and Dan-
621 ica Kragic. A robotic skill learning system built upon diffusion policies and foundation models.
622 *arXiv preprint arXiv:2403.16730*, 2024.
- 623 Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-
624 Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with mul-
625 timodal prompts. In *Proceedings of the International Conference on Machine Learning (ICML)*,
626 2023.
627
- 628 Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair,
629 Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source
630 vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- 631 Kartik Kuckreja, Muhammad S. Danish, Muzammal Naseer, Abhijit Das, Salman Khan, and Fa-
632 had S. Khan. Geochat: Grounded large vision-language model for remote sensing. *Proceedings
633 of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
634
- 635 Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised represen-
636 tations for reinforcement learning. In *Proceedings of the International Conference on Machine
637 Learning (ICML)*, pp. 5639–5650. PMLR, 2020.
- 638 Chunyuan Li, Cliff Wong, Sheng Zhang, Naoto Usuyama, Haotian Liu, Jianwei Yang, Tristan Nau-
639 mann, Hoifung Poon, and Jianfeng Gao. Llava-med: Training a large language-and-vision assis-
640 tant for biomedicine in one day. *arXiv preprint arXiv:2306.00890*, 2023.
- 641 Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma,
642 and Chunyuan Li. Llava-next: Tackling multi-image, video, and 3d in large
643 multimodal models, June 2024a. URL [https://llava-vl.github.io/blog/
644 2024-06-16-llava-next-interleave/](https://llava-vl.github.io/blog/2024-06-16-llava-next-interleave/).
645
- 646 Xiang Li, Jinghuan Shang, Srijan Das, and Michael S. Ryoo. Does self-supervised learning really
647 improve reinforcement learning from pixels? In *Advances in Neural Information Processing
Systems (NeurIPS)*, volume 35, pp. 30865–30881, 2022.

- 648 Xiang Li, Varun Belagali, Jinghuan Shang, and Michael S. Ryoo. Crossway diffusion: Improving
649 diffusion-based visuomotor policy via self-supervised learning. In *IEEE International Conference*
650 *on Robotics and Automation (ICRA)*, pp. 16841–16849. IEEE, 2024b.
- 651
- 652 Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence,
653 and Andy Zeng. Code as policies: Language model programs for embodied control. In *IEEE*
654 *International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500. IEEE, 2023.
- 655
- 656 Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr
657 Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of*
658 *the European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014.
- 659
- 660 Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic
661 manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024a.
- 662
- 663 Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances*
664 *in Neural Information Processing Systems (NeurIPS)*, 2023a.
- 665
- 666 Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction
667 tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*
668 *(CVPR)*, pp. 26296–26306, 2024b.
- 669
- 670 Xiao Liu, Da Yin, Chen Zhang, Yansong Feng, and Dongyan Zhao. The magic of if: Investigating
671 causal reasoning abilities in large language models of code. In *Annual Meeting of the Association*
672 *for Computational Linguistics*, 2023b. URL [https://api.semanticscholar.org/](https://api.semanticscholar.org/CorpusID:258968140)
673 [CorpusID:258968140](https://api.semanticscholar.org/CorpusID:258968140).
- 674
- 675 Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection.
676 *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024.
- 677
- 678 Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny
679 Driess, Ayzaan Wahid, Zhuo Xu, Quan Vuong, Tingnan Zhang, Edward Tsang-Wei Lee, Kuang-
680 Huei Lee Lee, Peng Xu, Sean Kirmani, Yuke Zhu, Andy Zeng, Karol Hausman, Nicolas Heess,
681 Chelsea Finn, Sergey Levine, and Brian Ichter. Pivot: Iterative visual prompting elicits actionable
682 knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.
- 683
- 684 Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor
685 Darrell, and Roei Herzig. Llarva: Vision-action instruction tuning enhances robot learning. *arXiv*
686 *preprint arXiv:2406.11815*, 2024.
- 687
- 688 Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep
689 Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang
690 Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine.
691 Octo: An open-source generalist robot policy. In *Robotics science and systems (RSS)*, Delft,
692 Netherlands, 2024.
- 693
- 694 OpenAI. GPT-4 technical report. <https://arxiv.org/abs/2303.08774>, 2023.
- 695
- 696 Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov,
697 Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning
698 robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- 699
- 700 Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander
701 Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic
learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.
- 702
- 703 Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu
704 Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint*
705 *arXiv:2306.14824*, 2023.
- 706
- 707 Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries
708 and 700 robot hours. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp.
709 3406–3413. IEEE, 2016.

- 702 Shengyi Qian, Weifeng Chen, Min Bai, Xiong Zhou, Zhuowen Tu, and Erran Li Li. Affordancellm:
703 Grounding affordance from vision language models. In *Proceedings of the IEEE/CVF Conference*
704 *on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- 705 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal,
706 Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual
707 models from natural language supervision. In *Proceedings of the International Conference on*
708 *Machine Learning (ICML)*, pp. 8748–8763. PMLR, 2021.
- 709 Kanchana Ranasinghe, Muzammal Naseer, Salman Hameed Khan, Fahad Shahbaz Khan, and
710 Michael S. Ryoo. Self-supervised video transformer. *Proceedings of the IEEE/CVF Conference*
711 *on Computer Vision and Pattern Recognition (CVPR)*, pp. 2864–2874, 2022.
- 712 Kanchana Ranasinghe, Xiang Li, Kumara Kahatapitiya, and Michael S. Ryoo. Understanding long
713 videos in one multimodal language model pass. *arXiv preprint arXiv:2403.16998*, 2024a.
- 714 Kanchana Ranasinghe, Satya Narayan Shukla, Omid Poursaeed, Michael S. Ryoo, and Tsung-Yu
715 Lin. Learning to localize objects improves spatial reasoning in visual-llms. In *Proceedings of the*
716 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024b.
- 717 Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov,
718 Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom
719 Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell,
720 Oriol Vinyals, Mahyar Bordbar, and Freitas de Nando. A generalist agent. In *Trans. on Machine*
721 *Learning Research*, 2022.
- 722 Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bach-
723 man. Data-efficient reinforcement learning with momentum predictive representations. In *Pro-*
724 *ceedings of the International Conference on Learning Representations (ICLR)*, 2021.
- 725 Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey
726 Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In
727 *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1134–1141. IEEE, 2018.
- 728 Jinghuan Shang, Kumara Kahatapitiya, Xiang Li, and Michael S. Ryoo. Starformer: Transformer
729 with state-action-reward representations for visual reinforcement learning. In *Proceedings of the*
730 *European Conference on Computer Vision (ECCV)*, pp. 462–479. Springer, 2022a.
- 731 Jinghuan Shang, Xiang Li, Kumara Kahatapitiya, Yu-Cheol Lee, and Michael S. Ryoo. Starformer:
732 Transformer with state-action-reward representations for robot learning. *IEEE Transactions on*
733 *Pattern Analysis and Machine Intelligence*, pp. 1–16, 2022b.
- 734 Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red
735 circle: Visual prompt engineering for vlms. In *Proceedings of the International Conference on*
736 *Computer Vision (ICCV)*, 2023.
- 737 Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter
738 Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using
739 large language models. In *IEEE International Conference on Robotics and Automation (ICRA)*,
740 pp. 11523–11530. IEEE, 2023.
- 741 Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy
742 Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model.
743 https://github.com/tatsu-lab/stanford_alpaca, 2023.
- 744 Omkar Thawkar, Abdelrahman Shaker, Sahal Shaji Mullappilly, Hisham Cholakkal, Rao Muham-
745 mad Anwer, Salman Khan, Jorma Laaksonen, and Fahad Shahbaz Khan. Xraygpt: Chest radio-
746 graphs summarization using medical vision-language models. *arXiv: 2306.07971*, 2023.
- 747 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko-
748 lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open founda-
749 tion and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

- 756 Sai H Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design
757 principles and model abilities. *IEEE Access*, 2024.
758
- 759 Jacob Walker, Ali Razavi, and Aäron van den Oord. Predicting video with vqvae. *ArXiv*,
760 abs/2103.01950, 2021. URL [https://api.semanticscholar.org/CorpusID:
761 232092596](https://api.semanticscholar.org/CorpusID:232092596).
- 762 Jianren Wang, Sudeep Dasari, Mohan Kumar Srirama, Shubham Tulsiani, and Abhinav Gupta. Ma-
763 nipulate by seeing: Creating manipulation controllers from pre-trained representations. In *Pro-
764 ceedings of the International Conference on Computer Vision (ICCV)*, pp. 3859–3868, 2023.
765
- 766 Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu,
767 Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot
768 manipulation. *arXiv preprint arXiv:2312.13139*, 2023a.
- 769 Yue Wu, Yewen Fan, Paul Pu Liang, Amos Azaria, Yuanzhi Li, and Tom M Mitchell. Read and reap
770 the rewards: Learning to play atari with the help of instruction manuals. In *Advances in Neural
771 Information Processing Systems (NeurIPS)*, 2023b.
772
- 773 Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improv-
774 ing sample efficiency in model-free reinforcement learning from images. In *Proceedings of the
775 AAAI Conference on Artificial Intelligence (AAAI)*, pp. 10674–10681, 2021.
776
- 777 Takuma Yoneda, Jiading Fang, Peng Li, Huanyu Zhang, Tianchong Jiang, Shengjie Lin, Ben Picker,
778 David Yunis, Hongyuan Mei, and Matthew R Walter. Statler: State-maintaining language models
779 for embodied reasoning. *arXiv preprint arXiv:2306.17840*, 2023.
- 780 Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao,
781 Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any gran-
782 ularity. *ArXiv*, abs/2310.07704, 2023. URL [https://api.semanticscholar.org/
783 CorpusID:263834718](https://api.semanticscholar.org/CorpusID:263834718).
- 784 Jifan Yu, Xiaozhi Wang, Shangqing Tu, Shulin Cao, Daniel Zhang-Li, Xin Lv, Hao Peng, Zijun
785 Yao, Xiaohan Zhang, Hanming Li, et al. Kola: Carefully benchmarking world knowledge of
786 large language models. *arXiv preprint arXiv:2306.09296*, 2023.
787
- 788 Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali,
789 Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance
790 prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.
- 791 Sukmin Yun, Jaehyung Kim, Dongyoon Han, Hwanjun Song, Jung-Woo Ha, and Jinwoo Shin. Time
792 is matter: Temporal self-supervision for video transformers. *arXiv preprint arXiv:2207.09067*,
793 2022.
794
- 795 Yuhang Zang, Wei Li, Jun Han, Kaiyang Zhou, and Chen Change Loy. Contextual object detection
796 with multimodal large language models. *arXiv preprint arXiv:2305.18279*, 2023.
- 797 Andy Zeng, Adrian Wong, Stefan Welker, Krzysztof Choromanski, Federico Tombari, Aavek Puro-
798 hit, Michael S. Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, et al. Socratic models:
799 Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*,
800 2022.
801
- 802 Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language
803 image pre-training. In *Proceedings of the International Conference on Computer Vision (ICCV)*,
804 pp. 11975–11986, 2023.
- 805 Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Kai Chen, and
806 Ping Luo. Gpt4roi: Instruction tuning large language model on region-of-interest. *arXiv preprint
807 arXiv:2307.03601*, 2023.
808
- 809 Yang Zhao, Zhijie Lin, Daquan Zhou, Zilong Huang, Jiashi Feng, and Bingyi Kang. Bubogpt:
Enabling visual grounding in multi-modal llms. *arXiv preprint arXiv:2307.08581*, 2023.

810 Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for
811 large-scale task planning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36,
812 2024.

813
814 Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and
815 Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint*
816 *arXiv:2403.09631*, 2024.

817 Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *Proceedings of*
818 *the International Conference on Machine Learning (ICML)*, 2022.

819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863

APPENDIX

The appendix covers all the implementation details, expanded experiment results in both simulation and real-world, further discussions, raw results, and all the prompt templates we used in the paper.

A IMPLEMENTATION DETAILS

In this section, we provide more implementation details regarding dataset preparation, model training, inference, and RT-2 style baselines.

A.1 DATASET PREPARATION

In this subsection, we give more details and examples of the datasets used in this paper.

A.1.1 BUILD *inBC* DATASET FROM EXPERT TRAJECTORIES

We formulate the dataset in a *single-image single-turn conversation* setting to emulate a policy where the user queries the Vision Language Model (VLM) with the current observation, and the VLM generates a response that can be directly translated into a concrete action, as introduced in Sec. 4. The image and text from the current observation can be directly processed by the VLM using a fixed template, while the conversion of numerical actions into text is facilitated through the use of normalized 2D image coordinates. Tab. 5 contains an example using a VIMA-Bench sample.

Given the current limitations of LLaVA (Liu et al., 2023a), to optimize performance, we propose two techniques:

- **Action history in query.** Each query to the VLM includes a history of actions previously executed in the episode. This approach enhances the VLM’s understanding of task context within the constraints of a single-turn conversation. (See the orange text in Tab. 5)
- **Plan ‘twice’ act once - multi-step planning.** In action generation, the dataset is prepared such that the VLM is designed to generate *all* successive actions in the response, literally performing multi-step action planning. However, only the first action is executed. For the next state, we query the VLM again with a new observation and a question and only take the first action generated by the VLM.

These designs proved beneficial in our ablation study, as detailed in Tab. 10.

The rest of this section details the process on VIMA-Bench (Jiang et al., 2023). VIMA-Bench introduces 17 simulated robot manipulation tasks using multimodal prompts that combine text and reference images. The full accompanying dataset includes 660k expert trajectories, covering 13 of the 17 tasks. All tasks occur in an environment where a robot arm, equipped with either a spatula or a suction cup, is positioned alongside a flat table. Multiple objects with diverse textures are randomly initialized on the table according to the specific task setting.

Each episode in the dataset features a multimodal task description that clarifies the episode’s goal, incorporating images referred to as ‘reference images’. Two third-person view cameras capture the scene, providing a top-down view and a front view looking down at the table. The dataset includes RGB, instance segmentation masks and meta information (*e.g.*, textures and shapes) of the objects in images for the reference images and the observations captured by these cameras. We first extract the bounding boxes of each object from the instance-level segmentation mask, as an object detection oracle. Additionally, the dataset contains the expert action sequence required to complete the episode. The action space consists of two poses: for the robot equipped with a spatula, these poses indicate the start and end points of a push; for the robot with a suction cup, they specify where the robot picks up and places an object.

Due to the constraints of LLaVA (Liu et al., 2023a) and our findings presented in Tab. 11, in each conversation, only the current image from the front view camera is retained as the visual input to the VLM. Other reference images are substituted with short captions that describe the texture (color) and shape of the referred object, which are details extracted from the meta information (See *inBC* in Tab. 5). However, in principle, there is nothing stopping us from extending the current framework

918 to a multi-image setting, taking advantage of a VLM that can handle multiple interleaving images in
919 a conversation.
920

921 A.1.2 BUILD \mathcal{D} -*inBC* FROM *inBC* 922

923 While *inBC* in Tab. 5 has shown its great power in many aspects when the reference image contains
924 a scene that has multiple objects instead of one, the *inBC* will fail to deliver any useful information
925 (e.g., Tab. 6). To address this issue, we utilize an object detector to parse an image of a scene into
926 a list of objects with its corresponding bounding box. The new dataset is named \mathcal{D} -*inBC* because a
927 reference image will be ‘described’ as a list of objects now. Tab. 6 shows an example where \mathcal{D} -*inBC*
928 delivers more critical information than *inBC*.

929 A.1.3 AUXILIARY DATASETS 930

931 The auxiliary datasets are created using the template outlined in Fig. 4. During dataset generation,
932 for each sample, one template is randomly selected from a pool of 15 templates. These templates
933 were initially rephrased from a single sentence using GPT-4 (OpenAI, 2023). The full list of the
934 pools are listed in Tab. 23, Tab. 24, Tab. 22, Tab. 25, Tab. 26, Tab. 27, Tab. 28, Tab. 29, Tab. 30,
935 Tab. 31, and Tab. 32. The qualitative examples are available in Tab. 7 and Tab. 8.
936

937 A.2 TRAINING 938

939 We initiate training using a pretrained LLaVA-1.5-7B (Liu et al., 2024b) model and finetune all
940 parameters, including the language model and the projection layer, with the exception of the vision
941 encoder. The training settings closely align with those of the original LLaVA stage 2. Specifically,
942 we utilize a single-cycle cosine annealing scheduling with 0.03 warm-up ratio and a maximum
943 learning rate of 2×10^{-5} . However, for VIMA-0.8k and VIMA-8k, we employ a batch size of 32,
944 whereas for VIMA-80k, we restore the batch size to 128.

945 A.3 INFERENCE 946

947 During inference, for the models trained on *inBC*, a new conversation is initiated at each step. The
948 model is queried with a single current image observation from the front-view camera and an in-
949 struction that has been prepared in the same format as the dataset. Additionally, a prompt randomly
950 selected from Tab. 21 is added to the instruction during inference only, although it appears to have
951 limited impact in retrospect (see Appendix B.7).

952 Model sampling is disabled during training, ensuring that the model consistently outputs the token
953 with the highest probability. This approach is designed to maximize the consistency and reliability
954 of the model’s responses.

955 For the models trained on \mathcal{D} -*inBC*, before we query the VLM, we would first run object detection on
956 all reference images and take the detection results to fill the instruction template of \mathcal{D} -*inBC*. Besides
957 this, the settings are identical to the models introduced ahead.

958 In this paper, we explore three approaches to object detection. The first method involves using a
959 single query from Tab. 23 and Tab. 24 and employing the same VLM as the policy to perform object
960 detection.
961

962 The second approach utilizes a Mask-RCNN (He et al., 2017) model, which features a ResNet50 (He
963 et al., 2016) backbone pretrained on the COCO dataset (Lin et al., 2014) and subsequently finetuned
964 using the VIMA dataset. A suffix ‘OD’ will be added to the model name if the model uses this
965 method.

966 Finally, we test the versions that use the groundtruth detection results. A suffix ‘Oracle’ will be
967 added to the model name if the model uses this information.
968

969 A.4 RT-2-STYLE BASELINES 970

971 For the RT-2-Style baseline, we follow the original RT-2’s design (Brohan et al., 2023a) and adapt
special tokens to present the numerical action in VIMA action space. The original action space

Table 5: Comparison between our converted instruction tuning datasets *inBC* and *D-inBC* for a VIMA-Bench sample. The task description of the episode is in blue. The description of an object in the reference image (oracle detection results) is in magenta. The action history is in orange.



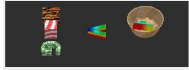
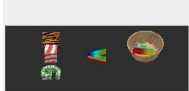
| | | |
|------|---------------|---|
| 972 | | |
| 973 | | |
| 974 | | |
| 975 | | |
| 976 | | |
| 977 | | |
| 978 | | |
| 979 | | |
| 980 | | |
| 981 | | |
| 982 | | {dragged_obj} |
| 983 | |  |
| 984 | | {base_obj} |
| 985 | |  |
| 986 | | |
| 987 | | |
| 988 | Task | First put {dragged_obj} into {base_obj} then put the object that was previously at its south into the same {base_obj} . |
| 989 | | |
| 990 | Input |  |
| 991 | | (current front camera view.) |
| 992 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 993 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 994 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 995 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 996 | <i>inBC</i> | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 997 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 998 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 999 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1000 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 1001 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 1002 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 1003 | | <task>First put <p>rainbow letter T</p> into <p>wooden bowl</p> then put the object that was previously at its south into the same <p>wooden bowl</p>.</task> |
| 1004 | Output | Step 2: Pick up the <p>rainbow letter V</p> at (0.500, 0.617) , rotate <r>[0]</r> degrees, and drop it at (0.746, 0.617) . |
| 1005 | | |
| 1006 | Input |  |
| 1007 | | (current front camera view.) |
| 1008 | | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1009 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1010 | | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1011 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1012 | <i>D-inBC</i> | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1013 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1014 | | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1015 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1016 | | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1017 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1018 | | <task>First put <p>rainbow letter T</p> at (0.500, 0.594), {0.102, 0.188} into <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328} then put the object that was previously at its south into the same <p>wooden bowl</p> at (0.457, 0.531), {0.195, 0.328}.</task> |
| 1019 | | Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r> . The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359. |
| 1020 | Output | [Same as <i>inBC</i>] |
| 1021 | | |
| 1022 | | |
| 1023 | | |
| 1024 | | |
| 1025 | | |

Table 6: Another comparison between two converted instruction tuning datasets. In this example, the reference images in the task description depict scenes with multiple objects. The task description of the episode is in blue. The description of an object in the reference image (oracle detection results) is in magenta.

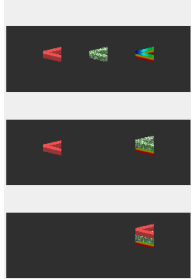


| | | |
|------|--------------------|--|
| 1026 | | |
| 1027 | | |
| 1028 | | |
| 1029 | | |
| 1030 | | |
| 1031 | | |
| 1032 | | |
| 1033 | | |
| 1034 | | |
| 1035 | | |
| 1036 | Original BC |  |
| 1037 | | |
| 1038 | | |
| 1039 | | |
| 1040 | | |
| 1041 | | Task Stack objects in this order {frame_0} {frame_1} {frame_2} |
| 1042 | | |
| 1043 | | |
| 1044 | | |
| 1045 | | |
| 1046 | | |
| 1047 | | |
| 1048 | | |
| 1049 | | |
| 1050 | | |
| 1051 | | |
| 1052 | | |
| 1053 | | |
| 1054 | | |
| 1055 | | |
| 1056 | | |
| 1057 | | |
| 1058 | | |
| 1059 | | |
| 1060 | | |
| 1061 | | |
| 1062 | | |
| 1063 | | |
| 1064 | | |
| 1065 | | |
| 1066 | | |
| 1067 | | |
| 1068 | | |
| 1069 | | |
| 1070 | | |
| 1071 | | |
| 1072 | | |
| 1073 | | |
| 1074 | | |
| 1075 | | |
| 1076 | | |
| 1077 | | |
| 1078 | | |
| 1079 | | |

Table 7: Qualitative examples of the auxiliary datasets. In this table, the examples of all the datasets take the same input image as shown in the first row. The description of an object or a scene (oracle detection results) is in magenta. The action is in orange.

| | |
|----------------------------|---|
| Image Input |  <p>(current front camera view.)</p> |
| Object Localization | <p>Input Could you tell me the location of <code><p>red letter V</p></code> in the image? Use the format <code>(x, y), {w, h}</code>, where <code>x</code> and <code>y</code> denote the center of the bounding box and <code>w</code> and <code>h</code> are the width and height. Coordinates start at the top left corner and should be normalized between 0 and 1.</p> <p>Output <code><p>red letter V</p> at (0.254, 0.570), {0.094, 0.156}</code>.</p> |
| Object Detection | <p>Input Enumerate and describe every object found in the image. For each object, utilize the format <code>(x, y), {w, h}</code>, where <code>x</code>, <code>y</code> are the bounding box center coordinates and <code>w</code>, <code>h</code> are the dimensions (width and height) of the bounding box. The coordinates begin at the top left corner and are normalized between 0 and 1.</p> <p>Output <code><scene></code> <code><p>red letter V</p> at (0.254, 0.570), {0.094, 0.156}</code> <code><p>green paisley letter V</p> at (0.500, 0.570), {0.094, 0.156}</code> <code><p>rainbow letter V</p> at (0.746, 0.570), {0.094, 0.156}</code> <code><p>green paisley block</p> at (0.668, 0.414), {0.094, 0.227}</code> <code></scene></code></p> |
| Action Prediction | <p>Input Can you explain what needs to be done to adjust the scene shown in the image to resemble the second scene? The second scene <code><scene><p>red letter V</p> at (0.254, 0.594), 0.094, 0.156</code>. <code><p>green paisley letter V</p> at (0.742, 0.547), 0.098, 0.156</code>. <code><p>rainbow letter V</p> at (0.746, 0.609), 0.094, 0.125</code>. <code><p>green paisley block</p> at (0.668, 0.414), 0.094, 0.227</code>.<code></scene></code> consists of object bounding boxes provided in the format <code>(x, y), {w, h}</code>. Here, <code>x</code> and <code>y</code> represent the center coordinates, and <code>w</code> and <code>h</code> are the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1. Every action you take must include two locations in the format of <code>(x, y)</code> and one clockwise rotation angle in the format of <code><r>[r]</r></code>. The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359.</p> <p>Output <code>Pick up the <p>green paisley letter V</p> at (0.465, 0.617)</code>, rotate <code><r>[0]</r></code> degrees, and drop it at <code>(0.711, 0.617)</code>.</p> |
| Future Prediction | <p>Input An image depicts a scene containing multiple objects. Now you <code>pick up the <p>green paisley letter V</p> at (0.465, 0.617)</code>, rotate <code><r>[0]</r></code> degrees, and drop it at <code>(0.711, 0.617)</code>, what will the scene look like? Write the list of object bounding boxes. The bounding boxes should be formatted as <code>(x, y), {w, h}</code>, where <code>x</code> and <code>y</code> denote the center coordinates, and <code>w</code> and <code>h</code> are the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.</p> <p>Output <code><scene></code> <code><p>red letter V</p> at (0.254, 0.594), {0.094, 0.156}</code> <code><p>green paisley letter V</p> at (0.742, 0.547), {0.098, 0.156}</code> <code><p>rainbow letter V</p> at (0.746, 0.609), {0.094, 0.125}</code> <code><p>green paisley block</p> at (0.668, 0.414), {0.094, 0.227}</code> <code></scene></code></p> |

1134 Table 8: (Continued) Qualitative examples of the auxiliary datasets. In this table, the examples of all the
 1135 datasets take the same input image as shown in the first row. The description of an object or a scene (oracle
 1136 detection results) is in magenta. The exemplar is in teal.

| | | |
|------|------------------------------|---|
| 1137 | | |
| 1138 | | |
| 1139 | Image Input |  |
| 1140 | | (current front camera view.) |
| 1141 | | |
| 1142 | | Input Can you explain the relative spatial positions of <code><p>red letter V</p></code> com- 1143 pared to <code><p>green paisley letter V</p></code> in this image? Use terms such as 1144 left, right, above, below, etc. Also, determine the 2D center distance and 1145 the Euclidean center distance between them. Your output should match 1146 this format: <code><p>green paisley block</p></code> is left and top from <code><p>rainbow 1147 letter V</p></code> with 2d center distance (x,y) of <code><d>(-0.078, -0.156)</d></code> and 1148 euclidean center distance of <code><e>0.175</e></code> . The coordinates are image co- 1149 ordinates normalized to a scale of 0 to 1 starting from the top left corner. |
| 1149 | Spatial Relationship | |
| 1150 | | Output <code><p>red letter V</p></code> is left and bottom from <code><p>green paisley letter V</p></code> 1151 with 2d center distance (x,y) of <code><d>(-0.246, 0.000)</d></code> and euclidean center 1152 distance of <code><e>0.246</e></code> . |
| 1152 | | Input At the initial timestamp, the image shows a scene, and the second image 1153 described as <code><scene><p>red letter V</p></code> at <code>(0.254, 0.594)</code> , <code>{0.094,</code> 1154 <code>0.156}</code> . <code><p>green paisley letter V</p></code> at <code>(0.742, 0.547)</code> , <code>{0.098,</code> 1155 <code>0.125}</code> . <code><p>rainbow letter V</p></code> at <code>(0.746, 0.609)</code> , <code>{0.094,</code> 1156 <code>0.227}</code> . <code></scene></code> depicts the next timestamp. Can you describe the 1157 change in the relative position of <code><p>red letter V</p></code> compared to 1158 <code><p>green paisley letter V</p></code> between these two timestamps? Use relative 1159 distance terms such as getting closer or moving further away, etc. Also, 1160 find the change in the 2D center distance and the Euclidean center distance 1161 between the two images. Your output must follow this format: <code><p>rainbow 1162 letter V</p></code> moves far away from <code><p>green paisley block</p></code> . 2d center 1163 distance (x,y) of <code><p>green paisley block</p></code> from <code><p>rainbow letter 1164 V</p></code> changes by <code><d>(0.000, -0.039)</d></code> and Euclidean center distance 1165 between them <code><e>0.036</e></code> . The coordinates are image coordinates nor- 1166 malized to a scale of 0 to 1 starting from the top left corner. |
| 1165 | Temporal Relationship | |
| 1166 | | Output <code><p>green paisley letter V</p></code> moves far away from <code><p>red letter V</p></code> . 1167 2d center distance (x,y) of <code><p>red letter V</p></code> from <code><p>green paisley letter 1168 V</p></code> changes by <code><d>(-0.242, 0.047)</d></code> and Euclidean center distance 1169 between them <code><e>0.244</e></code> . |

1170 of VIMA contains two points for pick and place. Each point contains a 2D coordinate in *robot*
 1171 *coordinates* for position and a 3D orientation of the gripper. Since we only have yaw rotation in the
 1172 dataset, we simplify two 3D orientations into one yaw angle. So the action space is now 5 Degrees
 1173 of Freedom (DoF): 4 numerical values present the 2D pick and place locations in *robot coordinates*,
 1174 and 1 value presents yaw rotation. In our implementation, the procedure begins by normalizing
 1175 each element of the action to a range from 0 to 1. We then quantize these values into 256 bins
 1176 and map the resulting discrete values to tokens indexed from 31000 to 31255 in the tokenizer. The
 1177 dataset employed is similar to *inBC*, but it omits the prompt for the output format, and the action is
 1178 represented by the mapped special tokens. Additionally, we have prepared a version similar to *D-*
 1179 *inBC*, named *D-RT-2 Style*, which takes advantage of object detection on reference images. *D-RT-2*
 1180 *Style (I)* is similar to *D-RT-2 Style*. However, four special tokens present pick and place positions in
 1181 two normalized 2D *image coordinates* for each action, and one token presents the rotation, instead
 1182 of an action in VIMABench action space. It is the same action presentation generated by our LLaRA
 1183 but our method responds in natural language while *D-RT-2 Style (I)* responds in special tokens.

B EXPANDED SIMULATION EXPERIMENTS

In this section, we include more experiment results. All the numerical results reported in this paper, especially in the figures, are collected in Tab. 14 and Tab. 15 (VIMA-0.8k), Tab. 16 and Tab. 17 (VIMA-8k), Tab. 18 and Tab. 19 (VIMA-80k), and Tab. 20 (full VIMA dataset).

B.1 CLARIFICATION ON L4 RESULTS

We adopted the testing protocol from VIMA-Bench (Jiang et al., 2023), conducting evaluations across four levels of difficulty, from L1 to L4. However, during our analysis, we discovered an inconsistency in the training set: the rotation information for the robot end effector was recorded as zero for all sweeping tasks where the end effector is a spatula. Given the importance of the spatula orientation in a sweeping task and the fact that sweeping tasks constitute 25% of the evaluations at the L4 difficulty level, we concluded that our ability to accurately evaluate our method at L4 was compromised. Considering that the original VIMA model released by the authors appears to include this rotation information, we have chosen not to report the results for L4 in our study.

B.2 EXTENDED COMPARISON TO VIMA AND OTHER BASELINES

We would first clarify the difference between our method and VIMA (Jiang et al., 2023) in terms of the inputs. VIMA takes both front and top view images from the environment while ours only takes the front view. We test the most capable model released by VIMA, which is trained on 660k expert trajectories, as well as other RT-2-Style baselines trained on the same whole 660k dataset. The results are listed in Tab. 9. Compared to VIMA (Jiang et al., 2023) and other baselines, our best model not only achieves better performance but also requires less input and is trained on only 12% of the data used in VIMA.

Table 9: Comparison to VIMA (Jiang et al., 2023) and other baselines. Our best model not only achieves better performance but also requires less input and is trained on only 12% of the data used in VIMA.

| Method | Config | Data | L1 (%) | L2 (%) | L3 (%) |
|---------------------------|----------------------------------|------|-------------|-------------|-------------|
| VIMA (Jiang et al., 2023) | VIMA-200M + Oracle | 100% | 80.7 | 81.9 | 77.9 |
| <i>RT-2 Style</i> | | 12% | 56.9 | 53.1 | 46.2 |
| | | 100% | 65.0 | 59.6 | 42.5 |
| <i>D-RT-2 Style</i> | <i>D-RT-2 Style</i> + Oracle | 12% | 74.2 | 70.4 | 69.2 |
| | <i>D-RT-2 Style</i> + Oracle | 100% | 86.2 | 88.1 | 75.8 |
| <i>D-RT-2 Style (I)</i> | <i>D-RT-2 Style (I)</i> + Oracle | 12% | 58.1 | 54.2 | 55.4 |
| | <i>D-RT-2 Style (I)</i> + Oracle | 100% | 70.0 | 70.0 | 59.2 |
| LLaRA (Ours) | <i>D-inBC</i> + Oracle | 12% | 87.3 | 85.4 | 82.1 |
| | <i>D-inBC</i> + Aux (B) + Oracle | 12% | 90.0 | 88.1 | 79.2 |
| | <i>D-inBC</i> + Aux (D) + Oracle | 12% | 83.8 | 88.1 | 78.8 |

B.3 WILL BASELINES BENEFIT FROM LONGER TRAINING?

The introduction of auxiliary datasets in the training set naturally increases the number of optimization steps per training epoch. In this subsection, we confirm that the performance improvement of LLaRA with auxiliary datasets is due to the data itself, rather than simply more optimization steps. To validate this, we trained several baselines without auxiliary data for longer epochs on three VIMA subsets. As shown in Fig. 9, Fig. 10, and Fig. 11, extending the training epochs does not enhance performance, indicating that the auxiliary data provides additional information that boosts model performance.

B.4 ABLATION ON ACTION HISTORY AND MULTI-STEP PLANNING

As described in Appendix A.1.1, we enable action history and multi-step planning when generating *inBC* and *D-inBC*. Tab. 10 shows that these designs are helpful.

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295

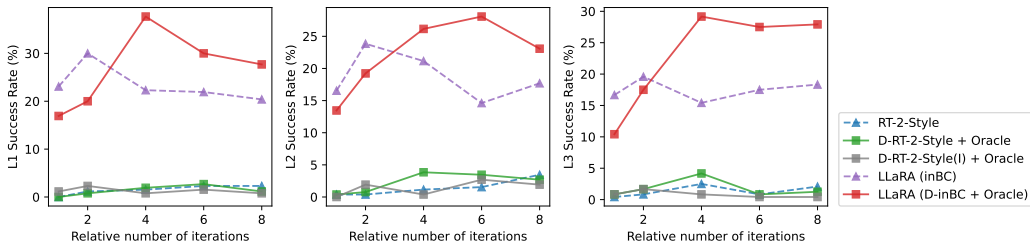


Figure 9: Model performances on VIMA-0.8k for longer epochs.

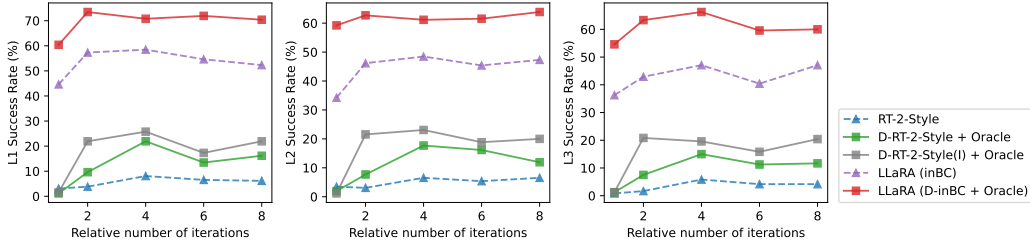


Figure 10: Model performances on VIMA-8k for longer epochs.

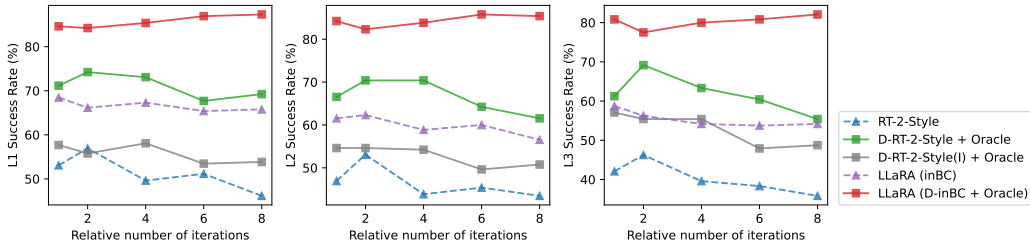


Figure 11: Model performances on VIMA-80k for longer epochs.

Table 10: Ablation on action history and multi-step planning. **His.** stands for enabling action history and **Plan** means enabling multi-step planning. All models are trained on VIMA-8k for 2 epochs.

| Method | His. | Plan | L1 (%) | L2 (%) | L3 (%) |
|--------------------|------|------|-------------|-------------|-------------|
| <i>inBC</i> | ✓ | ✓ | 57.3 | 46.2 | 42.9 |
| <i>inBC</i> | ✓ | ✗ | 43.5 | 36.9 | 36.7 |
| <i>inBC</i> | ✗ | ✓ | 45.8 | 39.6 | 35.8 |
| <i>D-inBC</i> + OD | ✓ | ✓ | 63.5 | 51.5 | 50.0 |
| <i>D-inBC</i> + OD | ✓ | ✗ | 58.5 | 41.5 | 47.1 |
| <i>D-inBC</i> + OD | ✗ | ✓ | 53.5 | 39.2 | 37.5 |

B.5 ABLATION ON MULTIPLE IMAGE INPUTS

This ablation studies multiple image inputs within a single conversation. Each image is processed by the vision encoder and the projection layer to generate a series of tokens. These image tokens are then integrated into the textual conversation at the points where the corresponding images are referenced.

However, because LLaVA is trained with one image per conversation instead of an interleaving style. The performance drops significantly when applying multiple image inputs, listed in Tab. 11.

Table 11: Ablation on multiple image inputs (**Mul.**). All models are trained on VIMA-8k for 2 epochs.

| Method | Mul. | L1 (%) | L2 (%) | L3 (%) |
|-------------------------|------|-------------|-------------|-------------|
| <i>inBC</i> | ✗ | 57.3 | 46.2 | 42.9 |
| <i>inBC</i> | ✓ | 44.6 | 27.7 | 34.2 |
| <i>D-inBC</i> + OD | ✗ | 63.5 | 51.5 | 50.0 |
| <i>D-inBC</i> + OD | ✓ | 27.7 | 24.6 | 34.2 |
| <i>D-inBC</i> + Aux (C) | ✗ | 64.6 | 58.8 | 49.6 |
| <i>D-inBC</i> + Aux (C) | ✓ | 31.2 | 28.5 | 27.1 |

B.6 ABLATION ON OBJECT DETECTOR

This ablation studies three types of object detectors in this paper: the VLM model itself, an external object detector separately trained on the training set (the methods with a suffix *OD*), and an oracle from the groundtruth dataset (Oracle). Fig. 12 shows that a reliable object detector is highly beneficial, enhancing the accuracy of image-based inputs and consequently improving model performance.

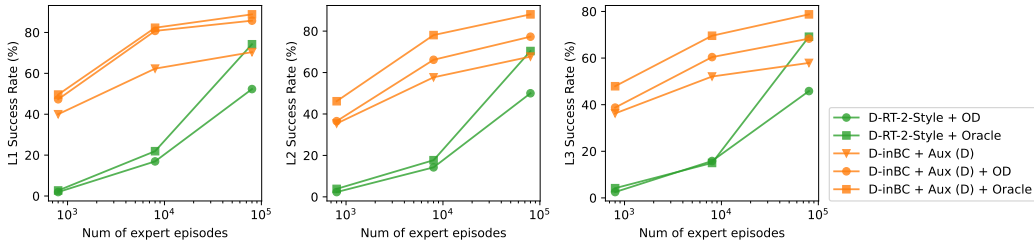


Figure 12: Impact of different object detectors.

B.7 ABLATION ON ACTION PROMPT DURING INFERENCE

This ablation studies the effect of the prompt sentence used in action generation. By default, during test time we formulate an instruction similar to Tab. 5 to query the VLM. In addition, we randomly sample one sentence from Tab. 21, named action prompt, and add it to the instruction. We study the following scenarios to understand the effect of this action prompt selection:

- *random*: this is the default behavior, a random action prompt will be added to each instruction;
- *no_prompt*: none instruction has the action prompt;
- *prompt* p_a , where $p_a \in [0, 14]$: all the instructions in this setting use the p_a^{th} prompt (0-index) in Tab. 21.

The results are presented in Fig. 13, where two variants of *D-inBC* are trained on VIMA-80k. In general, the selection of the action prompt does not result in a significant difference in performance except that *D-inBC* + Aux (D) seems to perform better on L1 and L3 without the action prompt.

B.8 FEW-SHOT EXPERIMENTS USING LLM / VLM

We expanded our evaluation to include the few-shot performance of existing Large Language Models (LLMs) and Vision Language Models (VLMs). Consistent with the inference settings described in Appendix A.3, we provided three additional examples from either *inBC* or *D-inBC* for each test scenario. The results of these evaluations are detailed in Tab. 12, where we examine three variables:

Vision: When this option is enabled, the model will have access to the current image observation, allowing it to integrate visual data into its response. If disabled, the model operates solely as a text-

1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

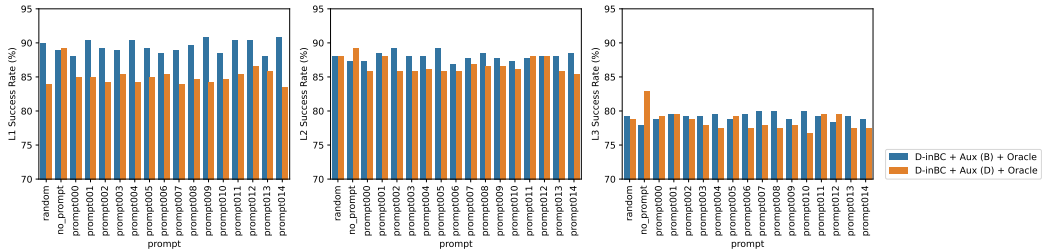


Figure 13: Effect of prompt sentence for action generation.

based system.

Oracle: This variable is checked when the examples are drawn from the \mathcal{D} -inBC dataset. It indicates that the examples provided for few-shot learning include descriptive textual information about the reference images, potentially offering richer context.

Same-task: When this is checked, the examples used for prompting are from the same task type as those in the evaluation environment.

From the observed low performance, we infer that the data utilized for robot control significantly deviates from the conversation contexts in which these models were originally trained. This indicates a pressing need to further finetune the models specifically for robot control tasks to achieve adequate performance. Additionally, our findings hint that visual input considerably enhances the functionality of GPT-4o.

Table 12: Three-shot performance of LLMs / VLMs

| Output | Vision | Oracle | Same-task | L1 (%) | L2 (%) | L3 (%) | L4 (%) |
|---------------------|--------|--------|-----------|--------|--------|--------|--------|
| GPT-4o | ✗ | ✗ | ✓ | 1.4 | 1.4 | 0.0 | N/A |
| GPT-4o | ✗ | ✗ | ✗ | 0.4 | 0.4 | 0.4 | 0.0 |
| GPT-4o | ✗ | ✓ | ✓ | 1.4 | 1.8 | 0.5 | N/A |
| GPT-4o | ✗ | ✓ | ✗ | 1.2 | 0.4 | 0.4 | 1.2 |
| GPT-4o | ✓ | ✗ | ✓ | 5.9 | 4.5 | 4.0 | N/A |
| GPT-4o | ✓ | ✗ | ✗ | 6.9 | 5.8 | 4.2 | 1.2 |
| Llama 3 70B (4-bit) | ✗ | ✗ | ✓ | 0.0 | 0.0 | 0.0 | N/A |
| Llama 3 70B (4-bit) | ✗ | ✗ | ✗ | 0.8 | 1.2 | 0.4 | 1.2 |
| Llama 3 70B (4-bit) | ✗ | ✓ | ✓ | 0.0 | 1.4 | 0.5 | N/A |
| Llama 3 70B (4-bit) | ✗ | ✓ | ✗ | 0.0 | 0.4 | 0.4 | 0.0 |
| LlaVA-1.5-7B | ✓ | ✗ | ✓ | 0.9 | 1.4 | 0.5 | N/A |
| LlaVA-1.5-7B | ✓ | ✗ | ✗ | 0.0 | 0.0 | 1.0 | 1.2 |
| LlaVA-1.6-34B | ✓ | ✗ | ✓ | 1.4 | 1.4 | 0.5 | N/A |
| LlaVA-1.6-34B | ✓ | ✗ | ✗ | 1.2 | 0.8 | 2.1 | 0.0 |

C DETAILS ON REAL-WORLD ROBOT EXPERIMENTS

In this section, we provide more details on real-world robot experiments first introduced in Sec. 6.2.

C.1 ENVIRONMENT SETTING

We utilize an xArm7 robot arm equipped with a gripper and a Logitech C140 RGB webcam positioned above the arm to gather observations. The camera is mounted in a fixed position above and slightly in front of the arm, providing a third-person view of the scene. The captured images have a raw resolution of 640×480 , which are then center-cropped to 640×320 to match the aspect ratio of

VIMA images. The robot’s action space includes 4 DoF for pick-and-place positioning and 1 DoF for rotation, consistent with the simplified VIMA setup.

The object detector for all real-world experiments is an OWLv2 (Minderer et al., 2024) (ViT-base, patch size 16) running in one-shot detection mode. The object detector takes a prompt image from the same domain and detects the objects in the observation. The list of plastic toys we used as {object} (and detected) is: {duck, croissant, cupcake, blueberries, carrot, banana, grapes, donut, tomato, corn, pepper}

C.2 GPT-4O BASELINE

In the GPT-4o baseline, the model is fed the same prompt as in RT-2, with an additional prompt describing the robot arm: “You are operating a robot arm at a table to complete a task specified between <task></task>. Given the current image, you need to generate a series of actions to control the robot arm to accomplish the task.”. The model’s answer is parsed to extract the pick and place points.

C.3 REAL-WORLD ROBOT DATASET

We further collect 1,256 real-world images from the same real-world robot setting as the in-domain data. Out of these images, 721 feature a single object placed randomly and uniformly on the table by a robotic arm, while the remaining images display multiple objects scattered across the table. We employed one-shot detection using OWLv2 (Minderer et al., 2024) to identify and generate bounding boxes for each object in the images.

These images, along with their bounding box data, have been utilized to create two specialized in-domain instruction tuning datasets, each containing 2,447 single-turn conversations of equivalent size: xArm-Det and xArm-Action. The xArm-Det dataset includes conversations that mirror the structure from our auxiliary object localization and detection dataset, with 1,191 examples dedicated to object localization and 1,256 to object detection. In contrast, the xArm-Action dataset is designed similarly to data for *D-inBC*, featuring two distinct tasks: *Rotation*: rotating an object (akin to Task T2 with 1,191 examples) and *Put on top*: putting one object on top of another (reminiscent of Task T3 with 1,256 examples).

In the *Rotation* task, the agent is instructed to rotate a specific object by a random angle ranging from -180 to 180 degrees, while ensuring the object’s position remains unchanged post-rotation. The expert action is straightforwardly generated by directly replicating the specified rotation angle from the task description and the pick and place points are set to the centroid of the object bounding box from object detection.

In the *Put on Top* task, applicable to images containing more than two objects, we randomly select two objects. The task involves generating a command that directs placing one object atop the other. The expert actions, specifically the pick and place points, are determined using the centroids of the bounding boxes of the two selected objects from object detection.

Tab. 13 shows examples of the real-world datasets we collected.


C.4 QUALITATIVE EXAMPLES ON UNSEEN REAL-WORLD TASKS

In Fig. 14 we qualitatively show that LLaRA can complete some unseen tasks that are not in the training dataset. In the first example, we ask LLaRA to ‘place all other objects into the large bowl’. LLaRA is able to recognize the objects to move and not to move and outputs the correct actions. In the second example, we let LLaRA ‘get the weight of the tomato and put it back’. LLaRA can put the tomato on the scale and move the tomato back to the similar place as the initial position, showing the understanding of ‘putting it back’. These examples show the potential generalization ability of LLaRA. Video demonstrations are also included in the supplementary material.

Table 13: Qualitative examples of our real-world datasets xArm-Det and xArm-Action. The description of an object in the image (one-shot detection results) is in **magenta**. The task description of the episode is in **blue**.

| | | |
|------|--|--|
| 1458 | | |
| 1459 | | |
| 1460 | | |
| 1461 | | |
| 1462 | | |
| 1463 | | |
| 1464 | | |
| 1465 | | |
| 1466 | | |
| 1467 | | |
| 1468 | | |
| 1469 | | |
| 1470 | | |
| 1471 | | |
| 1472 | | |
| 1473 | | |
| 1474 | | |
| 1475 | | |
| 1476 | | |
| 1477 | | |
| 1478 | | |
| 1479 | | |
| 1480 | | |
| 1481 | | |
| 1482 | | |
| 1483 | | |
| 1484 | | |
| 1485 | | |
| 1486 | | |
| 1487 | | |
| 1488 | | |
| 1489 | | |
| 1490 | | |
| 1491 | | |
| 1492 | | |
| 1493 | | |
| 1494 | | |
| 1495 | | |
| 1496 | | |
| 1497 | | |
| 1498 | | |
| 1499 | | |
| 1500 | | |
| 1501 | | |
| 1502 | | |
| 1503 | | |
| 1504 | | |
| 1505 | | |
| 1506 | | |
| 1507 | | |
| 1508 | | |
| 1509 | | |
| 1510 | | |
| 1511 | | |

Image Input



| | | |
|--------------------|---------------------|--|
| xArm-Det | Localization | <p>Input Could you tell me the location of <p>blueberry</p> in the image? Use the format (x, y), {w, h}, where x and y denote the center of the bounding box and w and h are the width and height. Coordinates start at the top left corner and should be normalized between 0 and 1.</p> <p>Output <p>blueberry</p> at (0.478, 0.725), {0.058, 0.125}.</p> |
| xArm-Det | Detection | <p>Input Describe all the objects seen in the image, and list them using the format (x, y), {w, h}. The x and y values are the coordinates for the center of the bounding box, while w and h represent its width and height. The coordinates should be normalized from the top left corner, within a range of 0 to 1.</p> <p>Output <scene> <p>cupcake</p> at (0.440, 0.128), {0.063, 0.123}. <p>corn</p> at (0.237, 0.826), {0.061, 0.246}. <p>carrot</p> at (0.608, 0.135), {0.152, 0.113}. <p>donut</p> at (0.396, 0.355), {0.079, 0.145}. <p>grape</p> at (0.318, 0.534), {0.077, 0.143}. <p>duck</p> at (0.795, 0.280), {0.066, 0.128}. <p>green_pepper</p> at (0.688, 0.411), {0.062, 0.187}. <p>blueberry</p> at (0.478, 0.725), {0.058, 0.125}. <p>tomato</p> at (0.543, 0.421), {0.078, 0.154}. </scene></p> |
| xArm-Action | Rotation | <p>Input <task>Rotate the <p>cupcake</p> at (0.440, 0.128), {0.063, 0.123} by -50 degrees.</task> Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r>. The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359.</p> <p>Output Pick up the <p>cupcake</p> at (0.440, 0.128), rotate <r>[-50]</r> degrees, and drop it at (0.440, 0.128).</p> |
| xArm-Action | Put on top | <p>Input <task>Move the <p>cupcake</p> at (0.440, 0.128), {0.063, 0.123} on the top of the <p>corn</p> at (0.237, 0.826), {0.061, 0.246}.</task> Every action you take must include two locations in the format of (x, y) and one clockwise rotation angle in the format of <r>[r]</r>. The first location is the image coordinate where you use a suction cup to pick up the object, and the second location is where you place the object. The image coordinate ranges from 0 to 1. The rotation angle indicates how many degrees you rotate the object clockwise, and it ranges from -359 to 359.</p> <p>Output Step 1: Pick up the <p>cupcake</p> at (0.440, 0.128), rotate <r>[0]</r> degrees, and drop it at (0.237, 0.826).</p> |



Figure 14: Examples of using LLaRA for unseen tasks. We show the input texts from the user and input images at each step of the task, and images about the robot’s actions to accomplish each step.

D LIMITATIONS

While our model has demonstrated significant achievements and holds considerable potential, it is important to acknowledge that there remains space for further optimization.

First, some concepts in the reference images still can not be easily and precisely described by language, while in many cases these features are critical for robot control. Due to the single image limit of the current VLM, we still rely on the object detector trained on a limited number of classes to describe the additional images in the task description, which limits the generalization ability of this method. In principle, in the future, this can be addressed by a VLM trained on interleave image text data like (Bai et al., 2023; Li et al., 2024a).

Second, the information extracted from the dataset can be noisy, which may lead to model confusion. In Tab. 5, there are discrepancies such as the location of an object in a reference image differing from its actual location in the current image observation. At the same time, in VIMA dataset, when referring to the shape of an object without a color, the object in the reference image will appear in gray, which is a totally different color than the object with the same shape in the current observations.

Finally, our method relies on 2D to 2D mapping to convert image coordinates into actual actions, which may face challenges when the complex 3D movement of the robot is required.

We believe enhancements in the mentioned aspects could further improve performance and broader applicability in complex, real-world applications.

E RAW RESULTS AND PROMPT TEMPLATES

This section contains all the numerical results of VIMA experiments and all prompt templates we used in the paper.

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619

Table 14: Results on VIMA-0.8k dataset. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|---------------------|----|-----|------|------|------|------|------|-------|--------|--------|--------|
| <i>RT-2 Style</i> | 1 | 1 | | | | | | | 0.0 | 0.4 | 0.4 |
| <i>RT-2 Style</i> | 2 | 2 | | | | | | | 1.2 | 0.4 | 0.8 |
| <i>RT-2 Style</i> | 4 | 4 | | | | | | | 1.5 | 1.2 | 2.5 |
| <i>RT-2 Style</i> | 6 | 6 | | | | | | | 2.3 | 1.5 | 0.8 |
| <i>RT-2 Style</i> | 8 | 8 | | | | | | | 2.3 | 3.5 | 2.1 |
| <i>inBC</i> | 1 | 1 | | | | | | | 23.1 | 16.5 | 16.7 |
| <i>inBC</i> | 2 | 2 | | | | | | | 30.0 | 23.8 | 19.6 |
| <i>inBC</i> | 4 | 4 | | | | | | | 22.3 | 21.2 | 15.4 |
| <i>inBC</i> | 6 | 6 | | | | | | | 21.9 | 14.6 | 17.5 |
| <i>inBC</i> | 8 | 8 | | | | | | | 20.4 | 17.7 | 18.3 |
| <i>inBC</i> | 10 | 10 | | | | | | | 20.4 | 18.1 | 14.6 |
| <i>inBC + Aux</i> | 1 | 1.2 | 0.1* | 0.1* | | | | | 13.8 | 13.1 | 9.2 |
| <i>inBC + Aux</i> | 1 | 3 | 1* | 1* | | | | | 28.8 | 26.5 | 24.2 |
| <i>inBC + Aux</i> | 2 | 2.4 | 0.1* | 0.1* | | | | | 29.6 | 21.9 | 23.3 |
| <i>inBC + Aux</i> | 2 | 2.4 | 0.1 | 0.1 | | | | | 23.5 | 20.4 | 17.5 |
| <i>inBC + Aux</i> | 2 | 4 | 0.5 | 0.5 | | | | | 26.9 | 23.8 | 24.6 |
| <i>inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | | | 35.0 | 24.2 | 26.7 |
| <i>inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | | 33.5 | 26.2 | 21.7 |
| <i>inBC + Aux</i> | 2 | 6 | 1* | 1* | | | | | 31.9 | 26.5 | 30.4 |
| <i>inBC + Aux</i> | 2 | 6 | 1 | 1 | | | | | 30.4 | 27.3 | 27.1 |
| <i>inBC + Aux</i> | 2 | 7 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | | 36.2 | 29.6 | 29.2 |
| <i>inBC + Aux</i> | 2 | 8 | 0.5* | 0.5* | 0.5 | 0.5 | 0.5* | 0.5 | 33.5 | 29.2 | 33.3 |
| <i>inBC + Aux</i> | 2 | 8 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 31.9 | 29.6 | 26.7 |
| <i>inBC + Aux</i> | 2 | 10 | 1 | 1 | 1 | 1 | | | 37.7 | 30.8 | 28.7 |
| <i>inBC + Aux</i> | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 38.8 | 34.6 | 29.6 |
| <i>inBC + Aux</i> | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 37.7 | 29.6 | 27.9 |
| <i>inBC + Aux</i> | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 40.0 | 35.8 | 31.2 |
| <i>D-inBC + Aux</i> | 1 | 1.2 | 0.1* | 0.1* | | | | | 18.5 | 17.7 | 13.8 |
| <i>D-inBC + Aux</i> | 1 | 3 | 1* | 1* | | | | | 28.8 | 25.4 | 23.3 |
| <i>D-inBC + Aux</i> | 2 | 2.4 | 0.1* | 0.1* | | | | | 23.5 | 16.2 | 22.5 |
| <i>D-inBC + Aux</i> | 2 | 2.4 | 0.1 | 0.1 | | | | | 21.5 | 21.2 | 23.3 |
| <i>D-inBC + Aux</i> | 2 | 4 | 0.5 | 0.5 | | | | | 28.1 | 25.8 | 21.7 |
| <i>D-inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | | | 32.3 | 29.6 | 22.1 |
| <i>D-inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | | 30.0 | 26.5 | 25.4 |
| <i>D-inBC + Aux</i> | 2 | 6 | 1* | 1* | | | | | 33.8 | 30.8 | 29.2 |
| <i>D-inBC + Aux</i> | 2 | 6 | 1 | 1 | | | | | 35.8 | 28.5 | 28.3 |
| <i>D-inBC + Aux</i> | 2 | 7 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | | 38.5 | 26.9 | 30.8 |
| <i>D-inBC + Aux</i> | 2 | 8 | 0.5* | 0.5* | 0.5 | 0.5 | 0.5* | 0.5 | 37.3 | 26.5 | 34.6 |
| <i>D-inBC + Aux</i> | 2 | 8 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 34.2 | 29.6 | 30.0 |
| <i>D-inBC + Aux</i> | 2 | 10 | 1 | 1 | 1 | 1 | | | 34.6 | 33.1 | 28.3 |
| <i>D-inBC + Aux</i> | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 40.0 | 38.5 | 37.5 |
| <i>D-inBC + Aux</i> | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 40.8 | 30.4 | 32.9 |
| <i>D-inBC + Aux</i> | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 40.0 | 35.4 | 36.2 |

1620
 1621
 1622
 1623
 1624
 1625
 1626
 1627
 1628
 1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1638
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1658
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1668
 1669
 1670
 1671
 1672
 1673

Table 15: Results on VIMA-0.8k dataset with object detector or Oracle. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|----------------------------------|----|----|------|------|------|------|------|-------|--------|--------|--------|
| <i>D-RT-2 Style</i> + OD | 1 | 1 | | | | | | | 0.0 | 0.0 | 0.0 |
| <i>D-RT-2 Style</i> + OD | 2 | 2 | | | | | | | 0.4 | 1.5 | 2.5 |
| <i>D-RT-2 Style</i> + OD | 4 | 4 | | | | | | | 1.9 | 1.5 | 2.5 |
| <i>D-RT-2 Style</i> + OD | 6 | 6 | | | | | | | 1.9 | 2.3 | 0.8 |
| <i>D-RT-2 Style</i> + OD | 8 | 8 | | | | | | | 1.9 | 1.2 | 0.0 |
| <i>D-inBC</i> + OD | 1 | 1 | | | | | | | 17.3 | 11.9 | 12.5 |
| <i>D-inBC</i> + OD | 2 | 2 | | | | | | | 17.3 | 14.6 | 15.0 |
| <i>D-inBC</i> + OD | 4 | 4 | | | | | | | 32.7 | 24.2 | 24.2 |
| <i>D-inBC</i> + OD | 6 | 6 | | | | | | | 28.1 | 23.1 | 21.7 |
| <i>D-inBC</i> + OD | 8 | 8 | | | | | | | 26.2 | 17.7 | 22.9 |
| <i>D-inBC</i> + Aux + OD | 2 | 6 | 1 | 1 | | | | | 36.9 | 25.8 | 30.4 |
| <i>D-inBC</i> + Aux + OD | 2 | 10 | 1 | 1 | 1 | 1 | | | 41.2 | 31.9 | 32.9 |
| <i>D-inBC</i> + Aux + OD | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 44.6 | 33.5 | 37.9 |
| <i>D-inBC</i> + Aux + OD | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 47.3 | 36.5 | 38.8 |
| <i>D-RT-2 Style</i> + Oracle | 1 | 1 | | | | | | | 0.0 | 0.4 | 0.8 |
| <i>D-RT-2 Style</i> + Oracle | 2 | 2 | | | | | | | 0.8 | 0.8 | 1.7 |
| <i>D-RT-2 Style</i> + Oracle | 4 | 4 | | | | | | | 1.9 | 3.8 | 4.2 |
| <i>D-RT-2 Style</i> + Oracle | 6 | 6 | | | | | | | 2.7 | 3.5 | 0.8 |
| <i>D-RT-2 Style</i> + Oracle | 8 | 8 | | | | | | | 1.2 | 2.7 | 1.2 |
| <i>D-RT-2 Style (I)</i> + Oracle | 1 | 1 | | | | | | | 1.2 | 0.0 | 0.8 |
| <i>D-RT-2 Style (I)</i> + Oracle | 2 | 2 | | | | | | | 2.3 | 1.9 | 1.7 |
| <i>D-RT-2 Style (I)</i> + Oracle | 4 | 4 | | | | | | | 0.8 | 0.4 | 0.8 |
| <i>D-RT-2 Style (I)</i> + Oracle | 6 | 6 | | | | | | | 1.5 | 2.7 | 0.4 |
| <i>D-RT-2 Style (I)</i> + Oracle | 8 | 8 | | | | | | | 0.8 | 1.9 | 0.4 |
| <i>D-inBC</i> + Oracle | 1 | 1 | | | | | | | 16.9 | 13.5 | 10.4 |
| <i>D-inBC</i> + Oracle | 2 | 2 | | | | | | | 20.0 | 19.2 | 17.5 |
| <i>D-inBC</i> + Oracle | 4 | 4 | | | | | | | 37.7 | 26.2 | 29.2 |
| <i>D-inBC</i> + Oracle | 6 | 6 | | | | | | | 30.0 | 28.1 | 27.5 |
| <i>D-inBC</i> + Oracle | 8 | 8 | | | | | | | 27.7 | 23.1 | 27.9 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 6 | 1 | 1 | | | | | 45.0 | 36.9 | 39.2 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 10 | 1 | 1 | 1 | 1 | | | 43.8 | 38.5 | 37.9 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 52.7 | 44.6 | 50.4 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 49.6 | 46.2 | 47.9 |

1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727

Table 16: Results on VIMA-8k dataset. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|---------------------|----|-----|------|------|------|------|------|-------|--------|--------|--------|
| <i>RT-2 Style</i> | 1 | 1 | | | | | | | 3.1 | 3.5 | 0.8 |
| <i>RT-2 Style</i> | 2 | 2 | | | | | | | 3.8 | 3.1 | 1.7 |
| <i>RT-2 Style</i> | 4 | 4 | | | | | | | 8.1 | 6.5 | 5.8 |
| <i>RT-2 Style</i> | 6 | 6 | | | | | | | 6.5 | 5.4 | 4.2 |
| <i>RT-2 Style</i> | 8 | 8 | | | | | | | 6.2 | 6.5 | 4.2 |
| <i>inBC</i> | 1 | 1 | | | | | | | 44.6 | 34.2 | 36.2 |
| <i>inBC</i> | 2 | 2 | | | | | | | 57.3 | 46.2 | 42.9 |
| <i>inBC</i> | 4 | 4 | | | | | | | 58.5 | 48.5 | 47.1 |
| <i>inBC</i> | 6 | 6 | | | | | | | 54.6 | 45.4 | 40.4 |
| <i>inBC</i> | 8 | 8 | | | | | | | 52.3 | 47.3 | 47.1 |
| <i>inBC</i> | 10 | 10 | | | | | | | 53.1 | 47.7 | 47.9 |
| <i>inBC + Aux</i> | 2 | 2.4 | 0.1* | 0.1* | | | | | 59.6 | 50.0 | 45.4 |
| <i>inBC + Aux</i> | 2 | 2.4 | 0.1 | 0.1 | | | | | 57.3 | 52.3 | 45.4 |
| <i>inBC + Aux</i> | 2 | 4 | 0.5 | 0.5 | | | | | 60.4 | 56.2 | 52.1 |
| <i>inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | | | 59.6 | 59.2 | 51.2 |
| <i>inBC + Aux</i> | 2 | 6 | 1* | 1* | | | | | 59.2 | 52.3 | 48.3 |
| <i>inBC + Aux</i> | 2 | 6 | 1 | 1 | | | | | 57.7 | 56.2 | 53.8 |
| <i>inBC + Aux</i> | 2 | 7 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | | 58.1 | 55.0 | 47.9 |
| <i>inBC + Aux</i> | 2 | 10 | 1 | 1 | 1 | 1 | | | 60.4 | 55.8 | 54.2 |
| <i>inBC + Aux</i> | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 60.8 | 55.4 | 50.0 |
| <i>inBC + Aux</i> | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 63.1 | 58.8 | 53.8 |
| <i>inBC + Aux</i> | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 59.2 | 58.8 | 52.1 |
| <i>D-inBC + Aux</i> | 2 | 2.4 | 0.1* | 0.1* | | | | | 53.5 | 48.8 | 45.4 |
| <i>D-inBC + Aux</i> | 2 | 2.4 | 0.1 | 0.1 | | | | | 58.5 | 55.8 | 50.8 |
| <i>D-inBC + Aux</i> | 2 | 4 | 0.5 | 0.5 | | | | | 58.8 | 50.8 | 54.6 |
| <i>D-inBC + Aux</i> | 2 | 4 | 0.2 | 0.2 | 0.2 | 0.2 | | | 58.1 | 52.3 | 51.2 |
| <i>D-inBC + Aux</i> | 2 | 6 | 1* | 1* | | | | | 63.1 | 57.7 | 55.4 |
| <i>D-inBC + Aux</i> | 2 | 6 | 1 | 1 | | | | | 60.0 | 53.5 | 50.0 |
| <i>D-inBC + Aux</i> | 2 | 7 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | | 62.3 | 55.8 | 48.8 |
| <i>D-inBC + Aux</i> | 2 | 10 | 1 | 1 | 1 | 1 | | | 63.1 | 58.1 | 49.6 |
| <i>D-inBC + Aux</i> | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 64.6 | 58.8 | 49.6 |
| <i>D-inBC + Aux</i> | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 62.3 | 54.6 | 51.2 |
| <i>D-inBC + Aux</i> | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 62.3 | 57.7 | 52.1 |
| <i>D-inBC + Aux</i> | 4 | 4.8 | 0.1 | 0.1 | | | | | 58.5 | 47.7 | 49.6 |
| <i>D-inBC + Aux</i> | 4 | 12 | 1 | 1 | | | | | 61.2 | 57.7 | 49.6 |
| <i>D-inBC + Aux</i> | 4 | 20 | 1 | 1 | 1 | 1 | | | 62.3 | 56.5 | 49.6 |
| <i>D-inBC + Aux</i> | 4 | 24 | 1 | 1 | 1 | 1 | 1 | | 63.5 | 56.2 | 52.5 |

1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781

Table 17: Results on VIMA-8k dataset with object detector or Oracle. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|----------------------------------|----|-----|------|------|------|------|------|-------|--------|--------|--------|
| <i>D-RT-2 Style</i> + OD | 1 | 1 | | | | | | | 1.5 | 3.1 | 1.2 |
| <i>D-RT-2 Style</i> + OD | 2 | 2 | | | | | | | 10.4 | 8.8 | 7.1 |
| <i>D-RT-2 Style</i> + OD | 4 | 4 | | | | | | | 16.9 | 14.2 | 15.8 |
| <i>D-RT-2 Style</i> + OD | 6 | 6 | | | | | | | 14.2 | 11.5 | 12.9 |
| <i>D-RT-2 Style</i> + OD | 8 | 8 | | | | | | | 15.8 | 10.4 | 6.7 |
| <i>D-inBC</i> + OD | 1 | 1 | | | | | | | 55.4 | 45.8 | 45.0 |
| <i>D-inBC</i> + OD | 2 | 2 | | | | | | | 63.5 | 51.5 | 50.0 |
| <i>D-inBC</i> + OD | 4 | 4 | | | | | | | 63.8 | 46.9 | 55.8 |
| <i>D-inBC</i> + Aux + OD | 2 | 6 | 1 | 1 | | | | | 74.2 | 58.5 | 56.7 |
| <i>D-inBC</i> + Aux + OD | 2 | 10 | 1 | 1 | 1 | 1 | | | 75.4 | 66.2 | 57.9 |
| <i>D-inBC</i> + Aux + OD | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 76.9 | 63.8 | 57.9 |
| <i>D-inBC</i> + Aux + OD | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 79.2 | 64.2 | 58.3 |
| <i>D-inBC</i> + Aux + OD | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 72.7 | 55.4 | 57.1 |
| <i>D-inBC</i> + Aux + OD | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 79.2 | 65.4 | 60.4 |
| <i>D-inBC</i> + Aux + OD | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 80.8 | 66.2 | 60.0 |
| <i>D-inBC</i> + Aux + OD | 4 | 4.8 | 0.1 | 0.1 | | | | | 65.4 | 50.0 | 55.0 |
| <i>D-inBC</i> + Aux + OD | 4 | 12 | 1 | 1 | | | | | 76.9 | 63.1 | 55.8 |
| <i>D-RT-2 Style</i> + Oracle | 1 | 1 | | | | | | | 1.2 | 1.9 | 1.2 |
| <i>D-RT-2 Style</i> + Oracle | 2 | 2 | | | | | | | 9.6 | 7.7 | 7.5 |
| <i>D-RT-2 Style</i> + Oracle | 4 | 4 | | | | | | | 21.9 | 17.7 | 15.0 |
| <i>D-RT-2 Style</i> + Oracle | 6 | 6 | | | | | | | 13.5 | 16.2 | 11.2 |
| <i>D-RT-2 Style</i> + Oracle | 8 | 8 | | | | | | | 16.2 | 11.9 | 11.7 |
| <i>D-RT-2 Style (I)</i> + Oracle | 1 | 1 | | | | | | | 1.5 | 1.2 | 1.2 |
| <i>D-RT-2 Style (I)</i> + Oracle | 2 | 2 | | | | | | | 21.9 | 21.5 | 20.8 |
| <i>D-RT-2 Style (I)</i> + Oracle | 4 | 4 | | | | | | | 25.8 | 23.1 | 19.6 |
| <i>D-RT-2 Style (I)</i> + Oracle | 6 | 6 | | | | | | | 17.3 | 18.8 | 15.8 |
| <i>D-RT-2 Style (I)</i> + Oracle | 8 | 8 | | | | | | | 21.9 | 20.0 | 20.4 |
| <i>D-inBC</i> + Oracle | 1 | 1 | | | | | | | 60.4 | 59.2 | 54.6 |
| <i>D-inBC</i> + Oracle | 2 | 2 | | | | | | | 73.5 | 62.7 | 63.3 |
| <i>D-inBC</i> + Oracle | 4 | 4 | | | | | | | 70.8 | 61.2 | 66.2 |
| <i>D-inBC</i> + Oracle | 6 | 6 | | | | | | | 71.9 | 61.5 | 59.6 |
| <i>D-inBC</i> + Oracle | 8 | 8 | | | | | | | 70.4 | 63.8 | 60.0 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 6 | 1 | 1 | | | | | 78.1 | 73.8 | 65.4 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 10 | 1 | 1 | 1 | 1 | | | 78.1 | 73.8 | 70.0 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 78.5 | 78.1 | 71.2 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 14 | 1* | 1* | 1 | 1 | 1* | 1 | 76.2 | 65.8 | 66.2 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 82.3 | 78.1 | 69.6 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 4.8 | 0.1 | 0.1 | | | | | 71.9 | 64.6 | 61.7 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 12 | 1 | 1 | | | | | 78.8 | 75.4 | 67.1 |

1782
 1783
 1784
 1785
 1786
 1787
 1788
 1789
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1798
 1799
 1800
 1801
 1802
 1803
 1804
 1805
 1806
 1807
 1808
 1809
 1810
 1811
 1812
 1813
 1814
 1815
 1816
 1817
 1818
 1819
 1820
 1821
 1822
 1823
 1824
 1825
 1826
 1827
 1828
 1829
 1830
 1831
 1832
 1833
 1834
 1835

Table 18: Results on VIMA-80k dataset. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|---------------------|----|----|------|------|------|------|------|-------|--------|--------|--------|
| <i>RT-2 Style</i> | 1 | 1 | | | | | | | 53.1 | 46.9 | 42.1 |
| <i>RT-2 Style</i> | 2 | 2 | | | | | | | 56.9 | 53.1 | 46.2 |
| <i>RT-2 Style</i> | 4 | 4 | | | | | | | 49.6 | 43.8 | 39.6 |
| <i>RT-2 Style</i> | 6 | 6 | | | | | | | 51.2 | 45.4 | 38.3 |
| <i>RT-2 Style</i> | 8 | 8 | | | | | | | 46.2 | 43.5 | 35.8 |
| <i>inBC</i> | 1 | 1 | | | | | | | 68.5 | 61.5 | 58.8 |
| <i>inBC</i> | 2 | 2 | | | | | | | 66.2 | 62.3 | 56.2 |
| <i>inBC</i> | 4 | 4 | | | | | | | 67.3 | 58.8 | 54.2 |
| <i>inBC</i> | 6 | 6 | | | | | | | 65.4 | 60.0 | 53.8 |
| <i>inBC</i> | 8 | 8 | | | | | | | 65.8 | 56.5 | 54.2 |
| <i>inBC</i> + Aux | 1 | 3 | 1 | 1 | | | | | 62.7 | 60.4 | 50.8 |
| <i>inBC</i> + Aux | 1 | 5 | 1 | 1 | 1 | 1 | | | 66.9 | 63.8 | 56.2 |
| <i>inBC</i> + Aux | 1 | 6 | 1 | 1 | 1 | 1 | 1 | | 66.2 | 59.6 | 55.8 |
| <i>inBC</i> + Aux | 1 | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 68.1 | 60.8 | 54.6 |
| <i>inBC</i> + Aux | 2 | 6 | 1 | 1 | | | | | 65.8 | 64.6 | 56.7 |
| <i>inBC</i> + Aux | 2 | 10 | 1 | 1 | 1 | 1 | | | 69.2 | 66.9 | 55.8 |
| <i>inBC</i> + Aux | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 65.4 | 60.0 | 50.0 |
| <i>inBC</i> + Aux | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 68.1 | 65.0 | 51.7 |
| <i>inBC</i> + Aux | 4 | 12 | 1 | 1 | | | | | 65.8 | 62.7 | 49.6 |
| <i>inBC</i> + Aux | 4 | 20 | 1 | 1 | 1 | 1 | | | 65.8 | 63.1 | 52.5 |
| <i>inBC</i> + Aux | 4 | 24 | 1 | 1 | 1 | 1 | 1 | | 63.8 | 59.6 | 48.8 |
| <i>inBC</i> + Aux | 4 | 28 | 1 | 1 | 1 | 1 | 1 | 1 | 68.1 | 65.0 | 52.9 |
| <i>inBC</i> + Aux | 6 | 18 | 1 | 1 | | | | | 67.7 | 61.5 | 51.2 |
| <i>inBC</i> + Aux | 6 | 30 | 1 | 1 | 1 | 1 | | | 67.3 | 60.4 | 49.2 |
| <i>inBC</i> + Aux | 6 | 36 | 1 | 1 | 1 | 1 | 1 | | 61.9 | 60.4 | 47.1 |
| <i>inBC</i> + Aux | 6 | 42 | 1 | 1 | 1 | 1 | 1 | 1 | 65.4 | 63.8 | 52.9 |
| <i>inBC</i> + Aux | 8 | 24 | 1 | 1 | | | | | 67.3 | 62.7 | 52.1 |
| <i>inBC</i> + Aux | 8 | 40 | 1 | 1 | 1 | 1 | | | 66.9 | 61.2 | 51.7 |
| <i>inBC</i> + Aux | 8 | 48 | 1 | 1 | 1 | 1 | 1 | | 66.2 | 58.1 | 50.0 |
| <i>inBC</i> + Aux | 8 | 56 | 1 | 1 | 1 | 1 | 1 | 1 | 66.9 | 63.8 | 50.8 |
| <i>D-inBC</i> + Aux | 1 | 3 | 1 | 1 | | | | | 66.2 | 63.5 | 57.5 |
| <i>D-inBC</i> + Aux | 1 | 5 | 1 | 1 | 1 | 1 | | | 66.2 | 60.8 | 57.1 |
| <i>D-inBC</i> + Aux | 1 | 6 | 1 | 1 | 1 | 1 | 1 | | 65.4 | 60.4 | 52.5 |
| <i>D-inBC</i> + Aux | 1 | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 63.8 | 61.9 | 54.2 |
| <i>D-inBC</i> + Aux | 2 | 6 | 1 | 1 | | | | | 68.5 | 61.9 | 54.6 |
| <i>D-inBC</i> + Aux | 2 | 10 | 1 | 1 | 1 | 1 | | | 70.0 | 65.4 | 56.2 |
| <i>D-inBC</i> + Aux | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 69.6 | 62.7 | 56.2 |
| <i>D-inBC</i> + Aux | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 70.4 | 63.5 | 56.7 |
| <i>D-inBC</i> + Aux | 4 | 12 | 1 | 1 | | | | | 67.7 | 65.4 | 60.0 |
| <i>D-inBC</i> + Aux | 4 | 20 | 1 | 1 | 1 | 1 | | | 68.1 | 66.5 | 57.5 |
| <i>D-inBC</i> + Aux | 4 | 24 | 1 | 1 | 1 | 1 | 1 | | 66.5 | 61.2 | 52.9 |
| <i>D-inBC</i> + Aux | 4 | 28 | 1 | 1 | 1 | 1 | 1 | 1 | 69.6 | 58.5 | 52.9 |
| <i>D-inBC</i> + Aux | 6 | 18 | 1 | 1 | | | | | 69.2 | 63.8 | 57.9 |
| <i>D-inBC</i> + Aux | 6 | 30 | 1 | 1 | 1 | 1 | | | 70.0 | 68.1 | 58.8 |
| <i>D-inBC</i> + Aux | 6 | 36 | 1 | 1 | 1 | 1 | 1 | | 66.5 | 62.3 | 47.5 |
| <i>D-inBC</i> + Aux | 6 | 42 | 1 | 1 | 1 | 1 | 1 | 1 | 70.0 | 61.9 | 54.2 |
| <i>D-inBC</i> + Aux | 8 | 24 | 1 | 1 | | | | | 67.7 | 61.5 | 50.8 |
| <i>D-inBC</i> + Aux | 8 | 40 | 1 | 1 | 1 | 1 | | | 69.6 | 66.5 | 57.1 |
| <i>D-inBC</i> + Aux | 8 | 48 | 1 | 1 | 1 | 1 | 1 | | 71.2 | 69.2 | 51.7 |
| <i>D-inBC</i> + Aux | 8 | 56 | 1 | 1 | 1 | 1 | 1 | 1 | 68.1 | 67.7 | 57.9 |

Table 19: Results on VIMA-80k dataset with Oracle. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|----------------------------------|----|----|------|------|------|------|------|-------|--------|--------|--------|
| <i>D-RT-2 Style</i> + OD | 1 | 1 | | | | | | | 49.2 | 43.1 | 36.2 |
| <i>D-RT-2 Style</i> + OD | 2 | 2 | | | | | | | 52.3 | 50.0 | 45.8 |
| <i>D-RT-2 Style</i> + OD | 4 | 4 | | | | | | | 52.3 | 46.5 | 37.9 |
| <i>D-RT-2 Style</i> + OD | 6 | 6 | | | | | | | 47.3 | 47.3 | 36.2 |
| <i>D-RT-2 Style</i> + OD | 8 | 8 | | | | | | | 44.2 | 42.3 | 36.7 |
| <i>D-inBC</i> + OD | 1 | 1 | | | | | | | 81.9 | 75.0 | 67.5 |
| <i>D-inBC</i> + OD | 2 | 2 | | | | | | | 78.8 | 75.0 | 69.2 |
| <i>D-inBC</i> + OD | 4 | 4 | | | | | | | 81.2 | 75.4 | 68.3 |
| <i>D-inBC</i> + OD | 6 | 6 | | | | | | | 81.9 | 75.8 | 67.1 |
| <i>D-inBC</i> + OD | 8 | 8 | | | | | | | 83.8 | 73.8 | 69.2 |
| <i>D-inBC</i> + Aux + OD | 1 | 3 | 1 | 1 | | | | | 85.0 | 79.2 | 65.8 |
| <i>D-inBC</i> + Aux + OD | 1 | 5 | 1 | 1 | 1 | 1 | | | 85.0 | 77.7 | 68.3 |
| <i>D-inBC</i> + Aux + OD | 1 | 6 | 1 | 1 | 1 | 1 | 1 | | 79.6 | 73.5 | 64.2 |
| <i>D-inBC</i> + Aux + OD | 1 | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 84.2 | 74.6 | 66.2 |
| <i>D-inBC</i> + Aux + OD | 2 | 6 | 1 | 1 | | | | | 81.5 | 75.8 | 64.2 |
| <i>D-inBC</i> + Aux + OD | 2 | 10 | 1 | 1 | 1 | 1 | | | 86.2 | 83.1 | 69.6 |
| <i>D-inBC</i> + Aux + OD | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 80.0 | 73.1 | 62.5 |
| <i>D-inBC</i> + Aux + OD | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 85.8 | 77.3 | 66.7 |
| <i>D-inBC</i> + Aux + OD | 4 | 12 | 1 | 1 | | | | | 78.5 | 73.8 | 68.8 |
| <i>D-inBC</i> + Aux + OD | 4 | 20 | 1 | 1 | 1 | 1 | | | 82.3 | 77.3 | 71.7 |
| <i>D-inBC</i> + Aux + OD | 4 | 24 | 1 | 1 | 1 | 1 | 1 | | 82.3 | 71.9 | 63.7 |
| <i>D-inBC</i> + Aux + OD | 4 | 28 | 1 | 1 | 1 | 1 | 1 | 1 | 83.5 | 71.5 | 62.5 |
| <i>D-inBC</i> + Aux + OD | 6 | 18 | 1 | 1 | | | | | 80.0 | 75.0 | 67.1 |
| <i>D-inBC</i> + Aux + OD | 6 | 30 | 1 | 1 | 1 | 1 | | | 86.2 | 81.5 | 69.6 |
| <i>D-inBC</i> + Aux + OD | 6 | 36 | 1 | 1 | 1 | 1 | 1 | | 79.6 | 74.6 | 64.6 |
| <i>D-inBC</i> + Aux + OD | 6 | 42 | 1 | 1 | 1 | 1 | 1 | 1 | 85.0 | 77.3 | 62.1 |
| <i>D-inBC</i> + Aux + OD | 8 | 24 | 1 | 1 | | | | | 78.5 | 73.1 | 68.3 |
| <i>D-inBC</i> + Aux + OD | 8 | 40 | 1 | 1 | 1 | 1 | | | 86.5 | 82.7 | 70.8 |
| <i>D-inBC</i> + Aux + OD | 8 | 48 | 1 | 1 | 1 | 1 | 1 | | 82.3 | 78.1 | 65.4 |
| <i>D-inBC</i> + Aux + OD | 8 | 56 | 1 | 1 | 1 | 1 | 1 | 1 | 82.7 | 76.5 | 68.3 |
| <i>D-RT-2 Style</i> + Oracle | 1 | 1 | | | | | | | 71.2 | 66.5 | 61.3 |
| <i>D-RT-2 Style</i> + Oracle | 2 | 2 | | | | | | | 74.2 | 70.4 | 69.2 |
| <i>D-RT-2 Style</i> + Oracle | 4 | 4 | | | | | | | 73.1 | 70.4 | 63.3 |
| <i>D-RT-2 Style</i> + Oracle | 6 | 6 | | | | | | | 67.7 | 64.2 | 60.4 |
| <i>D-RT-2 Style</i> + Oracle | 8 | 8 | | | | | | | 69.2 | 61.5 | 55.4 |
| <i>D-RT-2 Style (I)</i> + Oracle | 1 | 1 | | | | | | | 57.7 | 54.6 | 57.1 |
| <i>D-RT-2 Style (I)</i> + Oracle | 2 | 2 | | | | | | | 55.8 | 54.6 | 55.4 |
| <i>D-RT-2 Style (I)</i> + Oracle | 4 | 4 | | | | | | | 58.1 | 54.2 | 55.4 |
| <i>D-RT-2 Style (I)</i> + Oracle | 6 | 6 | | | | | | | 53.5 | 49.6 | 47.9 |
| <i>D-RT-2 Style (I)</i> + Oracle | 8 | 8 | | | | | | | 53.8 | 50.8 | 48.8 |
| <i>D-inBC</i> + Oracle | 1 | 1 | | | | | | | 84.6 | 84.2 | 80.8 |
| <i>D-inBC</i> + Oracle | 2 | 2 | | | | | | | 84.2 | 82.3 | 77.5 |
| <i>D-inBC</i> + Oracle | 4 | 4 | | | | | | | 85.4 | 83.8 | 80.0 |
| <i>D-inBC</i> + Oracle | 6 | 6 | | | | | | | 86.9 | 85.8 | 80.8 |
| <i>D-inBC</i> + Oracle | 8 | 8 | | | | | | | 87.3 | 85.4 | 82.1 |
| <i>D-inBC</i> + Aux + Oracle | 1 | 3 | 1 | 1 | | | | | 88.8 | 85.0 | 79.2 |
| <i>D-inBC</i> + Aux + Oracle | 1 | 5 | 1 | 1 | 1 | 1 | | | 86.2 | 84.6 | 79.2 |
| <i>D-inBC</i> + Aux + Oracle | 1 | 6 | 1 | 1 | 1 | 1 | 1 | | 82.3 | 77.3 | 69.6 |
| <i>D-inBC</i> + Aux + Oracle | 1 | 7 | 1 | 1 | 1 | 1 | 1 | 1 | 86.2 | 83.5 | 77.5 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 6 | 1 | 1 | | | | | 88.5 | 84.2 | 77.5 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 10 | 1 | 1 | 1 | 1 | | | 91.5 | 88.1 | 80.4 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 12 | 1 | 1 | 1 | 1 | 1 | | 81.2 | 80.4 | 67.5 |
| <i>D-inBC</i> + Aux + Oracle | 2 | 14 | 1 | 1 | 1 | 1 | 1 | 1 | 88.5 | 84.6 | 78.3 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 12 | 1 | 1 | | | | | 85.0 | 87.3 | 78.8 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 20 | 1 | 1 | 1 | 1 | | | 88.8 | 84.6 | 79.6 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 24 | 1 | 1 | 1 | 1 | 1 | | 81.5 | 77.3 | 65.8 |
| <i>D-inBC</i> + Aux + Oracle | 4 | 28 | 1 | 1 | 1 | 1 | 1 | 1 | 84.2 | 80.0 | 71.7 |
| <i>D-inBC</i> + Aux + Oracle | 6 | 18 | 1 | 1 | | | | | 84.6 | 85.4 | 76.7 |
| <i>D-inBC</i> + Aux + Oracle | 6 | 30 | 1 | 1 | 1 | 1 | | | 87.7 | 87.3 | 78.3 |
| <i>D-inBC</i> + Aux + Oracle | 6 | 36 | 1 | 1 | 1 | 1 | 1 | | 81.2 | 80.8 | 70.8 |
| <i>D-inBC</i> + Aux + Oracle | 6 | 42 | 1 | 1 | 1 | 1 | 1 | 1 | 88.8 | 81.2 | 70.0 |
| <i>D-inBC</i> + Aux + Oracle | 8 | 24 | 1 | 1 | | | | | 86.9 | 81.5 | 75.8 |
| <i>D-inBC</i> + Aux + Oracle | 8 | 40 | 1 | 1 | 1 | 1 | | | 90.0 | 88.1 | 79.2 |
| <i>D-inBC</i> + Aux + Oracle | 8 | 48 | 1 | 1 | 1 | 1 | 1 | | 82.3 | 83.1 | 70.4 |
| <i>D-inBC</i> + Aux + Oracle | 8 | 56 | 1 | 1 | 1 | 1 | 1 | 1 | 83.8 | 88.1 | 78.8 |

1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943

Table 20: Results on full VIMA dataset with Oracle. **Ep**: number of epoch; **U**: normalized number of model iterations. The following sizes of datasets are relative to the size of *inBC*. **Loc.**: relative size of the localization dataset; **Det.**: relative size of the detection dataset; **Act.**: relative size of the action prediction dataset; **Fut.**: relative size of the future prediction dataset; **Spa.**: relative size of the spatial relationship dataset; **Temp.**: relative size of the temporal relationship dataset.

| Method | Ep | U | Loc. | Det. | Act. | Fut. | Spa. | Temp. | L1 (%) | L2 (%) | L3 (%) |
|----------------------------------|----|---|------|------|------|------|------|-------|--------|--------|--------|
| <i>RT-2 Style</i> | 1 | 1 | | | | | | | 61.9 | 55.8 | 47.1 |
| <i>RT-2 Style</i> | 2 | 2 | | | | | | | 65.0 | 59.6 | 42.5 |
| <i>RT-2 Style</i> | 3 | 3 | | | | | | | 63.1 | 57.7 | 44.6 |
| <i>RT-2 Style</i> | 4 | 4 | | | | | | | 64.2 | 60.0 | 44.2 |
| <i>D-RT-2 Style + Oracle</i> | 1 | 1 | | | | | | | 82.3 | 80.8 | 74.6 |
| <i>D-RT-2 Style + Oracle</i> | 2 | 2 | | | | | | | 86.2 | 88.1 | 75.8 |
| <i>D-RT-2 Style + Oracle</i> | 3 | 3 | | | | | | | 87.7 | 83.5 | 76.2 |
| <i>D-RT-2 Style + Oracle</i> | 4 | 4 | | | | | | | 87.3 | 85.4 | 74.6 |
| <i>D-RT-2 Style (I) + Oracle</i> | 1 | 1 | | | | | | | 68.8 | 68.1 | 59.6 |
| <i>D-RT-2 Style (I) + Oracle</i> | 2 | 2 | | | | | | | 68.5 | 66.2 | 58.3 |
| <i>D-RT-2 Style (I) + Oracle</i> | 3 | 3 | | | | | | | 70.0 | 70.0 | 59.2 |
| <i>D-RT-2 Style (I) + Oracle</i> | 4 | 4 | | | | | | | 69.6 | 69.2 | 58.3 |

Table 21: Prompt candidates for action generation during inference.

‘Could you write down what needs to be done to complete the task on this scene?’
‘List out the actions needed to accomplish the task in this scene.’
‘What actions are necessary to perform the task on this scene?’
‘Can you describe what needs to be done on this scene to complete the task?’
‘What steps are required to perform the task shown in this scene?’
‘List the actions needed to perform the task given below.’
‘On the following scene, could you list what actions are required to perform the task?’
‘Describe what actions are needed on this scene to complete the task.’
‘What do you need to do on this scene to accomplish the task?’
‘List the actions required to perform the task given on this scene.’
‘Could you please describe the steps needed to perform the task on this scene?’
‘Write down the actions required to perform the task on this scene.’
‘Please write down the actions required to perform the task shown below.’
‘Can you explain what needs to be done to perform the task in this scene?’
‘Describe the actions required to complete the task on this scene.’

1944 Table 22: Prompt candidates for object localization. {object} will be replaced with the actual object name in
 1945 the dataset.

1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997

‘Where is {object} located in the image? Please use the format `(x, y),{w, h}` where x and y represent the center coordinates of the bounding box, and w and h are the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘Can you provide the location of {object} in the image? Format it as `(x, y),{w, h}`, with x and y as the center coordinates of the bounding box and w and h as the width and height. The coordinates should begin at the top left corner and be normalized from 0 to 1.’

‘What are the coordinates of {object} in the image? Use the format `(x, y),{w, h}`, where x and y are the center of the bounding box, and w and h represent the width and height. Coordinates should start at the top left corner and be normalized to a range of 0 to 1.’

‘Please specify the location of {object} in the image. List it in the format `(x, y),{w, h}`, where x and y denote the bounding box center coordinates, and w and h are the width and height. The coordinates begin from the top left corner and should be normalized to 0 to 1.’

‘What is the position of {object} within the image? Use the format `(x, y),{w, h}` to describe it, with x and y as the center coordinates of the bounding box, and w and h as the width and height. The coordinates start at the top left corner and are normalized to a scale of 0 to 1.’

‘Describe the location of {object} in the image using the format `(x, y),{w, h}`. In this format, x and y denote the center coordinates of the bounding box, while w and h represent its width and height. Coordinates should be normalized from the top left corner, ranging from 0 to 1.’

‘Can you detail the location of {object} in the image? Format it as `(x, y),{w, h}`, where x and y indicate the bounding box center, and w and h represent the width and height. The coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

‘Provide the location of {object} in the image using the format `(x, y),{w, h}`. Here, x and y are the center coordinates of the bounding box, and w and h are the width and height. The coordinates begin at the top left corner and are normalized from 0 to 1.’

‘Where is {object} positioned in the image? Use the format `(x, y),{w, h}`, where x and y denote the center coordinates of the bounding box, and w and h are the width and height. The coordinates should be normalized to a range of 0 to 1 starting from the top left corner.’

‘Specify the location of {object} in the image in the format `(x, y),{w, h}`. In this format, x and y represent the bounding box center, and w and h are the width and height. The coordinates should start from the top left corner and be normalized between 0 and 1.’

‘What is the exact position of {object} in the image? Format the coordinates as `(x, y),{w, h}`, where x and y are the center of the bounding box and w and h denote its width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘Describe where {object} is located in the image using the format `(x, y),{w, h}`. Here, x and y indicate the bounding box center coordinates, and w and h specify its width and height. The coordinates should be normalized starting from the top left corner, within the range of 0 to 1.’

‘Could you tell me the location of {object} in the image? Use the format `(x, y),{w, h}`, where x and y denote the center of the bounding box and w and h are the width and height. Coordinates start at the top left corner and should be normalized between 0 and 1.’

‘Provide the coordinates of {object} in the image in the format `(x, y),{w, h}`. Here, x and y are the center of the bounding box, while w and h represent its width and height. The coordinates should start from the top left corner and be normalized to 0 to 1.’

‘How is the {object} located in the image? List its coordinates using the format `(x, y),{w, h}`, where x and y are the center coordinates of the bounding box, and w and h indicate its width and height. The coordinates begin at the top left corner and are normalized to a range of 0 to 1.’

1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051

Table 23: Prompt candidates for object detection.

‘Identify and describe each object in the image. For each object, list it in the format `(x, y),{w, h}`, where x and y represent the coordinates of the bounding box center, and w and h represent the width and height of the bounding box. The image coordinates should start from the top left corner and be normalized between 0 and 1.’

‘Catalog all the objects present in the image. For every object, use the format `(x, y),{w, h}`, with x and y indicating the center of the object’s bounding box coordinates, and w and h specifying the width and height. The coordinates are normalized from the top left corner, ranging from 0 to 1.’

‘List each object in the image and describe it. Use the format `(x, y),{w, h}` for each object, where x and y denote the center coordinates of the bounding box, and w and h are the width and height of the bounding box. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Provide descriptions for all objects within the image. Each object should be listed using the format `(x, y),{w, h}`, where x and y are the coordinates of the bounding box center, and w and h are the width and height. The coordinates should be normalized, starting from the top left corner, within a range of 0 to 1.’

‘Enumerate and describe every object found in the image. For each object, utilize the format `(x, y),{w, h}`, where x, y are the bounding box center coordinates and w, h are the dimensions (width and height) of the bounding box. The coordinates begin at the top left corner and are normalized between 0 and 1.’

‘Detail all the objects within the image, listing each one using the format `(x, y),{w, h}`. Here, x and y represent the coordinates of the bounding box center, while w and h indicate the width and height. The coordinates start from the top left corner and are normalized to the range of 0 to 1.’

‘Document each object present in the image. For each object, use the format `(x, y),{w, h}`, where x and y are the coordinates of the center of the bounding box, and w and h are the width and height. The coordinates should be normalized, starting from the top left corner, and range from 0 to 1.’

‘For each object in the image, provide a description using the format `(x, y),{w, h}`. Here, x and y denote the coordinates of the bounding box center, and w and h represent the width and height of the bounding box. The coordinates are normalized to a scale of 0 to 1, starting from the top left corner.’

‘Describe all the objects seen in the image, and list them using the format `(x, y),{w, h}`. The x and y values are the coordinates for the center of the bounding box, while w and h represent its width and height. The coordinates should be normalized from the top left corner, within a range of 0 to 1.’

‘Identify and list each object found in the image. For each one, use the format `(x, y),{w, h}`. In this format, x and y are the coordinates for the bounding box center, and w and h are the width and height. The coordinates are to be normalized starting from the top left corner, ranging from 0 to 1.’

‘List and describe each object in the image using the format `(x, y),{w, h}`. Here, x and y correspond to the coordinates of the bounding box center, and w and h specify the width and height of the bounding box. The coordinates should start from the top left corner and be normalized to the range of 0 to 1.’

‘Provide a description for each object in the image, formatted as `(x, y),{w, h}`. The x and y values indicate the center coordinates of the bounding box, while w and h represent the width and height. The coordinates start from the top left corner and are normalized between 0 and 1.’

2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105

Table 24: (Continued) Prompt candidates for object detection.

‘Catalog each object within the image, using the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$ for each one. In this format, x and y are the coordinates for the center of the bounding box, and w and h are the width and height. The coordinates should be normalized, beginning at the top left corner and ranging from 0 to 1.’

‘Enumerate all the objects in the image, providing descriptions for each using the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$. The x and y values represent the center coordinates of the bounding box, while w and h indicate its width and height. The coordinates are normalized starting from the top left corner, within a range of 0 to 1.’

‘Describe each object in the image, listing them in the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$. Here, x and y denote the center coordinates of the bounding box, and w and h specify the width and height. The coordinates should be normalized from the top left corner, ranging from 0 to 1.’

Table 25: Prompt candidates for action prediction. {scene} will be replaced with a list of objects in the scene.

‘Could you detail the steps needed to transform the scene shown in the image into the second scene? The second scene is provided as a collection of object bounding boxes {scene}. The format for these bounding boxes is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h are the width and height. The coordinates should be normalized to a scale of 0 to 1, starting from the top left corner.’

‘Can you describe what actions are required to rearrange the scene shown in the image to match the second scene? The second scene is given as a set of object bounding boxes {scene}. These bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y indicate the center coordinates, and w and h represent the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Could you list the steps necessary to modify the scene shown in the image to the second scene? The second scene is described as a collection of object bounding boxes {scene}. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, with x and y denoting the center coordinates, and w and h representing the width and height. The coordinates are normalized to a scale of 0 to 1, starting from the top left corner.’

‘Can you explain what needs to be done to adjust the scene shown in the image to resemble the second scene? The second scene {scene} consists of object bounding boxes provided in the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$. Here, x and y represent the center coordinates, and w and h are the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Could you outline the necessary actions to arrange the scene shown in the image into the second scene? The second scene is defined by a collection of object bounding boxes {scene}. These bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y denote the center coordinates, and w and h are the width and height. The coordinates start from the top left corner and should be normalized to a scale of 0 to 1.’

‘Can you specify what needs to be done to convert the scene shown in the image into the second scene? The second scene is provided as a series of object bounding boxes {scene}. The format for these bounding boxes is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, with x and y representing the center coordinates, and w and h indicating the width and height. Coordinates should be normalized from the top left corner to a scale of 0 to 1.’

‘Could you describe the steps required to change the scene shown in the image to the second scene? The second scene is depicted as a collection of object bounding boxes {scene}. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y denote the center coordinates, and w and h represent the width and height. The coordinates are normalized to a scale of 0 to 1 starting from the top left corner.’

2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159

Table 26: (Continued) Prompt candidates for action prediction. {scene} will be replaced with a list of objects in the scene.

‘Can you list the actions necessary to transform the scene shown in the image into the second scene? The second scene is described using object bounding boxes {scene}. The format of these bounding boxes is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y are the center coordinates, and w and h represent the width and height. Coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you explain the process to arrange the scene shown in the image to match the second scene? The second scene is provided as a collection of object bounding boxes {scene}. These bounding boxes are formatted as $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h are the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Can you detail what needs to be done to rearrange the scene shown in the image to the second scene? The second scene is given as a series of object bounding boxes {scene}. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y denote the center coordinates, and w and h represent the width and height. Coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you specify the steps needed to modify the scene shown in the image to resemble the second scene? The second scene is described as a set of object bounding boxes {scene}. These bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h indicate the width and height. The coordinates start from the top left corner and should be normalized to a scale of 0 to 1.’

‘Can you outline the necessary actions to change the scene shown in the image into the second scene? The second scene {scene} consists of object bounding boxes provided in the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y denote the center coordinates, and w and h represent the width and height. Coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you describe the steps to adjust the scene shown in the image to the second scene? The second scene is given as a collection of object bounding boxes {scene}. The format for these bounding boxes is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h are the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Can you explain what needs to be done to transform the scene shown in the image into the second scene? The second scene is depicted using object bounding boxes {scene}. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, with x and y representing the center coordinates, and w and h indicating the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘Could you detail the steps necessary to convert the scene shown in the image to the second scene? The second scene is described as a set of object bounding boxes {scene}. These bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h denote the width and height. The coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

Table 27: Prompt candidates for future prediction. {pick and place} will be replaced with the action text.

2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213

‘The image shows a scene with multiple objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The bounding box format is $\langle b \rangle(x, y), \{w, h\}$, where x and y represent the center coordinates of the bounding box, and w and h are its width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘An image depicts a scene containing multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The bounding boxes should be formatted as $\langle b \rangle(x, y), \{w, h\}$, where x and y denote the center coordinates, and w and h are the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘The image presents a scene with several objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The format for these bounding boxes is $\langle b \rangle(x, y), \{w, h\}$, where x and y represent the center coordinates, and w and h are the width and height. Coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Displayed in the image is a scene containing multiple objects. Now you {pick and place}, what will the scene look like? Write down the list of object bounding boxes. These bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\}$, with x and y as the center coordinates, and w and h as the width and height. The coordinates should be normalized starting from the top left corner to a scale of 0 to 1.’

‘The image illustrates a scene with multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The bounding boxes are formatted as $\langle b \rangle(x, y), \{w, h\}$, where x and y denote the center coordinates, and w and h represent the width and height. Coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘The image depicts a scene with several objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The bounding box format is $\langle b \rangle(x, y), \{w, h\}$, where x and y represent the center coordinates, and w and h denote the width and height. The coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

‘In the image, there is a scene with multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The format of these bounding boxes is $\langle b \rangle(x, y), \{w, h\}$, where x and y indicate the center coordinates, and w and h represent the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘An image shows a scene with various objects. Now you {pick and place}, what will the scene look like? Write down the list of object bounding boxes. The bounding boxes follow the format $\langle b \rangle(x, y), \{w, h\}$, where x and y denote the center coordinates, and w and h are the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘The image presents a scene containing several objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The bounding box format is $\langle b \rangle(x, y), \{w, h\}$, where x and y represent the center coordinates, and w and h are the width and height. Coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘The image displays a scene with multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The bounding boxes should be in the format $\langle b \rangle(x, y), \{w, h\}$, where x and y denote the center coordinates, and w and h represent the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘An image illustrates a scene with multiple objects. Now you {pick and place}, what will the scene look like? Write down the list of object bounding boxes. These bounding boxes are formatted as $\langle b \rangle(x, y), \{w, h\}$, where x and y represent the center coordinates, and w and h denote the width and height. Coordinates should be normalized to a scale of 0 to 1 starting from the top left corner.’

2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267

Table 28: (Continued) Prompt candidates for future prediction. {pick and place} will be replaced with the action text.

‘The image shows a scene with various objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The format for these bounding boxes is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, with x and y representing the center coordinates, and w and h as the width and height. Coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

‘Displayed in the image is a scene containing multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y denote the center coordinates, and w and h are the width and height. The coordinates start from the top left corner and are normalized to a scale of 0 to 1.’

‘The image illustrates a scene with various objects. Now you {pick and place}, what will the scene look like? List the object bounding boxes. The bounding boxes are formatted as $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y indicate the center coordinates, and w and h represent the width and height. Coordinates should be normalized from the top left corner to a scale of 0 to 1.’

‘An image depicts a scene with multiple objects. Now you {pick and place}, what will the scene look like? Write the list of object bounding boxes. The bounding box format is $\langle b \rangle(x, y), \{w, h\} \langle /b \rangle$, where x and y represent the center coordinates, and w and h denote the width and height. The coordinates should start from the top left corner and be normalized to a scale of 0 to 1.’

Table 29: Prompt candidates for spatial relationship. {ego_obj} and {ref_obj} will be replaced with object names and {example} will be replaced with a random spatial relationship from the same image.

‘Can you describe the relative spatial locations of {ego_obj} compared to {ref_obj} in this image? Use relative location words like left, right, above, below, etc. Also, find the 2D center distance and the Euclidean center distance between them. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you describe the relative spatial positions of {ego_obj} in comparison to {ref_obj} in this image? Use terms like left, right, above, below, etc. Also, calculate the 2D center distance and the Euclidean center distance between them. Your output should be formatted as follows: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Please describe the relative spatial locations of {ego_obj} compared to {ref_obj} in this image. Use words like left, right, above, below, etc. Additionally, find the 2D center distance and the Euclidean center distance between them. Your output must be in this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Can you explain the relative spatial positions of {ego_obj} compared to {ref_obj} in this image? Use terms such as left, right, above, below, etc. Also, determine the 2D center distance and the Euclidean center distance between them. Your output should match this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Describe the relative spatial locations of {ego_obj} compared to {ref_obj} in this image using words like left, right, above, below, etc. Also, calculate the 2D center distance and the Euclidean center distance between them. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321

Table 30: (Continued) Prompt candidates for spatial relationship. {ego_obj} and {ref_obj} will be replaced with object names and {example} will be replaced with a random spatial relationship from the same image.

‘Could you describe the spatial relationship between {ego_obj} and {ref_obj} in this image using relative location words like left, right, above, below, etc.? Also, find the 2D center distance and the Euclidean center distance between them. Your output should be formatted as follows: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Can you detail the relative spatial positions of {ego_obj} compared to {ref_obj} in this image? Use words like left, right, above, below, etc. Also, determine the 2D center distance and the Euclidean center distance between them. Your output must be in this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you explain the spatial relationship between {ego_obj} and {ref_obj} in this image using terms such as left, right, above, below, etc.? Also, calculate the 2D center distance and the Euclidean center distance between them. Your output should match this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Describe the relative spatial positions of {ego_obj} compared to {ref_obj} in this image. Use relative location words like left, right, above, below, etc. Also, find the 2D center distance and the Euclidean center distance between them. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Can you describe how {ego_obj} is positioned relative to {ref_obj} in this image using words such as left, right, above, below, etc.? Also, find the 2D center distance and the Euclidean center distance between them. Your output should be in this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you detail the relative positions of {ego_obj} compared to {ref_obj} in this image using terms like left, right, above, below, etc.? Also, calculate the 2D center distance and the Euclidean center distance between them. Your output must be formatted as follows: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Please describe the spatial relationship of {ego_obj} in comparison to {ref_obj} in this image using relative location terms such as left, right, above, below, etc. Additionally, find the 2D center distance and the Euclidean center distance between them. Your output should match this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Can you describe the relative spatial locations of {ego_obj} compared to {ref_obj} in this image? Use relative location words like left, right, above, below, etc. Also, calculate the 2D center distance and the Euclidean center distance between them. Your output should follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘Could you describe the spatial locations of {ego_obj} relative to {ref_obj} in this image using words such as left, right, above, below, etc.? Additionally, find the 2D center distance and the Euclidean center distance between them. Your output must be in this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

2322
 2323
 2324
 2325
 2326
 2327
 2328
 2329
 2330
 2331
 2332
 2333
 2334
 2335
 2336
 2337
 2338
 2339
 2340
 2341
 2342
 2343
 2344
 2345
 2346
 2347
 2348
 2349
 2350
 2351
 2352
 2353
 2354
 2355
 2356
 2357
 2358
 2359
 2360
 2361
 2362
 2363
 2364
 2365
 2366
 2367
 2368
 2369
 2370
 2371
 2372
 2373
 2374
 2375

Table 31: Prompt candidates for temporal relationship {scene} will be replaced with a list of objects, {ego_obj} and {ref_obj} will be replaced with object names and {example} will be replaced with a random temporal relationship from the same image.

"The image shows a scene at the first timestamp, while the second image described as {scene} shows the next timestamp. Can you describe the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use relative distance words like getting closer or further away, etc. Also, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'In the first timestamp, the image shows a scene, and the second image described as {scene} depicts the next timestamp. Can you describe the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use terms like getting closer or moving further away, etc. Additionally, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'The scene in the first image is at the initial timestamp, and the second image described as {scene} shows the subsequent timestamp. Can you explain the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use words like getting closer or moving further apart, etc. Also, calculate the change in the 2D center distance and the Euclidean center distance between the two images. Your output should be formatted as follows: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'At the first timestamp, the image shows a scene, and the second image described as {scene} represents the next timestamp. Can you detail the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use relative distance words like moving closer or getting further away, etc. Additionally, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'The first image shows a scene at an initial timestamp, and the second image described as {scene} depicts the next timestamp. Can you describe the change in the relative position of {ego_obj} compared to {ref_obj} between these two timestamps? Use terms such as getting closer or moving further apart, etc. Also, determine the change in the 2D center distance and the Euclidean center distance between the two images. Your output should follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'The initial timestamp shows a scene in the first image, and the second image described as {scene} represents the next timestamp. Can you describe how the relative location of {ego_obj} compared to {ref_obj} changes between these two timestamps? Use relative distance words like getting closer or moving further away, etc. Also, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must be in this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

'The image shows a scene at the first timestamp, and the second image described as {scene} shows the subsequent timestamp. Can you detail the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use words like getting closer or moving further apart, etc. Also, calculate the change in the 2D center distance and the Euclidean center distance between the two images. Your output should be formatted as follows: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.'

2376
 2377
 2378
 2379
 2380
 2381
 2382
 2383
 2384
 2385
 2386
 2387
 2388
 2389
 2390
 2391
 2392
 2393
 2394
 2395
 2396
 2397
 2398
 2399
 2400
 2401
 2402
 2403
 2404
 2405
 2406
 2407
 2408
 2409
 2410
 2411
 2412
 2413
 2414
 2415
 2416
 2417
 2418
 2419
 2420
 2421
 2422
 2423
 2424
 2425
 2426
 2427
 2428
 2429

Table 32: (Continued) Prompt candidates for temporal relationship {scene} will be replaced with a list of objects, {ego_obj} and {ref_obj} will be replaced with object names and {example} will be replaced with a random temporal relationship from the same image.

‘At the initial timestamp, the image shows a scene, and the second image described as {scene} depicts the next timestamp. Can you describe the change in the relative position of {ego_obj} compared to {ref_obj} between these two timestamps? Use relative distance terms such as getting closer or moving further away, etc. Also, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘The scene in the first image is at the initial timestamp, and the second image described as {scene} shows the following timestamp. Can you describe the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use words like getting closer or moving further apart, etc. Additionally, calculate the change in the 2D center distance and the Euclidean center distance between the two images. Your output should follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘The first image shows a scene at an initial timestamp, and the second image described as {scene} depicts the next timestamp. Can you explain how the relative location of {ego_obj} compared to {ref_obj} changes between these two timestamps? Use relative distance words like moving closer or getting further away, etc. Also, determine the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘The image shows a scene at the initial timestamp, and the second image described as {scene} shows the next timestamp. Can you describe the change in the relative position of {ego_obj} compared to {ref_obj} between these two timestamps? Use words like getting closer or moving further apart, etc. Also, calculate the change in the 2D center distance and the Euclidean center distance between the two images. Your output should follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘At the first timestamp, the image shows a scene, and the second image described as {scene} depicts the next timestamp. Can you detail how the relative location of {ego_obj} compared to {ref_obj} changes between these two timestamps? Use relative distance terms such as moving closer or getting further away, etc. Also, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘The initial timestamp shows a scene in the first image, and the second image described as {scene} represents the next timestamp. Can you describe the change in the relative location of {ego_obj} compared to {ref_obj} between these two timestamps? Use words like getting closer or moving further apart, etc. Also, determine the change in the 2D center distance and the Euclidean center distance between the two images. Your output should follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’

‘The first image shows a scene at the initial timestamp, and the second image described as {scene} shows the following timestamp. Can you describe how the relative position of {ego_obj} compared to {ref_obj} changes between these two timestamps? Use terms like moving closer or getting further away, etc. Additionally, find the change in the 2D center distance and the Euclidean center distance between the two images. Your output must follow this format: {example}. The coordinates are image coordinates normalized to a scale of 0 to 1 starting from the top left corner.’