
BridgeVLA: Input-Output Alignment for Efficient 3D Manipulation Learning with Vision-Language Models

Peiyan Li^{1,2,3,*}, Yixiang Chen^{1,3}, Hongtao Wu^{2,*,†}, Xiao Ma^{2,*}, Xiangnan Wu¹
Yan Huang^{1,3,4,†}, Liang Wang^{1,3}, Tao Kong², Tieniu Tan^{1,3,5}

¹New Laboratory of Pattern Recognition (NLPR),

Institute of Automation, Chinese Academy of Sciences

²ByteDance Seed ³School of Artificial Intelligence, University of Chinese Academy of Sciences

⁴FiveAges ⁵Nanjing University

Abstract

Recently, leveraging pre-trained vision-language models (VLMs) for building vision-language-action (VLA) models has emerged as a promising approach to effective robot manipulation learning. However, only few methods incorporate 3D signals into VLMs for action prediction, and they do not fully leverage the spatial structure inherent in 3D data, leading to low data efficiency. In this paper, we introduce a new paradigm for constructing 3D VLAs. Specifically, we first pre-train the VLM backbone to take 2D images as input and produce 2D heatmaps as output. Using this pre-trained VLM as the backbone, we then fine-tune the entire VLA model while maintaining alignment between inputs and outputs by: (1) projecting raw point cloud inputs into multi-view images, and (2) predicting heatmaps before generating the final action. Extensive experiments show that the resulting model, BridgeVLA, can learn 3D manipulation both efficiently and effectively. BridgeVLA outperforms state-of-the-art baselines across three simulation benchmarks. In RL Bench, it improves the average success rate from 81.4% to 88.2%. In COLOSSEUM, it demonstrates significantly better performance in challenging generalization settings, boosting the average success rate from 56.7% to 64.0%. In Gem Bench, it surpasses all the comparing baseline methods in terms of average success rate. In real-robot experiments, BridgeVLA outperforms a state-of-the-art baseline method by 32% on average. It generalizes robustly in multiple out-of-distribution settings, including visual disturbances and unseen instructions. Remarkably, it is able to achieve a success rate of 95.4% on 10+ tasks with only 3 trajectories per task, while other VLA methods such as π_0 fail completely. Project Website: <https://bridgevla.github.io/>.

1 Introduction

Leveraging pre-trained vision-language models (VLMs) [3, 43, 2, 24] for developing large vision-language-action (VLA) models has become a promising method for learning generalizable and robust manipulation policies [26, 4, 17, 31, 7]. However, most VLA models only incorporate 2D image inputs and require extensive efforts on data collection. On the other hand, 3D robot policies leverage 3D structural priors in model design and demonstrate exceptional sample efficiency in learning complex 3D robot manipulation tasks [39, 25, 13–15]. Can we develop a unified 3D VLA model which combines the effectiveness of VLA models with the efficiency from 3D policies?

*Project lead

†Corresponding author

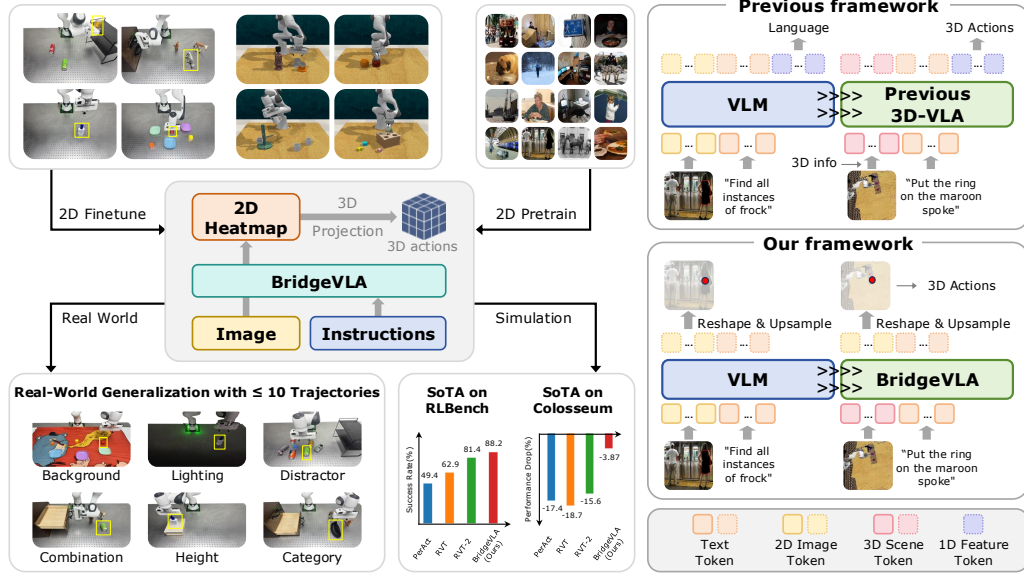


Figure 1: **Overview.** BridgeVLA is a novel 3D VLA model that aligns the input and output within a unified 2D image space. It is pre-trained on object grounding using 2D heatmaps and fine-tuned on action prediction for 3D manipulation. Experiment results in both simulation and the real world show that it is able to learn 3D manipulation both efficiently and effectively.

Although there have been some works exploring integrating 3D information into VLMs for developing 3D VLA models [52, 37], these works typically convert actions into token sequences that do not have spatial structure and use next-token prediction to predict actions. This strategy fails to take advantage of the 3D structural priors as previous efficient 3D policies [39, 25, 13–15] that align the observation input and action output into a unified space, therefore leading to poor sample efficiency. Another significant challenge in developing 3D VLA models lies in the misalignment between the 3D inputs used in action fine-tuning and the 2D image inputs used in original VLM pre-training, causing a large distributional shift from the original VLM pre-training.

To tackle the challenges mentioned above, as illustrated in Fig. 1, we present BridgeVLA, a novel 3D VLA model that achieves remarkable sample efficiency and strong generalization capabilities. To ensure input alignment with the pre-trained VLM backbone, BridgeVLA transforms a 3D point cloud observation into multiple 2D images captured from different orthographic projection views [14, 15]. To leverage the structural priors of the 3D input, BridgeVLA is trained to predict 2D heatmaps for translational action prediction. The 2D heatmaps, generated from the tokens corresponding to the projection images, share the same resolution as these images, aligning the input observations and output actions within a unified spatial structure. Given that the original VLM is pre-trained to predict token sequences, which is incompatible with our VLA’s 2D heatmap output, we also introduce a scalable pre-training method, which trains the model to ground objects with heatmaps conditioned on text inputs. This pre-training method equips the VLM with the capabilities to predict heatmaps before downstream fine-tuning for policy learning. **Overall, our design aligns the input and output within a shared 2D space in both pre-training and fine-tuning.**

We perform extensive experiments in both simulation and the real world to evaluate the proposed method. Results show that BridgeVLA is able to learn 3D manipulation both efficiently and effectively. It outperforms state-of-the-art baseline methods in RL Bench [19], improving the average success rate from 81.4% to 88.2%. In COLOSSEUM [35], it showcases strong performance in challenging generalization settings, boosting the success rate from 56.7% to 64.0%. In GemBench [12], it surpasses all the comparing baseline methods in terms of average success rate. In real-robot experiments, we evaluate on seven different settings, spanning from visual perturbations to manipulating objects from unseen categories. BridgeVLA surpasses a state-of-the-art method by 32% on average and demonstrates strong performance in generalizing to multiple out-of-distribution settings. Notably, BridgeVLA is able to achieve a success rate of 96.8% on 10+ tasks using only 3 trajectories per task

for training, highlighting its superb sample efficiency. In summary, the contributions of this paper are threefold:

- We introduce BridgeVLA, a novel 3D VLA model that efficiently and effectively learns 3D robot manipulation with a vision-language model via input-output alignment with 2D heatmaps.
- We propose a scalable pre-training method to equip the model with the capability to predict heatmaps conditioned on text inputs via object grounding.
- We conduct extensive experiments in both simulation and real-world environments to thoroughly evaluate the proposed method. Results show that BridgeVLA outperforms state-of-the-art methods in both settings and achieves exceptional sample efficiency in real-robot experiments.

2 Related Work

Language-Conditioned Visuomotor Policies. Most language-conditioned visuomotor policies employ transformers to process 2D visual inputs and directly generate 3D actions for manipulation [6, 7, 26, 4, 17, 10, 31, 30, 8, 51, 44]. In these works, leveraging pre-trained vision-language models (VLMs) for developing large vision-language-action (VLA) models has become popular for its effectiveness on learning complex manipulation [7, 26, 31, 4, 17]. However, such 2D image-based policies typically require significant efforts on data collection, often needing hundreds of trajectories per task to learn effectively. On the other hand, 3D manipulation policies hold great potential for efficient learning by taking advantage of the spatial structure inherent in the 3D inputs. A popular line of works take as inputs point cloud data [9, 48, 47, 13, 25]. For example, Act3D [13] proposes to create a 3D feature cloud by lifting image features to the observation point cloud and predicts translational actions via classification for 3D points in the observation space. Another line of works utilize voxels to represent the observation space and predict translational actions within the voxel space, unifying the input observation and output actions within the same space [39, 20]. Recently, RVT [14] and RVT-2 [15] propose to leverage orthographic projection of 3D point clouds to convert 3D signals to 2D images to avoid high computational cost on processing 3D inputs. Different from the above methods, our method aims to unify the effectiveness of VLA models and the efficiency of 3D policies within a single cohesive framework, combining the best of both worlds.

3D Vision-Language-Action (VLA) Models. While 2D VLA models have been extensively studied, 3D VLA models [52, 22, 47, 29] remain relatively under-explored. Zhen *et al.* [52] build 3D-VLA on top of a large language model (LLM) and train the model to perform 3D reasoning, multi-modal goal generation, and robot planning. Lift3D [22] proposes to enhance 2D foundation models (*e.g.*, DINOv2 [34]) with implicit and explicit 3D robotic representation for learning 3D manipulation policies. FP3 [47] leverages a transformer to fuse the information from point clouds, proprioceptive states, and language instructions. PointVLA [29] utilizes a VLM and a point cloud encoder to process 2D images and 3D point clouds, respectively. The embeddings from both encoders are injected into an action expert for action prediction. SpatialVLA [37] introduces Ego3D position encoding to inject 3D information into 2D image observation and adaptive action grids to represent robot movement in a more transferable way. Our method is different from the above methods in that it is designed in a way to take advantage of the spatial structure of 3D inputs in action prediction. In addition, it bridges the gap between the 2D image inputs of pre-trained VLMs and the 3D inputs by projecting the 3D inputs into multiple 2D images instead of injecting 3D information into the VLMs. Such design enables it to simultaneously leverages the broad knowledge in the VLM backbone and the spatial structure priors embedded in 3D inputs.

3 BridgeVLA

3.1 Preliminaries

BridgeVLA aims to learn a multi-task 3D robot manipulation policy π , which maps the observation $\mathbf{o}$ and a language instruction l to an action $\mathbf{a}$:

$$\pi : (\mathbf{o}, l) \mapsto \mathbf{a} \quad (1)$$

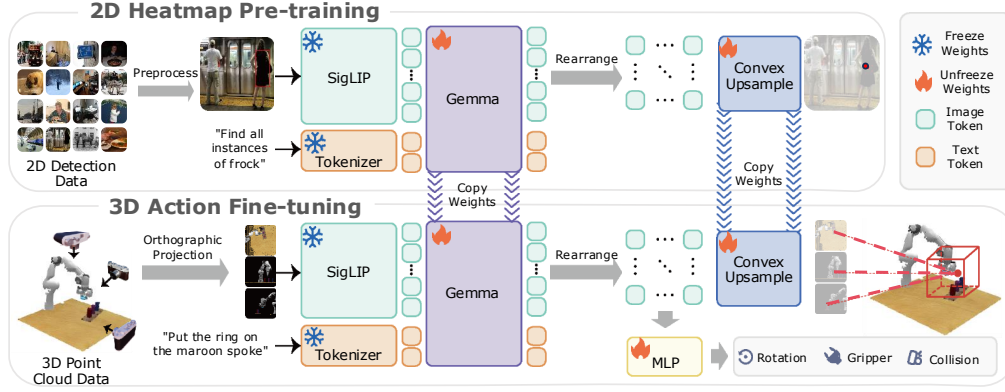


Figure 2: **Model Architecture.** (a) **2D Heatmap Pre-training:** we train BridgeVLA on 2D object detection datasets. The model takes as inputs an image and a language describing the target object and outputs a 2D heatmap which highlights regions of interest that correspond to the target object. Note that the bounding box shown here is for illustrative purposes only; it is not present in the image when input to the model. (b) **3D Action Fine-tuning:** the model takes as inputs three orthographic projection images of a 3D point cloud and a language instruction. It outputs three 2D heatmaps, which highlight the position of the end-effector in the next keyframe across all three views. For the remaining action components, it uses an MLP to process the image feature tokens to predict the rotation action, gripper action, and collision flag of the next keyframe.

We assume access to a set of expert demonstrations $\mathcal{D} = \{\tau^i\}_{i=1}^N$ containing N trajectories. And each trajectory contains a language instruction and a sequence of observation-action pairs, *i.e.*, $\tau^i = \{l^i, (\mathbf{o}_1^i, \mathbf{a}_1^i), \dots, (\mathbf{o}_H^i, \mathbf{a}_H^i)\}$. The observation $\mathbf{o}$ is one or multiple RGB-D images captured from one or multiple viewpoints. Following prior works [39, 14, 13], the action $\mathbf{a}$ consists of a 6-DoF end-effector pose $T \in SE(3)$, a target gripper state $g \in \{0, 1\}$, and a collision flag $c \in \{0, 1\}$ of the next key frame. The collision flag c indicates whether the motion planner should avoid collisions while moving towards the target pose. A key frame typically captures important or bottleneck steps in a trajectory (detailed in appendix B.3) [23]. BridgeVLA operates through an iterative process: 1) predicting the action $\mathbf{a}_t$ conditioned on the current observation $\mathbf{o}_t$ and instruction l , 2) moving to the predicted next keyframe pose T_t using a sampling-based motion planner [40, 28, 11], 3) updating observation and repeating until task completion or reaching a maximum step $H_{\max}$.

As illustrated in Fig. 2, BridgeVLA employs a dual-phase training recipe. During pre-training, it is trained to predict 2D heatmaps on object detection datasets. During fine-tuning, point clouds are projected into multiple 2D images as inputs to the VLM backbone. The model is trained to predict 2D heatmaps for estimating the translational action and other action components. **This design aligns the input and output within a shared 2D space in both pre-training and fine-tuning.**

3.2 2D-Heatmap Pre-training

The VLM backbone was originally pre-trained to predict token sequences without spatial structure. To equip it with the same ability to predict heatmaps as downstream policy learning, we introduce a pre-training stage which trains the model to ground target objects via heatmaps. Concretely, we leverage the 120K object detection split of RoboPoint [49] as our pre-training dataset. For each image, we construct the ground-truth heatmap H^{gt} from the bounding boxes of all objects of interest. Specifically, for each object, we construct a probability map with spatial truncation:

$$H_i^{\text{gt}}(\mathbf{x}) = \begin{cases} p_i(\mathbf{x}) & \text{if } p_i(\mathbf{x}) \geq p_{\min} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $\mathbf{x} = (u, v)$ denotes the pixel position, $p_i(\mathbf{x}) = \exp(-\|\mathbf{x} - \hat{\mathbf{x}}_i\|^2/2\sigma^2)$, $\hat{\mathbf{x}}_i$ is the center of the object bounding box, and $p_{\min}$ is a probability threshold. For all the objects of interest, we fuse the

probability map of all objects via averaging and normalization to obtain H^{gt} :

$$H^{\text{gt}}(\mathbf{x}) = \frac{H_{\text{avg}}(\mathbf{x})}{\sum_{\mathbf{x} \in \Omega} H_{\text{avg}}(\mathbf{x})}, \text{ where } H_{\text{avg}}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N H_i^{\text{gt}}(\mathbf{x}) \quad (3)$$

where Ω denotes the pixel space. Please refer to Fig. 9 for samples of the ground-truth heatmaps.

As illustrated in Fig. 2, we input an image along with the text prompt describing the objects of interest into the VLM backbone of BridgeVLA. In this paper, we employ PaliGemma [3] as the VLM backbone, which consists of a SigLIP vision encoder [50] and a Gemma transformer backbone [41]. During its pre-training, PaliGemma takes as input one or multiple 2D images together with a prefix text (*e.g.*, a question about the image) and outputs a suffix text (*e.g.*, an answer to the question). While the model uses causal attention for predicting suffix text tokens, it adopts bidirectional attention for the image tokens and the prefix text tokens. This allows the image tokens to fuse information from the prefix text.

To predict the heatmap, we first rearrange the output image tokens according to their patch positions to reconstruct the spatial feature grid. A convex upsampling block [42] then converts the grid into a heatmap with the same resolution as the input image. Unlike fixed methods (*e.g.*, bilinear or nearest-neighbor), this upsampling module learns pixel-wise interpolation weights, allowing for finer spatial detail recovery. The whole pipeline is trained with a cross-entropy loss to predict heatmaps that localize the position of all objects of interest in the image. We emphasize that the proposed pre-training strategy outputs a spatially aware 2D heatmap, in contrast to the conventional next-token-prediction used in prior works [52, 37]. Moreover, this approach is highly scalable, as it can, in principle, leverage any vision-language datasets that can be formulated as a heatmap prediction tasks, such as keypoint detection and semantic segmentation.

3.3 3D Action Fine-tuning

During fine-tuning, we first reconstruct a point cloud of the scene from the RGB-D images captured from calibrated cameras. To align with the 2D image input of the VLM backbone, we render three orthographic projection images of the point cloud from three viewpoints (top, front, and right) and use these images as the input images for the VLM backbone as in RVT [14] and RVT-2 [15]. These images, along with the task instruction, are then fed into the pre-trained VLM backbone to generate a heatmap for each of the three views. Importantly, we do not incorporate any additional information (*e.g.*, robot states) during the VLM forward pass to minimize the distribution shift between pre-training and fine-tuning.

For translational actions, we back-project the heatmaps of all three views to estimate the scores of all 3D point grids distributed uniformly across the robot workspace. The position of the 3D point with the highest score determines the end-effector’s translation in the next keyframe. Similar to previous works [14, 15], we use Euler angles to represent rotational actions where each axis is discretized into 72 bins. To predict the rotation, binary gripper action, and collision avoidance flag, we integrate features from global and local contexts. For the global feature, max-pooling is applied to the output tokens of each inputted orthographic projection image, resulting in three tokens in total – one for each view. For the local feature, we extract a token from the heatmap peak of each view, also resulting in three tokens in total. All these tokens are concatenated and passed through MLP to predict the rotation action, gripper action, and collision avoidance flag.

Following the approach in prior works [20, 15], BridgeVLA adopts a coarse-to-fine refinement strategy for accurate action prediction. After the initial prediction on the original point cloud, we zoom in and crop the point cloud with a cuboid centered at the predicted translation. A second forward pass is performed on the cropped, zoomed-in point cloud. The predicted action from the second pass is used for execution.

The training loss during fine-tuning consists of four components:

$$L = L_{\text{trans}} + L_{\text{rot}} + L_{\text{gripper}} + L_{\text{collision}} \quad (4)$$

Similar to pre-training, L_{trans} is a cross-entropy loss that supervises the heatmap prediction for translational actions. The ground-truth heatmap for each orthographic view is the normalized single-object probability map defined in Eq. 2, where $\hat{\mathbf{x}}_i$ represents the projected pixel position of the ground-truth end-effector position in the next keyframe. As we discretize the Euler angles for rotation

Overall			Task Success Rate (%)							
Models	Avg. SR (%) $\uparrow$	Avg. Rank $\downarrow$	Close Jar	Drag Stick	Insert Peg	Meat off Grill	Open Drawer	Place Cups	Place Wine	Push Buttons
Image-BC (CNN) [21, 39]	1.3	11.72	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0
Image-BC (ViT) [21, 39]	1.3	12.19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
C2F-ARM-BC [20, 39]	20.1	10.72	24.0	24.0	4.0	20.0	20.0	0.0	8.0	72.0
HiveFormer [16]	45.3	8.47	52.0	76.0	0.0	100.0	52.0	0.0	80.0	84.0
PolarNet [9]	46.4	7.61	36.0	92.0	4.0	100.0	84.0	0.0	40.0	96.0
PerAct [18]	49.4	7.0	55.2 $\pm$ 4.7	89.6 $\pm$ 4.1	5.6 $\pm$ 4.1	70.4 $\pm$ 2.0	88.0 $\pm$ 5.7	2.4 $\pm$ 3.2	44.8 $\pm$ 7.8	92.8 $\pm$ 3.0
Act3D [13]	65.0	4.89	92.0	92.0	27.0	94.0	93.0	3.0	80.0	99.0
RVT [14]	62.9	4.92	52.0 $\pm$ 2.5	99.2 $\pm$ 1.6	11.2 $\pm$ 3.0	88.0 $\pm$ 2.5	71.2 $\pm$ 6.9	4.0 $\pm$ 2.5	91.0 $\pm$ 5.2	100.0$\pm$0.0
3D Diffuser Actor [25]	81.3	2.67	96.0 $\pm$ 2.5	100.0$\pm$0.0	65.6 $\pm$ 4.1	96.8 $\pm$ 1.6	89.6 $\pm$ 4.1	24.0 $\pm$ 7.6	93.6 $\pm$ 4.8	98.4 $\pm$ 2.0
RVT-2 [15]	81.4	2.75	100.0$\pm$0.0	99.0 $\pm$ 1.7	40.0 $\pm$ 0.0	99.0 $\pm$ 1.7	74.0 $\pm$ 11.8	38.0 $\pm$ 4.5	95.0$\pm$3.3	100.0$\pm$0.0
BridgeVLA w/o heat	31.4	10.06	49.3 $\pm$ 2.3	65.3 $\pm$ 2.3	0.0 $\pm$ 0.0	81.3 $\pm$ 4.6	74.7 $\pm$ 10.1	1.3 $\pm$ 2.3	32.0 $\pm$ 14.4	54.7 $\pm$ 6.1
BridgeVLA w pos	56.2	5.97	96.0 $\pm$ 0.0	58.7 $\pm$ 6.1	26.7 $\pm$ 2.3	96.0 $\pm$ 0.0	97.3 $\pm$ 2.3	14.7 $\pm$ 4.6	81.3 $\pm$ 8.3	86.7 $\pm$ 2.3
BridgeVLA	88.2	2.03	100.0$\pm$0.0	100.0$\pm$0.0	88.0$\pm$2.8	100.0$\pm$0.0	100.0$\pm$0.0	58.4$\pm$10.0	88.0 $\pm$ 2.8	98.4 $\pm$ 2.2
Models	Put in Cupboard	Put in Drawer	Put in Safe	Screw Bulb	Slide Block	Sort Shape	Stack Blocks	Stack Cups	Sweep to Dustpan	Turn Tap
Image-BC (CNN) [21, 39]	0.0	8.0	4.0	0.0	0.0	0.0	0.0	0.0	0.0	8.0
Image-BC (ViT) [21, 39]	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	16.0
C2F-ARM-BC [20, 39]	0.0	4.0	12.0	8.0	16.0	8.0	0.0	0.0	0.0	68.0
HiveFormer [16]	32.0	68.0	76.0	8.0	64.0	8.0	8.0	0.0	28.0	80.0
PolarNet [9]	12.0	32.0	84.0	44.0	56.0	12.0	4.0	8.0	52.0	80.0
PerAct [18]	28.0 $\pm$ 4.4	51.2 $\pm$ 4.7	84.0 $\pm$ 3.6	17.6 $\pm$ 2.0	74.0 $\pm$ 13.0	16.8 $\pm$ 4.7	26.4 $\pm$ 3.2	2.4 $\pm$ 2.0	52.0 $\pm$ 0.0	88.0 $\pm$ 4.4
Act3D [13]	51.0	90.0	95.0	47.0	93.0	8.0	12.0	9.0	92.0	94.0
RVT [14]	49.6 $\pm$ 3.2	88.0 $\pm$ 5.7	91.2 $\pm$ 3.0	48.0 $\pm$ 5.7	81.6 $\pm$ 5.4	36.0 $\pm$ 2.5	28.8 $\pm$ 3.9	26.4 $\pm$ 8.2	72.0 $\pm$ 0.0	93.6 $\pm$ 4.1
3D Diffuser Actor [25]	85.6$\pm$4.1	96.0 $\pm$ 3.6	97.6 $\pm$ 2.0	82.4 $\pm$ 2.0	97.6$\pm$3.2	44.0 $\pm$ 4.4	68.3 $\pm$ 3.3	47.2 $\pm$ 8.5	84.0 $\pm$ 4.4	99.2$\pm$1.6
RVT-2 [15]	66.0 $\pm$ 4.5	96.0 $\pm$ 0.0	96.0 $\pm$ 2.8	88.0$\pm$4.9	92.0 $\pm$ 2.8	35.0 $\pm$ 7.1	80.0$\pm$2.8	69.0 $\pm$ 5.9	100.0$\pm$0.0	99.0 $\pm$ 1.7
BridgeVLA w/o heat	5.3 $\pm$ 2.3	0.0 $\pm$ 0.0	58.7 $\pm$ 22.7	2.7 $\pm$ 2.3	64.0 $\pm$ 0.0	4.0 $\pm$ 4.0	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	32.0 $\pm$ 4.0	40.0 $\pm$ 10.6
BridgeVLA w pos	10.7 $\pm$ 2.3	78.7 $\pm$ 2.3	97.3 $\pm$ 4.6	16.0 $\pm$ 4.0	72.0 $\pm$ 0.0	21.3 $\pm$ 8.3	17.3 $\pm$ 2.3	4.0 $\pm$ 4.0	53.3 $\pm$ 2.3	84.0 $\pm$ 0.0
BridgeVLA	73.6 $\pm$ 4.6	99.2$\pm$1.8	99.2$\pm$1.8	87.2 $\pm$ 6.6	96.0 $\pm$ 2.8	60.8$\pm$7.7	76.8 $\pm$ 8.7	81.6$\pm$3.6	87.2 $\pm$ 1.8	92.8 $\pm$ 3.3

Table 1: **Results on RL Bench.** The "Avg. Rank" column reports the average rank of each method across all 18 tasks, where lower values indicate better overall performance. "BridgeVLA w/o heat" refers to the ablated version that directly predicts actions without using intermediate heatmaps. "BridgeVLA w pos" refers to the ablated version that incorporates position features into the image features. BridgeVLA achieves the best performance in 10 out of the 18 tasks.

into bins, we also apply cross-entropy loss in L_{rot} to supervise rotation prediction. For gripper action and collision avoidance, we use the binary cross-entropy loss in L_{gripper} and $L_{\text{collision}}$ as supervision. To enhance geometric robustness, random rigid-body transformations are applied jointly to the point cloud and the ground-truth action during training. Additional training details can be found in Appendix A.

4 Experiments

In this section, we perform extensive experiments in both simulation and the real world to evaluate the proposed method. Through the experiments, we aim to answer five questions:

- Q1: How effectively does BridgeVLA learn 3D robot manipulation compared to state-of-the-art methods when sufficient data is available?
- Q2: Does BridgeVLA learn more efficiently than existing state-of-the-art methods when data is limited (e.g., 3 trajectories per task)?
- Q3: How robust is BridgeVLA in handling visual disturbances (e.g., distractors, background, and lighting)?
- Q4: How well does BridgeVLA generalize to novel object-skill combinations and objects from previously unseen categories?
- Q5: Are our architectural designs (e.g., predicting heatmaps before outputting actions) truly useful when constructing 3D VLA?

4.1 Simulation Experiments

4.1.1 Experiments on RL Bench

Setup. RL Bench [19] implements tasks in CoppeliaSim [38] using a Franka Panda robot mounted with a parallel-jaw gripper. The observation contains four RGB-D images captured from four calibrated cameras positioned at the front, left shoulder, right shoulder, and wrist. Following previous works [39, 13, 14, 25, 15], we perform experiments on 18 tasks from RL Bench. These tasks span 1)

non-prehensile manipulation (*e.g.*, *slide block to target*), 2) pick-and-place (*e.g.*, *stack cups*), and 3) high-precision insertion (*e.g.*, *insert peg*). Each task is provided with 100 expert demonstrations. And each demonstration is paired with language instruction and multiple keyframes. Models are evaluated via binary success rates over 25 trials per task, with a maximum of 25 action steps per trial.

Baselines. We compare BridgeVLA with multiple baselines. (1) **Image-BC (CNN)** and **Image-BC (ViT)** [21] are two 2D baseline methods which predict the actions directly from 2D images using CNN and ViT as the backbone, respectively. (2) **C2F-ARM-BC** [20] predicts the next keyframe action in the voxel space with a coarse-to-fine strategy. (3) **PerAct** [39] also operates in the voxel space and predicts the action with a perceiver transformer [18]. (4) **HiveFormer** incorporates historical information using a unified multi-modal transformer architecture. (5) **PolarNet** employs PointNext [36] to encode the 3D scene and predicts both heatmaps and offsets for all points to estimate translational actions. (6) **Act3D** [13] predicts the next keyframe action by selecting the point with the highest score from a set of randomly sampled points in the workspace. (7) **3D Diffuser Actor** [25] generates 3D trajectories via a diffusion process conditioned on 3D observation and language instructions. (8) **RVT** [14] uses multi-view transformer to aggregate information from multiple orthographic views of the point cloud observation. (9) And **RVT-2** [15], the current state-of-the-art method, further improves the precision of its prior via a coarse-to-fine strategy.

Results. In total, we evaluate BridgeVLA five times to minimize statistical bias. The results are shown in Table 1. BridgeVLA outperforms all the comparing baseline methods, achieving an average success rate of 88.2% and an average rank of 1.9 across all the 18 tasks, establishing a new state of the art in this benchmark. These results address Q1, demonstrating the effectiveness of BridgeVLA in learning complex 3D manipulation tasks. We highlight that BridgeVLA outperforms the best baseline method by a large margin in *Insert Peg* (88.0% vs 40.0%) and *Sort Shape* (60.8% vs 35.0%). These two tasks demand highly precise alignment between the peg and hole and the block and sorter, respectively. The high success rates of our method showcase its strong capabilities of learning precise manipulation which is highly desirable in many industrial applications. Among the 18 tasks, BridgeVLA performs the worst in *Place Cups*, despite surpassing all the comparing baseline methods. We hypothesize this is because the target keypoints are often occluded in all orthographic projection views, which makes learning and prediction more challenging. In the future, we plan to explore dynamically selecting the projection views for rendering to avoid this problem.

4.1.2 Experiments on COLOSSEUM & GemBench

To further evaluate the generalization capabilities of BridgeVLA, we conduct experiments on the COLOSSEUM benchmark [35] and GemBench [12]. These two benchmarks extend RL Bench. The COLOSSEUM benchmark evaluates models in environments with 12 axes of perturbations, which were not seen during training. These perturbations include variations in object texture, color, size, background, lighting, distractors, and camera poses. As such, this benchmark is used to assess Q3.

GemBench is a hierarchical generalization benchmark. Its training set consists of 16 tasks (31 variations) covering seven core action primitives: press, pick, push, screw, close, open, and stack/put. The test set includes 44 tasks (92 variations), categorized into four increasingly challenging settings. These settings incorporate novel object-skill combinations and new object categories, making it suitable for evaluating Q4.

BridgeVLA outperforms all existing state-of-the-art 3D manipulation methods on both benchmarks, addressing Q3 and Q4. Due to space limitations, the details of the environment setup, baselines, and analysis can be found in Appendix B.1 and Appendix B.2.

4.2 Real-Robot Experiments

Setup. In this section, we perform real-robot experiments to validate the effectiveness of BridgeVLA in the real world. Our real-robot setup includes a Franka Research 3 robot arm mounted with a parallel-jaw gripper (Fig. 3). A static ZED 2i depth camera is used to provide the colored point cloud observation. In total, we evaluate on 13 tasks (see Tab. 12 for a full list of tasks). These tasks range from simple pick-and-place to complex long-horizon tasks, requiring the robot to open a drawer and put items into the drawer. Each task contains 3-9 keyframes (see Fig. 7 and 8 for visualization). For each task, we collect 10 expert trajectories for training.

Method	Put the soda can in the bottom shelf	Put the giraffe in the lower drawer	Place the red block in the blue plate	Press Sanitizer	Put the RedBull can in the top shelf	Put the RedBull can in the bottom shelf	Put the coke can in the top shelf
SpatialVLA(50) [37]	1/10	1/10	5/10	6/10	3/10	1/10	2/10
SpatialVLA(10) [37]	0/10	0/10	0/10	2/10	0/10	0/10	0/10
π_0 [5]	0/10	0/10	2/10	1/10	0/10	1/10	0/10
ACT [51]	2/10	2/10	3/10	2/10	3/10	1/10	2/10
RVT-2 [15]	10/10	8/10	8/10	10/10	9/10	10/10	10/10
BridgeVLA	9/10	9/10	10/10	10/10	10/10	10/10	10/10

Method	Place the orange block in the green plate	Place the red block in the purple plate	Place the yellow block in the green plate	Put the zebra in the upper drawer	Put the zebra in the lower drawer	Put the wolf in the upper drawer	Average
SpatialVLA(50) [37]	6/10	3/10	5/10	2/10	0/10	2/10	28.5%
SpatialVLA(10) [37]	1/10	1/10	0/10	0/10	0/10	0/10	3.1%
π_0 [5]	0/10	0/10	1/10	0/10	0/10	0/10	3.8%
ACT [51]	2/10	3/10	4/10	1/10	2/10	1/10	22.3%
RVT-2 [15]	10/10	9/10	9/10	7/10	8/10	9/10	90%
BridgeVLA	10/10	10/10	10/10	9/10	10/10	9/10	96.9%

Table 2: **Per-task Success Rate in the Basic Setting.** Except for SpatialVLA(50), which was trained with 50 trajectories, all other methods were trained with 10 trajectories. BridgeVLA outperforms all baseline methods, achieving an almost perfect success rate of 96.9%.

In total, we design 7 different settings to comprehensively evaluate our model’s performance. (1) **Basic:** The model is evaluated in environments that are similar to the training data. (2) **Distractor:** Distractor objects that are visually similar to at least one target object are added to the scene. (3) **Lighting:** The model is tested in a visually distinct lighting condition in which the lights are turned off. (4) **Background:** Three different tablecloths are used to change the background. (5) **Height:** All objects for manipulation are placed on a drawer that is 9.5cm high. (6) **Combination:** We combine objects and skills that are not paired together in the training datasets. That is, while the objects (*e.g.*, red block and green plate) and skill (*e.g.*, place A in B) are seen during training, the instruction that pairs them together is novel (*e.g.*, place the red block in the green plate). In total, we evaluate 13 novel object-skill combinations (Fig. 11 and 12). (7) **Category:** To test whether BridgeVLA is able to transfer the broad knowledge from pre-training to downstream policy learning, we evaluate on manipulating objects from categories that are *unseen* in the robot training data. In total, we test 7 novel objects (Fig. 13). Distractor, Lighting, Background, and Height aim to evaluate the robustness against visual disturbances, while Combination and Category evaluate the generalization capabilities for unseen instructions.

To demonstrate BridgeVLA’s advantages over existing manipulation policy, we compare it with four types of representative methods:

- 1) **SpatialVLA** [37]: A state-of-the-art **3D VLA** model that incorporates 3D information through Ego3D positional encoding and leverages Adaptive Action Grids to accelerate inference.
- 2) π_0 [5]: A state-of-the-art **2D VLA** model pretrained on a large-scale cross-embodiment dataset. It adopts a vision-language model (VLM) backbone and employs a flow matching action expert to generate final actions.
- 3) **ACT** [51]: A state-of-the-art **2D non-VLA** model using a Conditional Variational Autoencoder (CVAE) to model action distributions. Though effective for fine-grained manipulation, ACT does not support language conditioning, so we train a separate single-task model for each task, which should theoretically perform better than a multi-task version.
- 4) **RVT-2** [15]: A state-of-the-art **3D non-VLA** model performing the best in our simulation experiments. (See Sec. 4.1)

Results. We first compare BridgeVLA with these baselines on the basic setting. For every task, we evaluated every baseline over 10 trials to ensure statistical robustness. For fair comparison, we photographed each test scene and manually aligned the scenes across all methods. Results are provided in Tab. 2. As we can see, most methods completely fails when given only 10 trajectories per task except two 3D related methods: RVT-2 and BridgeVLA. Notably, although SpatialVLA also utilises 3D information, its data efficiency is still very low. Even when the data are increased to 50 trajectories per task, its success rate is still much lower than BridgeVLA, which indicates only adding 3D information is not enough for constructing 3D VLA model and a carefully designed network architecture is still very important. To assess the data efficiency of BridgeVLA, we also train the model with only 3 trajectories per task. Remarkably, despite the limited data, BridgeVLA achieves a success rate of 95.4% in Basic, matching the performance achieved with 10 trajectories per task. This result underscores the data efficiency of the proposed method, directly addressing Q2. Detailed

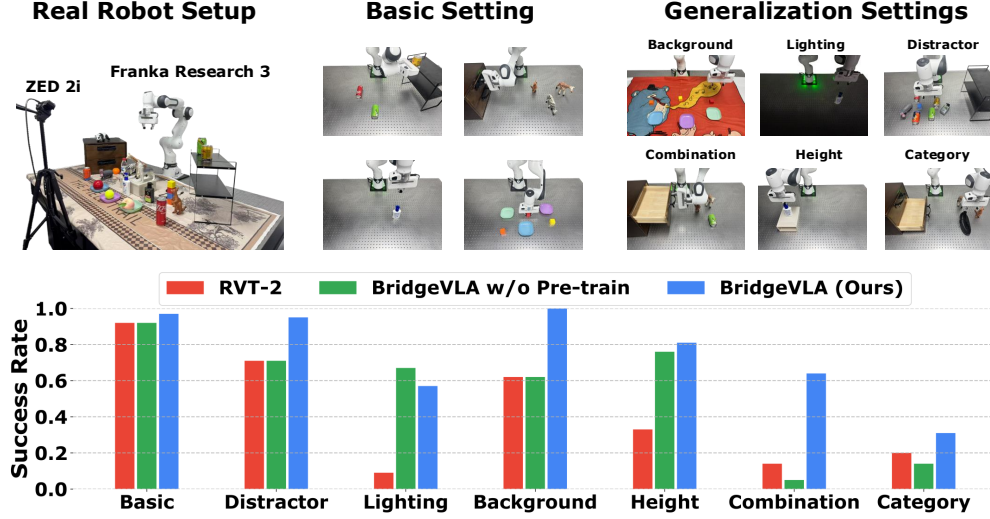


Figure 3: **Real-Robot Experiments and Results.** We use a Franka Research 3 robot arm and a ZED 2i camera to capture point clouds of the scene. To evaluate the model’s performance, we design 7 different settings including one basic setting and six generalization settings. Experimental results show that BridgeVLA outperforms the state-of-the-art baseline method RVT-2 [15] by an average of 32%.

per-task results are provided in Appendix C.5 and observations about the baselines are detailed in Appendix C.2.

Considering that only RVT-2 and BridgeVLA have a good performance in basic setting, we only further evaluate these two methods on other generalization settings. The average success rates are shown in Fig. 3. BridgeVLA outperforms RVT-2 in all the seven settings. As we can see, RVT-2 struggles in both visual generalization settings and semantic generalization settings, while BridgeVLA performs much better, especially in Lighting and Combination. These results addresses Q3 and Q4, indicating that BridgeVLA is able to handle visual disturbance and novel instructions robustly.

Although our method outperforms baseline methods in the Category setting, its absolute success rate is not high. A common failure mode is that the robot often ignores the target object and moves directly to the destination during pick-and-place manipulation. We believe this relatively low performance is *not* due to BridgeVLA forgetting the knowledge gained from pre-training, as it still predicts heatmaps accurately when provided with samples from the pre-training dataset after fine-tuning (see Fig. 4 and Appendix C.4). Instead, we hypothesize that the reduced performance stems from two factors: 1) The images in the pre-training dataset are mostly captured from third-person views, which differ significantly from the projection images in our robot data; 2) The pre-training task focuses solely on object localization, whereas manipulation involves predicting keypoints that do not correspond to an object. To address these issues, we plan to expand both the scale and diversity of the pre-training dataset and explore more expressive action-decoding methods to better leverage the preserved pre-training knowledge.

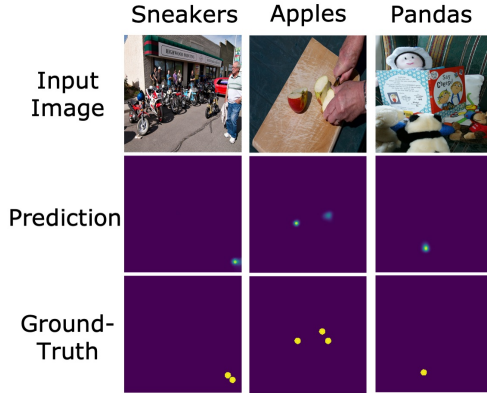


Figure 4: **Prediction on Pre-training Data after Fine-tuning.** To simulate the multi-view inputs during fine-tuning, we repeat each pre-training image three times and feed them into the fine-tuned model to generate heatmaps. Note that these samples are *not* cherry-picked. Additional samples can be found in Appendix C.4.

4.3 Ablation Studies

To prove the effectiveness of our model design and provide insights for the community, we conduct three ablation studies:

Whether we need to predict heatmaps before predicting actions. Our approach avoids direct action prediction by first generating 2D heatmaps using a convex upsampling module. Target positions are then computed by projecting 3D workspace points onto the heatmaps and selecting the point with the highest mean probability. For ablation, we replaced the convex upsampling module (309M parameters) with a similarly sized Transformer decoder (303M) to directly predict target positions, supervised by MSE loss. All other modules were kept the same as before. We performed a hyperparameter grid search and evaluated the model on RL Bench. Results are shown in the Tab. 1. Replacing heatmap prediction with direct position regression reduced the average success rate from 88.2% to 31.4%, confirming the effectiveness of our heatmap-based design. The ablated model was also harder to train and more sensitive to hyperparameters—requiring a batch size of 192 and careful learning rate tuning—while our original model trains reliably even with a batch size of 64. We see three main reasons for this outcome: (1) Heatmaps offer denser supervision than 3D position vectors, enabling more effective learning. (2) Projecting 3D points onto heatmaps introduces helpful spatial priors, easing the learning process. (3) The 2D heatmaps share the same spatial structure as the input images, enhancing alignment and improving performance.

Whether we need to remove the 3D position input to the VLM backbone. Unlike typical 3D VLA models like SpatialVLA, we deliberately avoid using per-pixel 3D position inputs and rely solely on RGB images. This design preserves alignment between the input feature spaces of fine-tuning and VLM pretraining, which we find crucial for effective vision-language-action (VLA) modeling. To test this, we added a 3D convolutional module to encode per-pixel 3D positions, fused them with 2D features, and fed the result into the backbone. Although this adds richer spatial cues, it alters the image feature distribution seen during pretraining, leading to a performance drop from 88.2% to 56.2% on RL Bench. Detailed results are shown in the Tab. 1.

Whether we need to do 2D heatmap pre-training to the VLM backbone. The original VLM backbone can not predict heatmaps, while our downstream policy learning requires such ability. To bridge the gap, we do 2D heatmap pre-training to the VLM backbone. To verify its effectiveness, we ablate this pre-training and evaluate model’s performance in the real world, the results are shown in Fig. 3. As we can see, BridgeVLA w/o Pre-train is not able to generalize well in both language-related generalization settings and can not even beat RVT-2, while BridgeVLA achieves the best performance across the two generalization settings especially in Combination, highlighting its ability to understand language semantics. We hypothesize that the 2D-heatmap pre-training equips BridgeVLA with the ability to connect the semantics in language instructions with image observations in the heatmap space. All the above experiment results address Q5 and highlight the effectiveness of our architectural designs.

5 Conclusions & Future Work

This paper has introduced BridgeVLA, a novel and efficient 3D vision-language-action (VLA) model built on top of a pre-trained vision-language model (VLM) [3]. Keys to our method are that (1) it converts 3D inputs to 2D images to align with the 2D image inputs of the pre-trained VLM; (2) it aligns the input observation and the output action to a unified 2D image space via 2D heatmap prediction; (3) it adopts a scalable pre-training method to equip the VLM with the capability to predict heatmaps before fine-tuning on action prediction. Extensive experiments show that the proposed method is able to learn 3D manipulation efficiently and effectively in both simulation and the real world. In the future, we plan to explore pre-training on more diverse tasks, including semantic segmentation and keypoint detection. We also want to incorporate more expressive action-decoding methods (*e.g.*, diffusion [10]) into the framework to continue improving the policy performance.

6 Acknowledgments

This work was jointly supported by National Natural Science Foundation of China (62322607, 62236010 and 62276261), Beijing Natural Science Foundation (L252033), FiveAges Grant, and Youth Innovation Promotion Association of Chinese Academy of Sciences under Grant 2021128.

References

- [1] AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.
- [2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibor Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- [3] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.
- [4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0 : A vision-language-action flow model for general robot control, 2024. URL <https://arxiv.org/abs/2410.24164>.
- [5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.
- [7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Chormanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.
- [8] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.
- [9] Shizhe Chen, Ricardo Garcia Pinel, Cordelia Schmid, and Ivan Laptev. Polarnet: 3d point clouds for language-guided robotic manipulation. In *Conference on Robot Learning*, pages 1761–1781. PMLR, 2023.
- [10] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.
- [11] David Coleman, Ioan Sucan, Sachin Chitta, and Nikolaus Correll. Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*, 2014.
- [12] Ricardo Garcia, Shizhe Chen, and Cordelia Schmid. Towards generalizable vision-language robotic manipulation: A benchmark and llm-guided 3d policy. *arXiv preprint arXiv:2410.01345*, 2024.
- [13] Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *Conference on Robot Learning*, pages 3949–3965. PMLR, 2023.
- [14] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.
- [15] Ankit Goyal, Valts Blukis, Jie Xu, Yijie Guo, Yu-Wei Chao, and Dieter Fox. Rvt-2: Learning precise manipulation from few demonstrations. In *RSS 2024 Workshop: Data Generation for Robotics*, 2024.
- [16] Pierre-Louis Guhur, Shizhe Chen, Ricardo Garcia Pinel, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. Instruction-driven history-aware policies for robotic manipulations. In *Conference on Robot Learning*, pages 175–187. PMLR, 2023.

- [17] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi 0.5$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.
- [18] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations*.
- [19] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020.
- [20] Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13739–13748, 2022.
- [21] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.
- [22] Yueru Jia, Jiaming Liu, Sixiang Chen, Chenyang Gu, Zhilue Wang, Longzan Luo, Lily Lee, Pengwei Wang, Zhongyuan Wang, Renrui Zhang, et al. Lift3d foundation policy: Lifting 2d large-scale pretrained models for robust 3d robotic manipulation. *arXiv preprint arXiv:2411.18623*, 2024.
- [23] Edward Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 4613–4619. IEEE, 2021.
- [24] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.
- [25] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In *8th Annual Conference on Robot Learning*, 2024.
- [26] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In *8th Annual Conference on Robot Learning*.
- [27] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.
- [28] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, 2000.
- [29] Chengmeng Li, Junjie Wen, Yan Peng, Yaxin Peng, Feifei Feng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *arXiv preprint arXiv:2503.07511*, 2025.
- [30] Peiyan Li, Hongtao Wu, Yan Huang, Chilam Cheang, Liang Wang, and Tao Kong. Gr-mg: Leveraging partially-annotated data via multi-modal goal-conditioned policy. *IEEE Robotics and Automation Letters*, 2025.
- [31] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023.

- [32] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. *Advances in Neural Information Processing Systems*, 36:72983–73007, 2023.
- [33] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.
- [34] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- [35] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024.
- [36] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022.
- [37] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.
- [38] Eric Rohmer, Surya PN Singh, and Marc Freese. V-rep: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, pages 1321–1326. IEEE, 2013.
- [39] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.
- [40] Ioan A. Șucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. doi: 10.1109/MRA.2012.2205651. <https://ompl.kavrakilab.org>.
- [41] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.
- [42] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020.
- [43] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.
- [44] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, 2024.
- [45] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024.
- [46] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.
- [47] Rujia Yang, Geng Chen, Chuan Wen, and Yang Gao. Fp3: A 3d foundation policy for robotic manipulation. *arXiv preprint arXiv:2503.08950*, 2025.
- [48] Wentao Yuan, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. M2t2: Multi-task masked transformer for object-centric pick and place. *arXiv preprint arXiv:2311.00926*, 2023.

- [49] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.
- [50] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.
- [51] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
- [52] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

NeurIPS Paper Checklist

1. Claims

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

Answer: [\[Yes\]](#)

Justification: Please refer to **Abstract** and Sec. 1

Guidelines:

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

2. Limitations

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

Answer: [\[Yes\]](#)

Justification: We discuss our limitations in Sec. 4.2 and Sec. 5

Guidelines:

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

3. Theory assumptions and proofs

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

Answer: [\[NA\]](#)

Justification: All results in this paper are supported by experiments and we do not include any theoretical results.

Guidelines:

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

4. Experimental result reproducibility

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

Answer: [\[Yes\]](#)

Justification: We have showed the main information of our method and experiment settings in the main paper. Meanwhile, we also display details in the Appendix.

Guidelines:

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

5. Open access to data and code

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

Answer: [Yes]

Justification: We will opensource the code, data and checkpoints upon acceptance.

Guidelines:

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

6. Experimental setting/details

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

Answer: [Yes]

Justification: We have specified these details in Sec. 3, Sec. 4, Sec. A, Sec. B and Sec. C.

Guidelines:

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

7. Experiment statistical significance

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

Answer: [Yes]

Justification: We report both the success rate and variance in Tab. 1, Tab. 6 and Tab. 7

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

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

Answer: [Yes]

Justification: We provide these information in Sec. A

Guidelines:

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

9. Code of ethics

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

Answer: [Yes]

Justification: We did not identify any aspects that violate the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

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

Answer: [NA]

Justification: Our work introduces a novel 3D VLA framework aimed at advancing the AI and robotics community. We do not identify any direct societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

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

Answer: [NA]

Justification: We do not find such risks in this project.

Guidelines:

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

12. Licenses for existing assets

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

Answer: [Yes]

Justification: For the code, data and other assets we use in this paper, we have cited original paper and respected their terms.

Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

13. **New assets**

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

Answer: [NA]

Justification: This paper does not release new assets.

Guidelines:

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

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

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

Answer: [NA]

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

Guidelines:

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

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

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

Answer: [NA]

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

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

Appendix

Table 3: Training hyperparameters for BridgeVLA

	Pretrain	RLBench Finetune	Colosseum Finetune	Real-robot Finetune
learning rate	5e-5	8e-5	8e-5	2e-5
optimizer	AdamW	AdamW	AdamW	AdamW
batch size	384	192	192	192
warmup steps	400	-	-	-

A Training & Inference Details

Detailed training configurations are summarized in Tab. 3. Throughout both pre-training and fine-tuning, we keep the SigLIP vision encoder and language token embeddings frozen.

Computational Resources:

1. Pre-training: 8 NVIDIA A100 GPUs for 3,800 steps (≈ 2 hours)
2. RLBench fine-tuning: 48 NVIDIA H100 GPUs for 83,000 steps (≈ 20 hours)
3. COLOSSEUM fine-tuning: 48 NVIDIA H100 GPUs for 83,000 steps (≈ 20 hours)
4. GemBench fine-tuning: 40 NVIDIA A100 GPUs for 50 epochs (≈ 2.1 hours)
5. Real-world fine-tuning: 8 NVIDIA A100 GPUs for 300 epochs (≈ 1.5 hours)

For inference, we run BridgeVLA on a machine equipped with an NVIDIA RTX 4090 GPU. To evaluate its inference speed, we conducted 100 trials. From point cloud input to action output, the average end-to-end inference time is 0.21 seconds.

B Simulation Experiments

B.1 Experiments on COLOSSEUM

Setup. The COLOSSEUM benchmark is an extension to the RLBench benchmark. The model is trained on the data from the original RLBench benchmark but evaluated in environments spanning 12 axes of perturbations. These perturbations, which are unseen during training, encompass changes in object texture, color, and size, backgrounds, lighting, distractors and camera poses. In total, the COLOSSEUM creates 20,371 unique task perturbations instances to comprehensively evaluate the generalization capabilities of the model. Specifically, our evaluation includes three steps: 1) train the model with the original RLBench data without perturbations (100 trajectories per task) on 20 tasks, 2) evaluate each task over 25 trials per perturbation, 3) compute the average success rate of all evaluated tasks for every perturbation. Besides the 12 types of perturbations, we also evaluate on basic variations from the original RLBench (denoted as **RLBench** in Tab. 4), and a more challenging setting which combines all the 12 types of perturbations (denoted as **All Perturbations** in Tab. 4).

Baselines. We compare BridgeVLA with five baseline methods. **R3M-MLP** and **MVP-MLP** are two 2D methods that utilize pre-trained visual encoders to process observation images and an MLP for action prediction. Specifically, R3M-MLP uses R3M [33] that is pre-trained on large-scale egocentric human videos; MVP-MLP uses MVP [46] that is pre-trained on millions of in-the-wild data. Both visual encoders show strong adaptability on various robotics tasks in both simulation and the real world. We also compare with three 3D methods introduced in Sec. 4.1.1, *i.e.*, **PerAct** [39], **RVT** [14], and **RVT-2** [15].

Overall			Success Rate (%)					
Models	Avg. SR (%) $\uparrow$	Avg. Rank $\downarrow$	All Perturbations	MO-COLOR	RO-COLOR	MO-TEXTURE	RO-TEXTURE	MO-SIZE
R3M-MLP[33]	0.8	5.71	0.6	0.4	0.0	0.0	0.0	1.8
MVP-MLP[46]	1.6	5.0	0.8	1.2	0.0	0.4	0.0	4.44
PerAct[18]	27.9	3.71	7.2	24.0	29.2	28.8	17.71	35.6
RVT[14]	35.4	3.28	6.4	26.0	31.3	44.8	41.1	35.3
RVT-2[15]	56.7	1.92	15.6 $\pm$ 0.8	53.0 $\pm$ 0.9	54.6 $\pm$ 0.6	59.7 $\pm$ 0.7	56.7 $\pm$ 1.4	60.9 $\pm$ 0.9
BridgeVLA (Ours)	64.0	1.07	18.7 $\pm$ 2.2	60.5 $\pm$ 1.1	63.8 $\pm$ 0.1	63.5 $\pm$ 1.5	68.4 $\pm$ 3.3	69.3 $\pm$ 1.0
Models	RO-SIZE	Light Color	Table Color	Table Texture	Distractor	Background Texture	RLBench	Camera Pose
R3M-MLP[33]	0.0	1.0	1.4	0.2	1.6	1.2	2.0	0.8
MVP-MLP[46]	0.0	1.6	1.6	1.0	3.8	2.2	2.0	2.6
PerAct[18]	29.3	29.1	30.4	23.2	27.1	33.5	39.4	36.3
RVT[14]	40.5	34.0	30.0	45.2	18.8	46.4	53.4	42.2
RVT-2[15]	53.4 $\pm$ 1.5	58.0 $\pm$ 1.1	62.6 $\pm$ 0.9	56.6 $\pm$ 0.9	60.8 $\pm$ 0.5	68.7 $\pm$ 1.1	68.8 $\pm$ 1.3	64.4 $\pm$ 0.5
BridgeVLA (Ours)	61.7 $\pm$ 0.8	69.7 $\pm$ 1.2	75.7 $\pm$ 0.9	71.3 $\pm$ 0.7	51.8 $\pm$ 1.5	74.8 $\pm$ 1.0	73.1 $\pm$ 0.2	73.8 $\pm$ 0.3

Table 4: **Results on the COLOSSEUM Benchmark.** The table shows the success rates across 14 generalization settings. The ‘‘Avg. Rank’’ column reports the average rank of each method across all perturbations, where lower values indicate better overall performance. Compared to the state-of-the-art baseline, BridgeVLA improves the average success rate by 7.3%.

Results. Results are shown in Tab. 4. We use the results of R3M-MLP [33], MVP-MLP [46], RVT [14], and PerAct [39] from the original COLOSSEUM paper [35]. For RVT-2 [15] and BridgeVLA, we perform our own training and evaluation process. We performed three test repetitions and report the average success rate and variance of BridgeVLA and RVT-2 for each task under different perturbations in Tab.6 and Tab.7, respectively. BridgeVLA outperforms all the comparing baseline methods in terms of average success rate, significantly outperforming the best baseline method by 7.3%. Among all the 14 evaluated perturbations, our method ranks the best among all methods in 13 of them. These results address Q3, showcasing that BridgeVLA possesses strong robustness against visual perturbation.

B.2 Experiments on GemBench

Setup. GemBench [12] is a hierarchical generalization benchmark built on the RLBench simulator [19]. Its training set contains 16 tasks (31 variations) covering seven core action primitives—press, pick, push, screw, close, open, and stack/put. The test set consists of 44 tasks (92 variations), categorized into four increasingly challenging settings:

L1 (Novel Placements): L1 consists of the original 16 tasks (31 variations). The object placements are randomized within the workspace. In addition, chromatic distractors are introduced to test the ability to handle additional visual complexity.

L2 (Novel Rigid Objects): L2 involves 15 unseen tasks (28 variations) that require interaction with 8 novel rigid objects using learned primitives. The generalization capabilities are evaluated across two categories: novel object-color compositions and novel object shapes.

L3 (Novel Articulated Objects): L3 consists of 18 unseen tasks (21 variations) that involve interacting with articulated objects. It evaluates the generalization capabilities across three categories: novel action-part compositions, novel object instances, and novel object categories.

L4 (Novel Long-Horizon Tasks): L4 includes 6 complex long-horizon tasks (12 variations) that require combining multiple actions to finish a whole task.

Baselines. In total, we compare with six baseline methods. **3D-LOTUS** [12] processes point cloud inputs through a language-conditioned point cloud transformer architecture [45]. It showcases notable multi-tasking capabilities and high training efficiency. Its enhanced variant, **3D-LOTUS++** [12], integrates the generalization capabilities of large-scale models into 3D-LOTUS with a modular architecture consisting of three components: (1) LLM-based task planning [1], (2) VLM-based object grounding [32, 27], and (3) motion control inherited from 3D-LOTUS. We also compare with four methods introduced in Sec. 4.1.1, *i.e.*, **Hiveformer** [16], **PolarNet** [9], **3D Diffuser Actor** [25], **RVT-2** [15]

Results. Overall results are shown in Tab. 5 and per-task success rates on the four settings of GemBench are shown in Tab. 8, 9, 10, 11. The results of baseline methods are sourced from [12]. In total, we evaluate on 5 random seeds to reduce statistical variance. And for every seed, we

Method	Average	L1	L2	L3	L4
Hiveformer [16]	30.4	60.3 ± 1.5	26.1 ± 1.4	35.1 ± 1.7	0.0 ± 0.0
PolarNet [9]	38.4	77.7 ± 0.9	37.1 ± 1.4	38.5 ± 1.7	0.1 ± 0.2
3D Diffuser Actor [25]	44.0	91.9 ± 0.8	43.4 ± 2.8	37.0 ± 2.2	0.0 ± 0.0
RVT-2 [15]	44.0	89.1 ± 0.8	51.0 ± 2.3	36.0 ± 2.2	0.0 ± 0.0
3D-LOTUS [12]	45.7	94.3 ± 1.4	49.9 ± 2.2	38.1 ± 1.1	0.3 ± 0.3
3D-LOTUS++ [12]	48.0	68.7 ± 0.6	64.5 ± 0.9	41.5 ± 1.8	17.4 ± 0.4
BridgeVLA (Ours)	50.0	91.1 ± 1.1	65.0 ± 1.3	43.8 ± 1.2	0.0 ± 0.0

Table 5: **Results on GemBench.** We show the average success rates on the four evaluation settings of GemBench. BridgeVLA establishes a new state of the art on this benchmark, achieving an average success rate of 50.0%.

run 20 trials per task variation. BridgeVLA consistently outperforms all the comparing baseline methods in terms of average success rate across the four evaluation settings. Notably, BridgeVLA achieves state-of-the-art results in both the L2 and L3 settings, demonstrating strong generalization capabilities, addressing Q4. However, similar to most baseline approaches, BridgeVLA exhibits limited performance in the L4 setting, where each task comprises multiple sub-tasks. In the future, we plan to explore leveraging large language models (LLMs) for long-horizon task decomposition and further improve the performance in such setting.

B.3 Key frame Selection

For all the simulation and real-robot experiments, we adopt the same key frame selection strategy as PerAct [39]. A time step is labeled as a key frame if (i) the robot is stationary, (ii) the gripper state changes, or (iii) the step is the final state of the episode. The robot is considered stationary when the absolute velocities of all joints fall below 0.1 rad/s.

B.4 Data

Following [39, 14, 15], we select 18 tasks from RLBench [19] to evaluate the performance of our method on complex manipulation tasks. These tasks are visualized in Fig. 5.

To assess the generalization capability of BridgeVLA, we also evaluate on the COLOSSEUM benchmark [35] and GemBench [12]. The COLOSSEUM benchmark includes 20 basic tasks and 12 types of perturbations. These perturbations, which are unseen during training, encompass changes in object texture, color, and size, backgrounds, lighting, distractors and camera poses. The benchmark evaluates on all the 12 types of perturbations, a setting with basic variations from the original RLBench, and a more challenging setting which combines all the 12 types of perturbations. We visualize all perturbations except the one from the original RLBench in Fig. 6.

For GemBench, the training set includes 16 tasks (31 variations) spanning seven fundamental action primitives (press, pick, push, screw, close, open, stack/put). The test set includes 44 tasks (92 variations) organized into four increasingly challenging settings. Unlike RLBench and COLOSSEUM, where demo augmentation is used, we train BridgeVLA using only keyframes from each trajectory without performing any demo augmentation in GemBench.

C Real-Robot Experiments

C.1 Experiment Setup

Fig. 3 illustrates our real-robot setting. The platform comprises a 7-DoF Franka Research 3 manipulator with a parallel-jaw gripper and a ZED 2i stereo camera mounted on a tripod for capturing point clouds of the workspace. We collect expert trajectories with a kinesthetic teaching approach. We first move the manipulator to keypoints of an expert trajectory and then play back the keypoints to record the observation and action at each keypoint.

C.2 Basic Setting

This setting provides a scene similar to the training dataset, where only the object layouts are modified. To highlight BridgeVLA’s advantages over existing manipulation policies, we compare it with four representative methods in this setting. The behaviors of these baselines are as follows:

SpatialVLA [37]: In the experimental setup, we initially trained SpatialVLA using only 10 trajectories per task. However, it failed on nearly all tasks, often struggling to move toward the correct target object. To improve performance, we augmented the dataset with an additional 40 trajectories per task. While this improved performance, it still lagged significantly behind BridgeVLA—particularly on more challenging tasks, such as "Put the giraffe in the lower drawer." These findings suggest that BridgeVLA provides a more effective and data-efficient solution for 3D VLA.

π_0 [5]: Similarly, π_0 fails with only 10 trajectories per task, likely due to overfitting—it performs well on the training set but often fails during online testing. Common failure modes include missing or failing to grasp the target and prematurely opening the gripper before reaching the goal. Notably, both BridgeVLA and π_0 share the same PaliGemma backbone and are trained end-to-end. This highlights a key contribution of our work: while VLAs like π_0 perform well with large-scale data, they struggle in low-data regimes—even on simpler tasks, such as "Press sanitizer." In contrast, BridgeVLA achieves near-perfect success and generalizes robustly across diverse settings.

ACT [51]: ACT also underperforms compared to BridgeVLA. It demonstrates limited spatial generalization, performing well only in areas densely covered during training, but often failing when the target is near the workspace boundaries. This behavior is consistent with its design: ACT models actions using a Gaussian prior, which assigns low probability to peripheral regions, limiting its spatial generalization capabilities.

RVT-2 [15]: RVT-2 performs the best among all the baselines. It can successfully solve most tasks, but it is not as robust as BridgeVLA. For instance, it sometimes fails to pick up the block precisely or place the object accurately, leading to task failure. Meanwhile, by utilizing the capabilities of VLM, BridgeVLA’s advantages are further amplified in generalization settings, as detailed in Sec. 4.2.

C.3 Generalization Settings

We evaluate on a total of six generalization settings: Distractor, Lighting, Background, Height, Combination, and Category. For Distractor, Lighting, Background, and Height, we visualize these settings in Fig. 10. We visualize the settings of Combination and Category in Fig. 11 and Fig. 12, respectively.

In Distractor, we add distractor objects that are visually similar to at least one target object to the scene. In Lighting, we evaluate the model in a novel lighting condition in which the lights are off. In Background, we use three different tablecloths to change the background. For Height, we evaluate all objects for manipulation with a drawer that is about 10cm high. Distractor, Lighting, Background, and Height aim to evaluate the robustness against visual disturbances.

In Combination, we combine objects and skills that are not paired together in the training datasets. That is, while the object for manipulation and the manipulation skill are seen during training, the instruction that pairs them together is novel. The setting of Combination helps us evaluate whether the model is able to generalize across novel object-skill combinations. In Category, we want to evaluate whether BridgeVLA is able to manipulate objects from categories that are *unseen* in the robot training data. In total, we test 7 novel objects.

C.4 Preservation of Object Grounding Capability after Fine-tuning

We observe that even after fine-tuning on robot action data, BridgeVLA retains the object grounding capability learned during pre-training. We visualize its predictions on the pre-training dataset after fine-tuning in Fig. 14. It is important to note that the samples in Fig. 14 are not cherry-picked. BridgeVLA does not forget its pre-training knowledge after 3D action fine-tuning.

C.5 Per-task Success Rate

We showcase per-task success rates of BridgeVLA in the basic setting in Tab. 12. Notably, BridgeVLA achieves exceptionally high success rates even with only 3 trajectories per task, highlighting its superb sample efficiency.

Task Name	Original	All Perturbations	MO-COLOR	RO-COLOR	MO-TEXTURE	RO-TEXTURE	MO-SIZE	RO-SIZE	Light Color	Table Color	Table Texture	Distractor	Background Texture	RL Bench	Camera Pose
basketball_in_hoop	100.0±0.0	4.0±3.3	94.7±1.9	96.0±0.0	84.0±5.7	-	100.0±0.0	68.0±0.0	100.0±0.0	100.0±0.0	100.0±0.0	37.3±1.9	100.0±0.0	100.0±0.0	100.0±0.0
close_box	100.0±0.0	72.0±0.0	94.7±1.9	-	-	-	93.3±1.9	-	100.0±0.0	100.0±0.0	98.7±1.9	98.7±1.9	100.0±0.0	97.3±1.9	100.0±0.0
close_laptop_lid	100.0±0.0	11.1±15.7	82.7±3.8	-	-	-	67.9±14.6	-	89.3±8.2	92.0±0.0	97.3±3.8	82.7±6.8	96.0±3.3	100.0±0.0	96.0±0.0
empty_dishwasher	0.0±0.0	0.0±0.0	1.3±1.9	1.3±1.9	-	1.3±1.9	4.0±3.3	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	1.3±1.9	1.3±1.9	0.0±0.0
get_ice_from_fridge	94.7±1.9	5.3±1.9	86.7±1.9	90.7±7.5	90.7±5.0	-	84.0±3.3	73.3±1.9	96.0±3.3	98.7±1.9	89.3±7.5	56.0±8.6	94.7±1.9	96.0±3.3	98.7±1.9
hockey	57.3±5.0	9.3±3.8	44.0±6.5	50.7±8.2	-	50.7±13.2	46.7±8.2	65.3±5.0	45.3±1.9	64.0±8.6	53.3±1.9	20.0±3.3	56.0±5.7	49.3±5.0	50.7±5.0
insert_onto_square_peg	93.3±3.8	23.3±2.4	52.0±3.3	94.7±1.9	-	76.0±8.6	85.3±3.8	70.7±3.8	84.0±0.0	88.0±3.3	88.0±3.3	44.0±11.8	86.7±1.9	77.3±5.0	96.0±0.0
meat_on_grill	96.0±0.0	9.3±1.9	32.0±0.0	88.0±5.7	-	-	100.0±0.0	-	100.0±0.0	92.0±6.5	90.7±1.9	98.7±1.9	97.3±1.9	100.0±0.0	100.0±0.0
move_hanger	37.3±3.8	2.7±3.8	26.7±3.8	46.7±3.8	-	-	-	-	52.0±0.0	84.0±0.0	52.0±5.7	52.0±5.7	33.3±5.0	42.7±1.9	24.0±0.0
open_drawer	96.0±0.0	60.0±3.3	97.3±1.9	-	-	-	90.7±1.9	-	88.0±3.3	93.3±1.9	100.0±0.0	90.7±1.9	100.0±0.0	94.7±1.9	96.0±0.0
place_wine_at_rack_location	88.0±5.7	17.3±13.6	82.7±5.0	89.3±7.5	-	92.0±6.5	93.3±3.8	90.7±3.8	90.7±5.0	97.3±1.9	88.0±3.3	74.7±3.8	90.7±6.8	92.0±3.3	92.0±8.6
put_money_in_safe	94.7±1.9	6.7±5.0	78.7±1.9	74.7±1.9	81.3±6.8	89.3±5.0	92.0±3.3	-	37.3±12.4	84.0±3.3	84.0±3.3	84.0±3.3	89.3±1.9	86.7±8.2	86.7±1.9
reach_and_drag	100.0±0.0	0.0±0.0	89.3±3.8	96.0±0.0	94.7±5.0	84.0±5.7	94.7±1.9	38.7±5.0	92.0±3.3	88.0±5.7	78.7±3.8	28.0±8.6	100.0±0.0	100.0±0.0	94.7±3.8
scoop_with_spatula	96.0±3.3	6.7±1.9	94.7±1.9	93.3±1.9	85.3±3.8	85.3±3.8	78.7±3.8	86.7±5.0	90.7±1.9	88.0±6.5	77.3±1.9	20.0±5.7	90.7±6.8	89.3±1.9	93.3±1.9
setup_chess	10.7±1.9	0.0±0.0	1.3±1.9	8.0±0.0	8.0±3.3	-	13.3±1.9	-	12.0±5.7	21.3±8.2	13.3±3.8	5.3±1.9	20.0±5.7	16.0±5.7	4.0±3.3
slide_block_to_target	100.0±0.0	24.0±3.3	74.7±1.9	-	92.0±3.3	-	-	-	100.0±0.0	100.0±0.0	98.7±1.9	84.0±9.8	100.0±0.0	100.0±0.0	100.0±0.0
stack_cups	58.7±3.8	29.3±1.9	66.7±1.9	-	50.7±1.9	-	44.0±3.3	-	62.7±1.9	64.0±3.3	65.3±8.2	26.7±7.5	73.3±8.2	64.0±14.2	72.0±8.6
straighten_ropes	61.3±6.8	8.0±5.7	16.0±5.7	-	48.0±3.3	-	-	-	61.3±9.4	65.3±1.9	54.7±8.2	37.3±5.0	70.7±8.2	66.7±7.5	72.0±6.5
turn_oven_on	93.3±1.9	85.3±3.8	94.7±3.8	-	-	-	90.7±1.9	-	93.3±3.8	94.7±7.5	96.0±3.3	96.0±3.3	96.0±0.0	88.0±3.3	100.0±0.0
wipe_desk	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	-	0.0±0.0	-	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
Task Mean	73.9±0.7	18.7±2.2	60.5±1.1	63.8±0.1	63.5±1.5	68.4±3.3	69.3±1.0	61.7±0.8	69.7±1.2	75.7±0.9	71.3±0.7	51.8±1.5	74.8±1.0	73.1±0.2	73.8±0.3

Table 6: Success Rates of BridgeVLA under Different Perturbations of COLOSSEUM.

Task Name	No variations	All Perturbations	MO-COLOR	RO-COLOR	MO-TEXTURE	RO-TEXTURE	MO-SIZE	RO-SIZE	Light Color	Table Color	Table Texture	Distractor	Background Texture	RL Bench	Camera Pose
basketball_in_hoop	100±0.0	40±4.9	99±1.7	33±1.7	96±0.0	-	99±1.7	100±0.0	87±4.4	54±2.0	91±11.4	55±9.1	100±0.0	97±1.7	100±0.0
close_box	96±4.9	32±16.7	43±3.3	-	-	-	91±5.2	-	84±2.8	78±3.5	91±9.1	96±2.8	98±3.5	96±2.8	95±3.3
close_laptop_lid	30±4.5	48±7.5	50±4.5	-	-	-	56±6.9	-	28±2.8	23±4.4	42±12.8	49±5.2	44±0.0	33±1.7	48±2.8
empty_dishwasher	0±0.0	0±0.0	0±0.0	0±0.0	-	1±1.7	1±1.7	1±1.7	0±0.0	0±0.0	0±0.0	0±0.0	0±0.0	1±1.7	1±1.7
get_ice_from_fridge	66±4.5	2±2.0	67±3.3	11±1.7	67±5.2	-	71±3.3	44±2.8	24±0.0	35±1.7	77±4.4	65±3.3	70±3.5	69±1.7	71±4.4
hockey	12±2.8	0±0.0	18±6.0	0±0.0	-	14±3.5	2±3.5	9±3.3	7±5.9	10±2.0	16±15.0	5±1.7	13±1.7	5±1.7	9±4.4
meat_on_grill	45±1.7	56±8.5	62±4.5	33±1.7	-	-	64±2.8	-	61±1.7	65±1.7	49±5.2	67±11.1	63±3.3	62±4.5	51±3.3
move_hanger	0±0.0	0±0.0	0±0.0	0±0.0	-	-	-	-	0±0.0	0±0.0	0±0.0	21±9.1	0±0.0	0±0.0	0±0.0
wipe_desk	0±0.0	0±0.0	0±0.0	0±0.0	0±0.0	-	0±0.0	-	0±0.0	0±0.0	0±0.0	0±0.0	0±0.0	0±0.0	0±0.0
open_drawer	97±1.7	9±5.9	100±0.0	-	-	-	100±0.0	-	90±4.5	56±0.0	100±0.0	89±5.2	100±0.0	97±1.7	96±0.0
slide_block_to_target	100±0.0	37±10.7	100±0.0	-	100±0.0	-	-	-	100±0.0	91±1.7	88±20.8	90±12.8	100±0.0	100±0.0	100±0.0
reach_and_drag	86±2.0	1±1.7	34±2.0	64±2.8	75±4.4	74±4.5	95±1.7	79±1.7	20±0.0	24±0.0	72±15.7	43±21.4	81±1.7	75±1.7	80±2.8
put_money_in_safe	63±1.7	1±1.7	62±2.0	5±1.7	58±2.0	75±1.7	64±2.8	-	60±0.0	47±3.3	82±4.5	60±4.9	60±0.0	60±2.8	48±0.0
place_wine_at_rack_location	96±4.9	59±11.4	94±4.5	94±2.0	-	96±2.8	91±1.7	88±2.8	87±5.9	93±4.4	94±2.0	80±16.0	88±2.8	95±3.3	96±4.9
insert_onto_square_peg	5±1.7	0±0.0	0±0.0	13±1.7	-	9±3.3	16±0.0	6±3.5	0±0.0	0±0.0	0±0.0	2±3.5	4±0.0	4±0.0	5±1.7
stack_cups	44±0.0	2±3.5	42±2.0	-	50±4.5	-	8±0.0	-	20±0.0	13±1.7	10±6.0	15±3.3	40±0.0	36±0.0	24±0.0
turn_oven_on	97±1.7	5±1.7	34±4.5	-	-	-	68±2.8	-	96±2.8	97±1.7	98±2.0	97±1.7	96±2.8	92±0.0	93±5.2
straighten_ropes	54±2.0	0±0.0	32±0.0	-	57±4.4	-	-	-	77±1.7	51±4.4	14±11.8	27±20.3	61±1.7	66±2.0	59±1.7
setup_chess	5±3.3	0±0.0	1±1.7	4±2.8	7±3.3	-	4±2.8	-	8±4.9	4±2.8	12±2.8	10±10.4	18±8.2	15±5.2	7±4.4
scoop_with_spatula	96±0.0	0±0.0	11±1.7	73±3.3	82±24.2	85±1.7	81±1.7	81±5.2	57±1.7	87±3.3	83±4.4	57±11.8	93±1.7	85±3.3	92±2.8
Average	55±0.5	15±1.9	42±0.6	25±0.2	59±2.6	51±1.8	54±1.0	51±1.3	45±0.3	41±0.0	51±2.7	46±2.2	56±0.5	54±0.5	54±0.4

Table 7: Success Rates of RVT-2 under Different Perturbations of COLOSSEUM.

Method	Avg.	Close Fridge+0	Close Jar+15	Close Jar+16	CloseLaptop Lid+0	Close Microwave+0	LightBulb In+17	LightBulb In+19	Open Box+0	Open Door+0	Open Drawer+0
Hiveformer [16]	60.3 $\pm$ 1.5	96 $\pm$ 4.2	64 $\pm$ 13.9	92 $\pm$ 2.7	90 $\pm$ 3.5	88 $\pm$ 7.6	12 $\pm$ 4.5	13 $\pm$ 6.7	4 $\pm$ 4.2	53 $\pm$ 15.2	15 $\pm$ 12.2
PolarNet [9]	77.6 $\pm$ 0.9	99 $\pm$ 2.2	99 $\pm$ 2.2	99 $\pm$ 2.2	95 $\pm$ 3.5	98 $\pm$ 2.7	72 $\pm$ 12.5	71 $\pm$ 6.5	32 $\pm$ 11.5	69 $\pm$ 8.9	61 $\pm$ 12.4
3D diffuser actor [25]	91.9 $\pm$ 0.8	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	99 $\pm$ 2.2	100 $\pm$ 0.0	85 $\pm$ 5.0	88 $\pm$ 2.7	11 $\pm$ 2.2	96 $\pm$ 4.2	82 $\pm$ 9.1
RVT-2 [15]	89.0 $\pm$ 0.8	77 $\pm$ 11.0	97 $\pm$ 4.5	98 $\pm$ 2.7	77 $\pm$ 13.0	100 $\pm$ 0.0	93 $\pm$ 5.7	91 $\pm$ 8.2	7 $\pm$ 4.5	98 $\pm$ 4.5	93 $\pm$ 5.7
3D-LOTUS [12]	94.3 $\pm$ 3.5	96 $\pm$ 3.7	100 $\pm$ 0.0	100 $\pm$ 0.0	98 $\pm$ 2.5	98 $\pm$ 4.0	84 $\pm$ 7.4	85 $\pm$ 9.5	99 $\pm$ 2.0	77 $\pm$ 2.5	83 $\pm$ 8.7
3D-LOTUS++ [12]	68.7 $\pm$ 0.6	95 $\pm$ 0.0	100 $\pm$ 0.0	99 $\pm$ 2.0	28 $\pm$ 2.5	87 $\pm$ 5.1	55 $\pm$ 10.5	45 $\pm$ 8.9	55 $\pm$ 8.9	79 $\pm$ 9.7	68 $\pm$ 12.5
BridgeVLA (Ours)	91.1 $\pm$ 1.1	99 $\pm$ 2.0	98 $\pm$ 4.0	100 $\pm$ 0.0	97 $\pm$ 2.5	85 $\pm$ 5.5	90 $\pm$ 5.5	87 $\pm$ 7.5	76 $\pm$ 10.2	70 $\pm$ 12.3	86 $\pm$ 5.8
Method	Open Drawer+2	Pick& Lift+0	Pick& Lift+2	Pick& Lift+7	PickUp Cup+8	PickUp Cup+9	PickUp Cup+11	Push Button+0	Push Button+3	Push Button+4	PutIn Cupboard+0
Hiveformer [16]	59 $\pm$ 7.4	86 $\pm$ 4.2	92 $\pm$ 6.7	93 $\pm$ 2.7	83 $\pm$ 7.6	69 $\pm$ 12.9	61 $\pm$ 19.8	84 $\pm$ 11.9	68 $\pm$ 6.7	87 $\pm$ 7.6	34 $\pm$ 8.2
PolarNet [9]	90 $\pm$ 7.1	92 $\pm$ 9.1	84 $\pm$ 7.4	88 $\pm$ 5.7	82 $\pm$ 7.6	79 $\pm$ 4.2	72 $\pm$ 10.4	100 $\pm$ 0.0	100 $\pm$ 0.0	99 $\pm$ 2.2	52 $\pm$ 7.6
3D diffuser actor [25]	97 $\pm$ 4.5	99 $\pm$ 2.2	99 $\pm$ 2.2	99 $\pm$ 2.2	96 $\pm$ 2.2	97 $\pm$ 4.5	98 $\pm$ 2.7	98 $\pm$ 2.7	96 $\pm$ 4.2	98 $\pm$ 2.7	85 $\pm$ 5.0
RVT-2 [15]	94 $\pm$ 4.2	99 $\pm$ 2.2	98 $\pm$ 2.7	100 $\pm$ 0.0	99 $\pm$ 2.2	99 $\pm$ 2.2	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	88 $\pm$ 8.4
3D-LOTUS [12]	93 $\pm$ 6.0	99 $\pm$ 2.0	100 $\pm$ 0.0	99 $\pm$ 2.0	97 $\pm$ 4.0	96 $\pm$ 3.7	94 $\pm$ 4.9	99 $\pm$ 2.0	99 $\pm$ 2.0	100 $\pm$ 0.0	89 $\pm$ 5.8
3D-LOTUS++ [12]	75 $\pm$ 4.5	97 $\pm$ 6.0	94 $\pm$ 3.7	93 $\pm$ 5.1	86 $\pm$ 8.0	88 $\pm$ 6.8	91 $\pm$ 4.9	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	1 $\pm$ 2.0
BridgeVLA (Ours)	99 $\pm$ 2.0	99 $\pm$ 2.0	100 $\pm$ 0.0	98 $\pm$ 2.5	96 $\pm$ 2.0	94 $\pm$ 3.7	99 $\pm$ 2.0	100 $\pm$ 0.0	98 $\pm$ 4.0	98 $\pm$ 4.0	74 $\pm$ 6.6
Method	PutIn Cupboard+3	PutMoney InSafe+0	PutMoney InSafe+1	Reach& Drag+14	Reach& Drag+18	Slide Block+0	Slide Block+1	Stack Blocks+30	Stack Blocks+36	Stack Blocks+39	
Hiveformer [16]	74 $\pm$ 6.5	85 $\pm$ 3.5	88 $\pm$ 2.7	37 $\pm$ 5.7	32 $\pm$ 7.6	99 $\pm$ 2.2	91 $\pm$ 12.4	6 $\pm$ 5.5	7 $\pm$ 4.5	6 $\pm$ 4.2	
PolarNet [9]	88 $\pm$ 4.5	93 $\pm$ 4.5	95 $\pm$ 5.0	99 $\pm$ 2.2	99 $\pm$ 2.2	100 $\pm$ 0.0	0 $\pm$ 0.0	34 $\pm$ 10.8	30 $\pm$ 9.4	36 $\pm$ 12.9	
3D diffuser actor [25]	82 $\pm$ 11.5	95 $\pm$ 5.0	98 $\pm$ 2.7	100 $\pm$ 0.0	99 $\pm$ 2.2	100 $\pm$ 0.0	89 $\pm$ 4.2	88 $\pm$ 7.6	85 $\pm$ 6.1	89 $\pm$ 5.5	
RVT-2 [15]	80 $\pm$ 6.1	93 $\pm$ 8.4	96 $\pm$ 8.5	85 $\pm$ 10.0	94 $\pm$ 2.2	100 $\pm$ 0.0	37 $\pm$ 6.7	88 $\pm$ 5.7	93 $\pm$ 2.7	88 $\pm$ 11.5	
3D-LOTUS [12]	72 $\pm$ 11.2	94 $\pm$ 3.7	99 $\pm$ 2.0	99 $\pm$ 2.0	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	94 $\pm$ 5.8	91 $\pm$ 6.6	90 $\pm$ 4.5	
3D-LOTUS++ [12]	2 $\pm$ 2.5	22 $\pm$ 6.8	16 $\pm$ 4.9	94 $\pm$ 3.7	62 $\pm$ 8.7	100 $\pm$ 0.0	65 $\pm$ 5.5	86 $\pm$ 5.8	20 $\pm$ 4.5	28 $\pm$ 13.6	
BridgeVLA (Ours)	84 $\pm$ 6.6	79 $\pm$ 9.7	86 $\pm$ 3.7	96 $\pm$ 5.8	97 $\pm$ 4.0	100 $\pm$ 0.0	90 $\pm$ 5.5	77 $\pm$ 8.1	87 $\pm$ 4.0	85 $\pm$ 7.8	

Table 8: Per-task Success Rate on GemBench Level 1.

Method	Avg.	Push Button+13	Push Button+15	Push Button+17	Pick& Lift+14	Pick& Lift+16	Pick& Lift+18	PickUp Cup+10	PickUp Cup+12	PickUp Cup+13
Hiveformer	26.1 $\pm$ 1.4	97 $\pm$ 2.7	85 $\pm$ 10.0	88 $\pm$ 2.7	21 $\pm$ 6.5	9 $\pm$ 4.2	8 $\pm$ 6.7	30 $\pm$ 7.1	22 $\pm$ 13.5	26 $\pm$ 10.6
PolarNet	37.1 $\pm$ 1.4	100 $\pm$ 0.0	100 $\pm$ 0.0	85 $\pm$ 7.9	3 $\pm$ 4.5	1 $\pm$ 2.2	0 $\pm$ 0.0	48 $\pm$ 11.0	46 $\pm$ 8.9	16 $\pm$ 6.5
3D diffuser actor	43.4 $\pm$ 2.8	87 $\pm$ 13.0	81 $\pm$ 6.5	60 $\pm$ 9.4	9 $\pm$ 4.2	18 $\pm$ 9.1	0 $\pm$ 0.0	84 $\pm$ 5.5	60 $\pm$ 11.7	62 $\pm$ 13.0
RVT-2	51.0 $\pm$ 2.3	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	47 $\pm$ 7.6	29 $\pm$ 9.6	8 $\pm$ 4.5	81 $\pm$ 8.2	59 $\pm$ 9.6	72 $\pm$ 9.7
3D-LOTUS	49.9 $\pm$ 2.2	99 $\pm$ 2.0	100 $\pm$ 0.0	100 $\pm$ 0.0	3 $\pm$ 2.5	18 $\pm$ 8.7	33 $\pm$ 9.3	89 $\pm$ 3.7	78 $\pm$ 8.7	57 $\pm$ 7.5
3D-LOTUS++	64.5 $\pm$ 0.9	99 $\pm$ 2.0	100 $\pm$ 0.0	99 $\pm$ 2.0	94 $\pm$ 3.7	96 $\pm$ 3.7	95 $\pm$ 3.2	79 $\pm$ 4.9	89 $\pm$ 9.7	84 $\pm$ 10.2
BridgeVLA (Ours)	65.0 $\pm$ 1.3	100 $\pm$ 0.0	100 $\pm$ 0.0	100 $\pm$ 0.0	74 $\pm$ 9.7	89 $\pm$ 4.9	0 $\pm$ 0.0	91 $\pm$ 3.7	90 $\pm$ 3.2	90 $\pm$ 6.3
Method	Stack Blocks+24	Stack Blocks+27	Stack Blocks+33	Slide Block+2	Slide Block+3	Close Jar+3	Close Jar+4	LightBulb In+1	LightBulb In+2	Lamp On+0
Hiveformer	0 $\pm$ 0.0	4 $\pm$ 4.2	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	4 $\pm$ 4.2	0 $\pm$ 0.0	7 $\pm$ 4.5
PolarNet	1 $\pm$ 2.2	2 $\pm$ 2.7	6 $\pm$ 8.2	0 $\pm$ 0.0	0 $\pm$ 0.0	20 $\pm$ 10.6	82 $\pm$ 5.7	22 $\pm$ 11.5	17 $\pm$ 8.4	14 $\pm$ 10.8
3D diffuser actor	66 $\pm$ 13.9	82 $\pm$ 2.7	50 $\pm$ 14.6	0 $\pm$ 0.0	0 $\pm$ 0.0	23 $\pm$ 16.8	82 $\pm$ 5.7	51 $\pm$ 17.8	60 $\pm$ 10.0	7 $\pm$ 7.6
RVT-2	18 $\pm$ 4.5	56 $\pm$ 16.7	45 $\pm$ 13.7	0 $\pm$ 0.0	1 $\pm$ 2.2	7 $\pm$ 7.6	77 $\pm$ 5.7	68 $\pm$ 14.4	6 $\pm$ 6.5	0 $\pm$ 0.0
3D-LOTUS	13 $\pm$ 8.1	40 $\pm$ 9.5	69 $\pm$ 5.8	0 $\pm$ 0.0	0 $\pm$ 0.0	71 $\pm$ 5.8	90 $\pm$ 4.5	24 $\pm$ 4.9	41 $\pm$ 8.6	0 $\pm$ 0.0
3D-LOTUS++	22 $\pm$ 9.3	83 $\pm$ 7.5	59 $\pm$ 3.7	27 $\pm$ 9.8	5 $\pm$ 3.2	98 $\pm$ 2.5	96 $\pm$ 3.7	56 $\pm$ 9.7	43 $\pm$ 7.5	2 $\pm$ 2.0
BridgeVLA (Ours)	61 $\pm$ 10.7	51 $\pm$ 13.2	79 $\pm$ 8.6	12 $\pm$ 9.3	3 $\pm$ 4.0	66 $\pm$ 6.6	88 $\pm$ 4.0	66 $\pm$ 8.6	74 $\pm$ 5.8	7 $\pm$ 4.0
Method	Reach& Drag+5	Reach& Drag+7	PutCube InSafe+0	Pick&Lift Cylinder+0	Pick&Lift Star+0	Pick&Lift Moon+0	Pick&Lift Toy+0	PutIn Cupboard+7	PutIn Cupboard+8	
Hiveformer	1 $\pm$ 2.2	0 $\pm$ 0.0	4 $\pm$ 2.2	78 $\pm$ 5.7	73 $\pm$ 7.6	88 $\pm$ 2.7	87 $\pm$ 4.5	0 $\pm$ 0.0	0 $\pm$ 0.0	
PolarNet	61 $\pm$ 8.2	10 $\pm$ 6.1	40 $\pm$ 14.1	93 $\pm$ 6.7	88 $\pm$ 8.4	93 $\pm$ 6.7	90 $\pm$ 3.5	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D diffuser actor	0 $\pm$ 0.0	64 $\pm$ 6.5	3 $\pm$ 2.7	99 $\pm$ 2.2	43 $\pm$ 17.9	91 $\pm$ 9.6	30 $\pm$ 9.4	0 $\pm$ 0.0	3 $\pm$ 4.5	
RVT-2	91 $\pm$ 2.2	89 $\pm$ 6.5	6 $\pm$ 5.5	98 $\pm$ 2.7	98 $\pm$ 4.5	94 $\pm$ 4.2	78 $\pm$ 8.4	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D-LOTUS	95 $\pm$ 4.5	18 $\pm$ 10.8	25 $\pm$ 5.5	88 $\pm$ 8.7	69 $\pm$ 6.6	80 $\pm$ 8.4	96 $\pm$ 3.7	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D-LOTUS++	94 $\pm$ 2.0	64 $\pm$ 12.4	37 $\pm$ 5.1	91 $\pm$ 2.0	94 $\pm$ 3.7	29 $\pm$ 6.6	71 $\pm$ 2.0	1 $\pm$ 2.0	0 $\pm$ 0.0	
BridgeVLA (Ours)	94 $\pm$ 3.7	96 $\pm$ 3.7	3 $\pm$ 2.5	98 $\pm$ 2.5	99 $\pm$ 2.0	95 $\pm$ 3.2	93 $\pm$ 5.1	0 $\pm$ 0.0	0 $\pm$ 0.0	

Table 9: Per-task Success Rate on GemBench Level 2.

Method	Avg.	Close Door+0	Close Box+0	Close Fridge2+0	CloseLaptop Lid2+0	Close Microwave2+0	Open Door2+0	Open Box2+0
Hiveformer	35.1 $\pm$ 1.7	0 $\pm$ 0.0	1 $\pm$ 2.2	34 $\pm$ 9.6	52 $\pm$ 9.1	15 $\pm$ 7.1	32 $\pm$ 11.5	5 $\pm$ 3.5
PolarNet	38.5 $\pm$ 1.7	0 $\pm$ 0.0	0 $\pm$ 0.0	78 $\pm$ 5.7	26 $\pm$ 8.2	74 $\pm$ 6.5	33 $\pm$ 6.7	23 $\pm$ 8.4
3D diffuser actor	37.0 $\pm$ 2.2	0 $\pm$ 0.0	0 $\pm$ 0.0	97 $\pm$ 2.7	23 $\pm$ 6.7	88 $\pm$ 7.6	86 $\pm$ 7.4	67 $\pm$ 9.8
RVT-2	36.0 $\pm$ 2.2	1 $\pm$ 2.2	2 $\pm$ 2.7	72 $\pm$ 6.7	42 $\pm$ 14.0	71 $\pm$ 8.9	79 $\pm$ 6.5	5 $\pm$ 6.1
3D-LOTUS	38.1 $\pm$ 1.1	0 $\pm$ 0.0	58 $\pm$ 8.1	36 $\pm$ 9.7	54 $\pm$ 10.7	85 $\pm$ 7.1	42 $\pm$ 6.8	11 $\pm$ 6.6
3D-LOTUS++	41.5 $\pm$ 1.8	1 $\pm$ 2.0	29 $\pm$ 8.6	93 $\pm$ 2.5	50 $\pm$ 9.5	99 $\pm$ 2.0	52 $\pm$ 10.3	16 $\pm$ 8.0
BridgeVLA (Ours)	43.8 $\pm$ 1.2	0 $\pm$ 0.0	1 $\pm$ 2.0	95 $\pm$ 5.5	77 $\pm$ 4.0	54 $\pm$ 10.2	68 $\pm$ 10.8	74 $\pm$ 4.9
Method	Open Drawer2+0	Open Drawer3+0	OpenDrawer Long+0	OpenDrawer Long+1	OpenDrawer Long+2	OpenDrawer Long+3	Toilet SeatUp+0	Open Fridge+0
Hiveformer	59 $\pm$ 11.9	39 $\pm$ 11.9	78 $\pm$ 8.4	82 $\pm$ 4.5	49 $\pm$ 4.2	57 $\pm$ 11.5	6 $\pm$ 4.2	0 $\pm$ 0.0
PolarNet	91 $\pm$ 4.2	29 $\pm$ 8.2	84 $\pm$ 11.9	88 $\pm$ 5.7	63 $\pm$ 8.4	37 $\pm$ 7.6	2 $\pm$ 2.7	4 $\pm$ 2.2
3D diffuser actor	19 $\pm$ 8.2	1 $\pm$ 2.2	15 $\pm$ 5.0	35 $\pm$ 13.7	26 $\pm$ 9.6	79 $\pm$ 12.9	0 $\pm$ 0.0	7 $\pm$ 5.7
RVT-2	81 $\pm$ 11.9	0 $\pm$ 0.0	84 $\pm$ 8.2	39 $\pm$ 10.8	11 $\pm$ 8.9	75 $\pm$ 6.1	7 $\pm$ 5.7	0 $\pm$ 0.0
3D-LOTUS	90 $\pm$ 3.2	22 $\pm$ 8.1	56 $\pm$ 13.9	33 $\pm$ 11.2	17 $\pm$ 8.1	75 $\pm$ 6.3	0 $\pm$ 0.0	4 $\pm$ 5.8
3D-LOTUS++	70 $\pm$ 5.5	41 $\pm$ 4.9	72 $\pm$ 4.0	52 $\pm$ 10.8	23 $\pm$ 8.1	78 $\pm$ 5.1	8 $\pm$ 5.1	0 $\pm$ 0.0
BridgeVLA (Ours)	65 $\pm$ 6.3	87 $\pm$ 6.0	59 $\pm$ 8.6	34 $\pm$ 8.0	18 $\pm$ 10.3	85 $\pm$ 8.4	6 $\pm$ 5.8	7 $\pm$ 2.5
Method	OpenLaptop Lid+0	Open Microwave+0	PutMoney InSafe+2	Open Drawer+1	Close Drawer+0	Close Grill+0		
Hiveformer	100 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	83 $\pm$ 5.7	44 $\pm$ 10.8		
PolarNet	100 $\pm$ 0.0	0 $\pm$ 0.0	1 $\pm$ 2.2	4 $\pm$ 4.2	29 $\pm$ 11.9	42 $\pm$ 11.5		
3D diffuser actor	100 $\pm$ 0.0	0 $\pm$ 0.0	2 $\pm$ 4.5	0 $\pm$ 0.0	66 $\pm$ 7.4	65 $\pm$ 13.7		
RVT-2	93 $\pm$ 5.7	0 $\pm$ 0.0	0 $\pm$ 0.0	6 $\pm$ 2.2	78 $\pm$ 8.4	9 $\pm$ 4.2		
3D-LOTUS	100 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	87 $\pm$ 8.1	29 $\pm$ 6.6		
3D-LOTUS++	86 $\pm$ 6.6	0 $\pm$ 0.0	13 $\pm$ 8.1	0 $\pm$ 0.0	69 $\pm$ 5.8	19 $\pm$ 13.9		
BridgeVLA (Ours)	95 $\pm$ 0.0	0 $\pm$ 0.0	2 $\pm$ 2.5	0 $\pm$ 0.0	58 $\pm$ 12.9	35 $\pm$ 12.3		

Table 10: Per-task Success Rate on GemBench Level 3.

Method	Avg.	Push Buttons4+1	Push Buttons4+2	Push Buttons4+3	TakeShoes OutOfBox+0	PutItems InDrawer+0	PutItems InDrawer+2
Hiveformer	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
PolarNet	0.1 $\pm$ 0.2	1 $\pm$ 2.2	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
3D diffuser actor	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
RVT-2	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
3D-LOTUS	0.3 $\pm$ 0.3	3 $\pm$ 4.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
3D-LOTUS++	17.4 $\pm$ 0.4	76 $\pm$ 7.4	49 $\pm$ 8.6	37 $\pm$ 8.1	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
BridgeVLA (Ours)	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0
Method	PutItems InDrawer+4	Tower4+1	Tower4+3	Stack Cups+0	Stack Cups+3	PutAllGroceries InCupboard+0	
Hiveformer	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
PolarNet	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D diffuser actor	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
RVT-2	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D-LOTUS	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
3D-LOTUS++	0 $\pm$ 0.0	17 $\pm$ 10.8	30 $\pm$ 13.4	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	
BridgeVLA (Ours)	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	0 $\pm$ 0.0	

Table 11: Per-task Success Rate on GemBench Level 4.



Figure 5: **Visualization of 18 RL Bench [19] Tasks.**

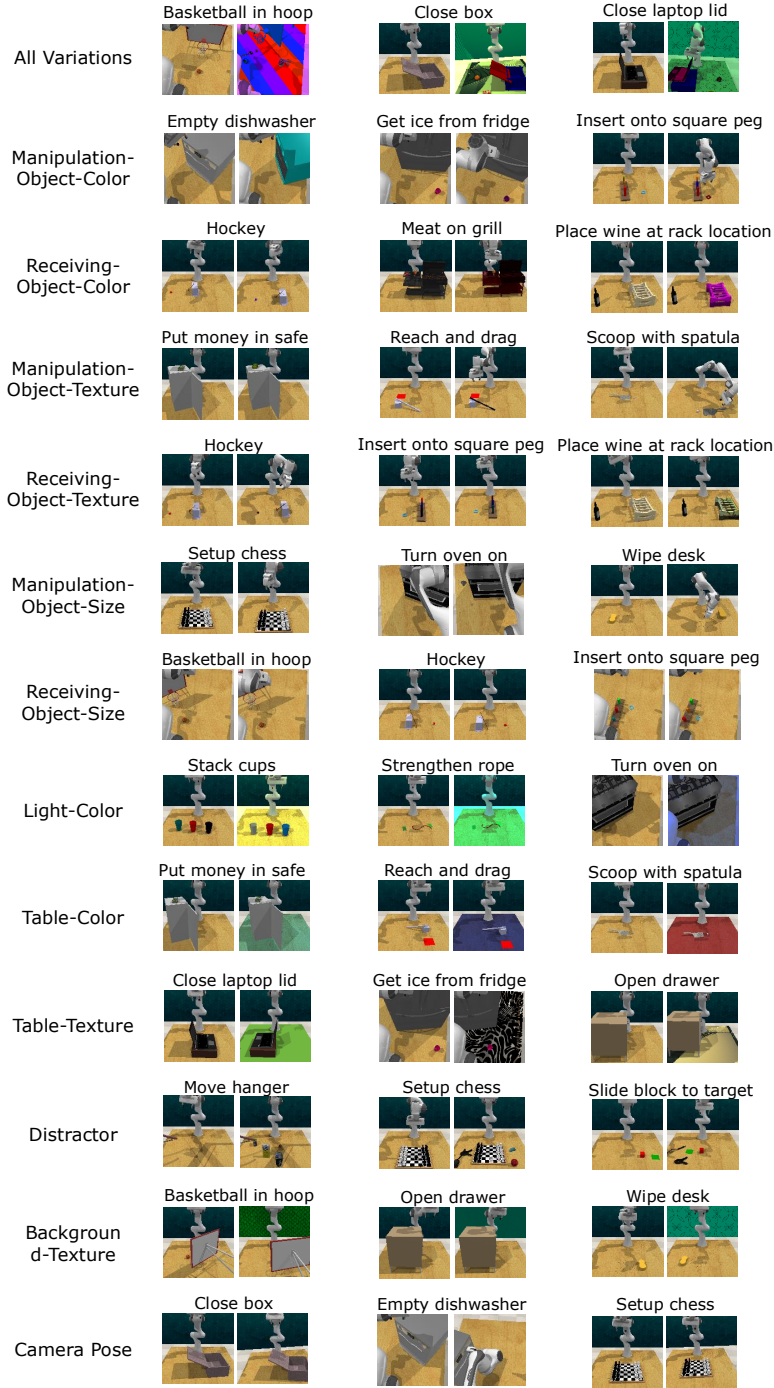


Figure 6: Visualization of Perturbations in COLOSSEUM [35].

Task	3 trajectories	10 trajectories
Put the RedBull can in the top shelf	9/10	10/10
Put the soda can in the bottom shelf	9/10	9/10
Put the RedBull can in the bottom shelf	10/10	10/10
Put the coke can in the top shelf	10/10	10/10
Place the red block in the blue plate	10/10	10/10
Place the orange block in the green plate	10/10	10/10
Put the wolf in the upper drawer	7/10	9/10
Place the red block in the purple plate	10/10	10/10
Place the yellow block in the green plate	10/10	10/10
Press sanitizer	10/10	10/10
Put the zebra in the upper drawer	9/10	9/10
Put the giraffe in the lower drawer	10/10	9/10
Put the zebra in the lower drawer	10/10	10/10

Table 12: **Per-task Success Rates of BridgeVLA in the Basic Setting.**

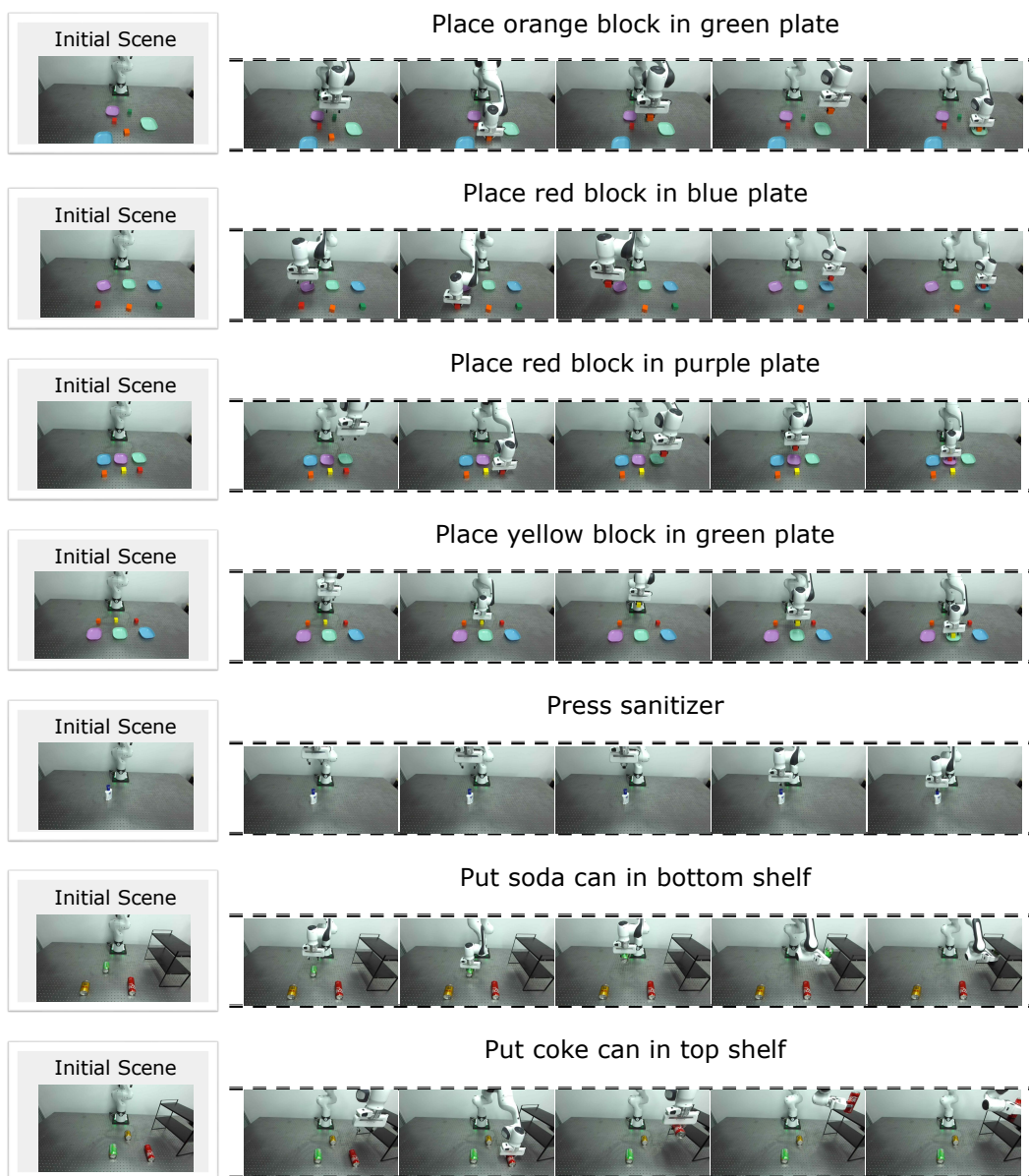


Figure 7: **Real-Robot Rollouts (I).**

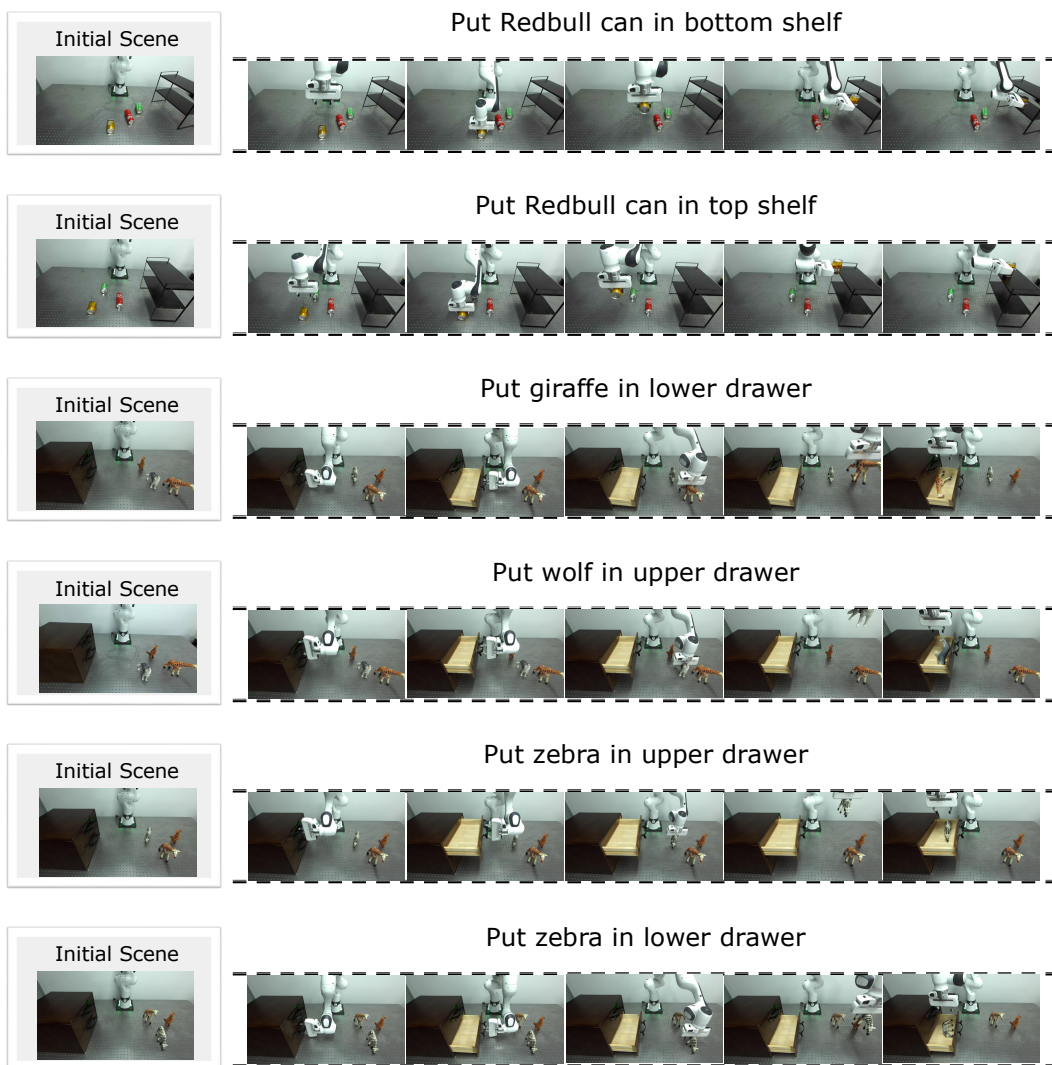


Figure 8: **Real-Robot Rollouts (II).**

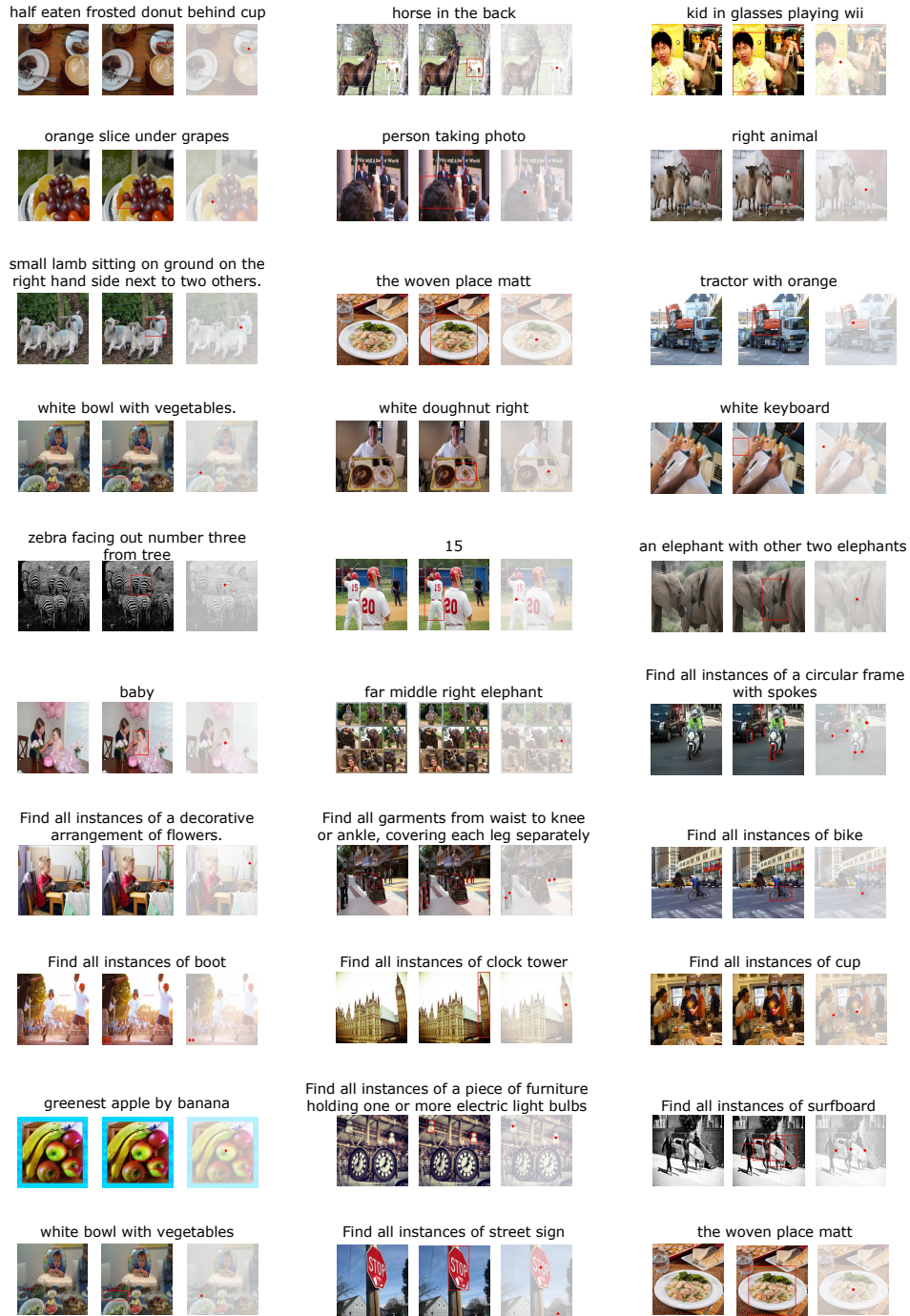


Figure 9: **Visualization of Pre-training Data.** We list some samples of pre-training data. For every sample, the left shows the original image; the middle shows the bounding boxes of the objects of interest; the right shows the ground-truth heatmap used for training.

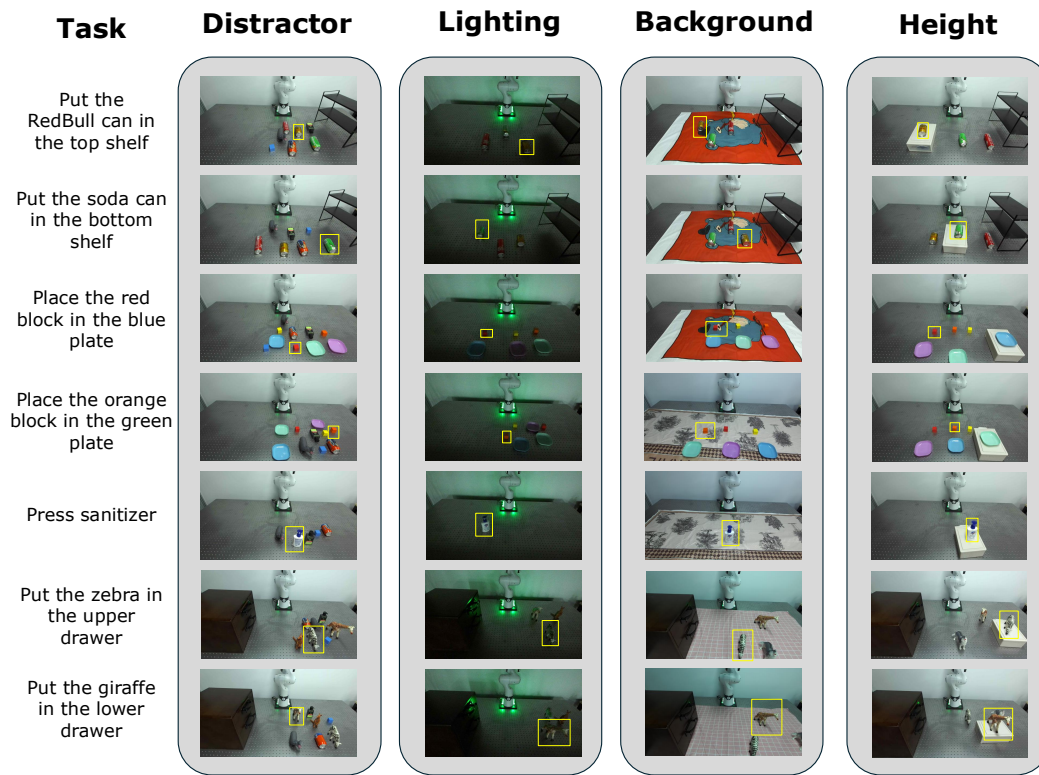


Figure 10: Visualization of the Distractor, Lighting, Background, and Height settings.

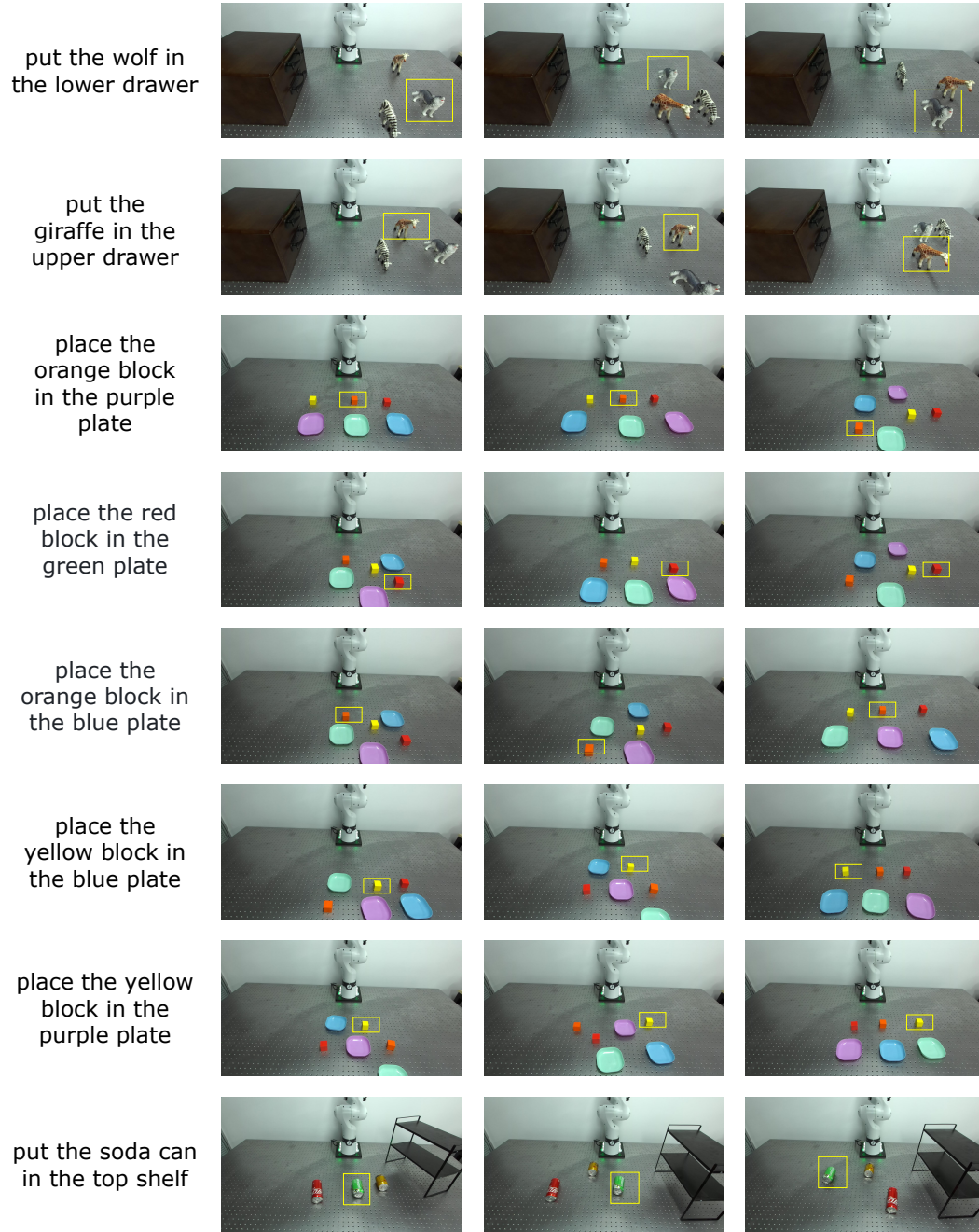
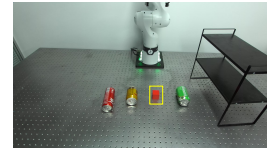
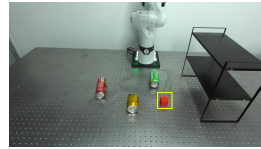
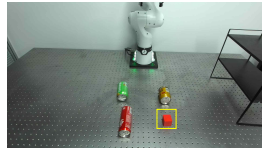
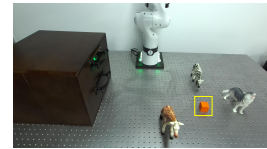
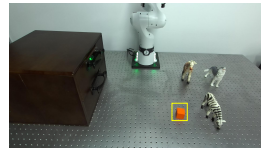
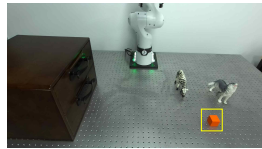


Figure 11: **Visualization of the Combination Setting (I).** During training, the manipulated objects and skills are seen, but their combinations are unseen.

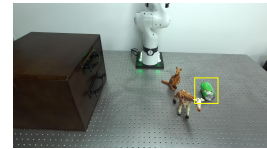
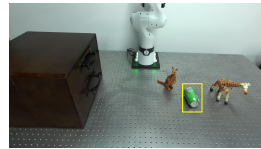
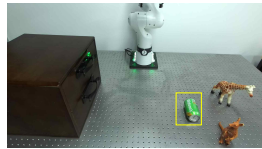
Put the red block
in the bottom
shelf



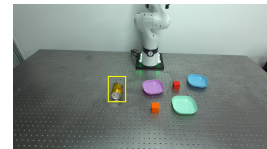
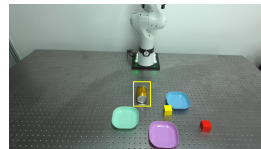
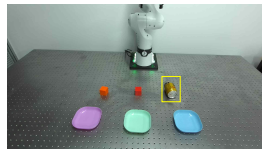
Put the orange
block in the lower
drawer



Put the soda can
in the upper
drawer



Put the Redbull
can in the green
plate



Place the zebra in
the blue plate

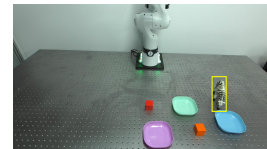
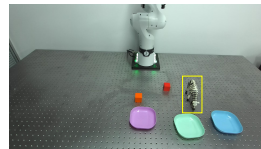
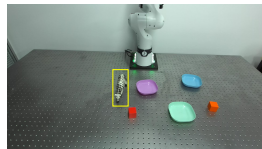


Figure 12: **Visualization of the Combination Setting (II).** During training, the manipulated objects and skills are seen, but their combinations are unseen.

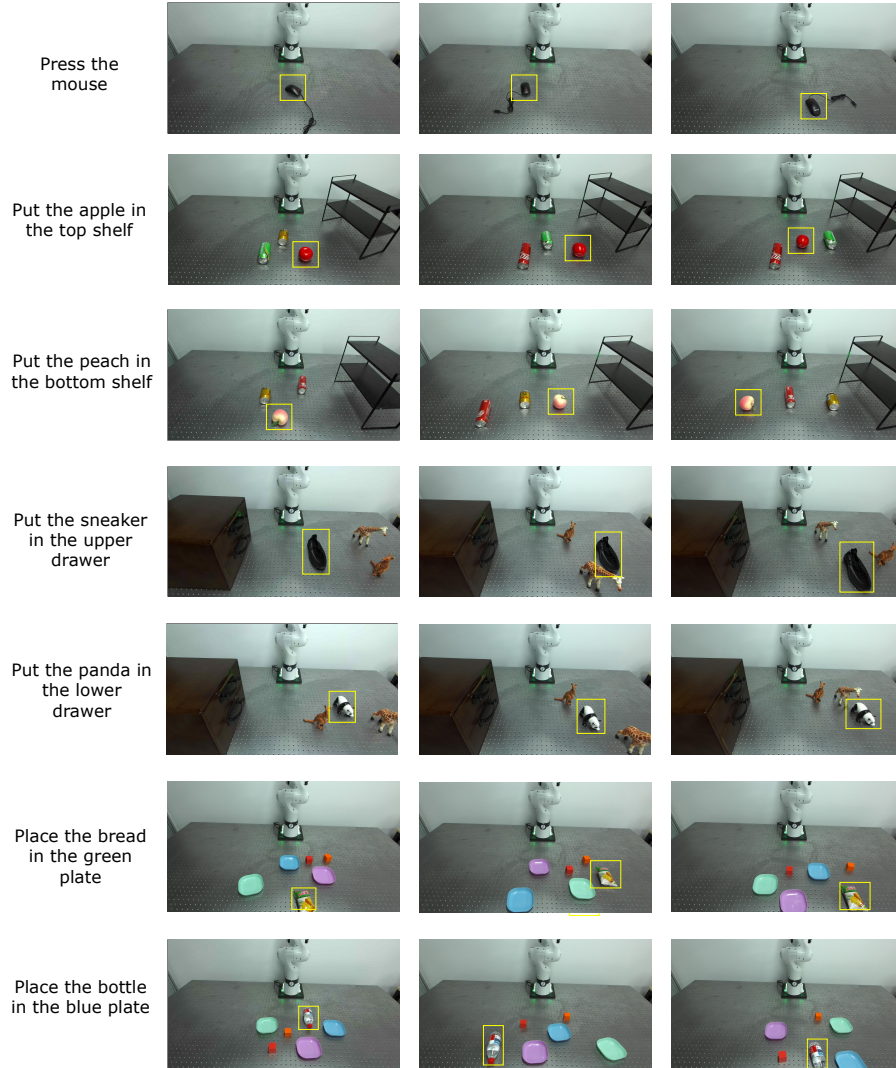


Figure 13: **Visualization of the Category Setting.** In total, we evaluate on 7 objects from novel categories that are unseen during training.

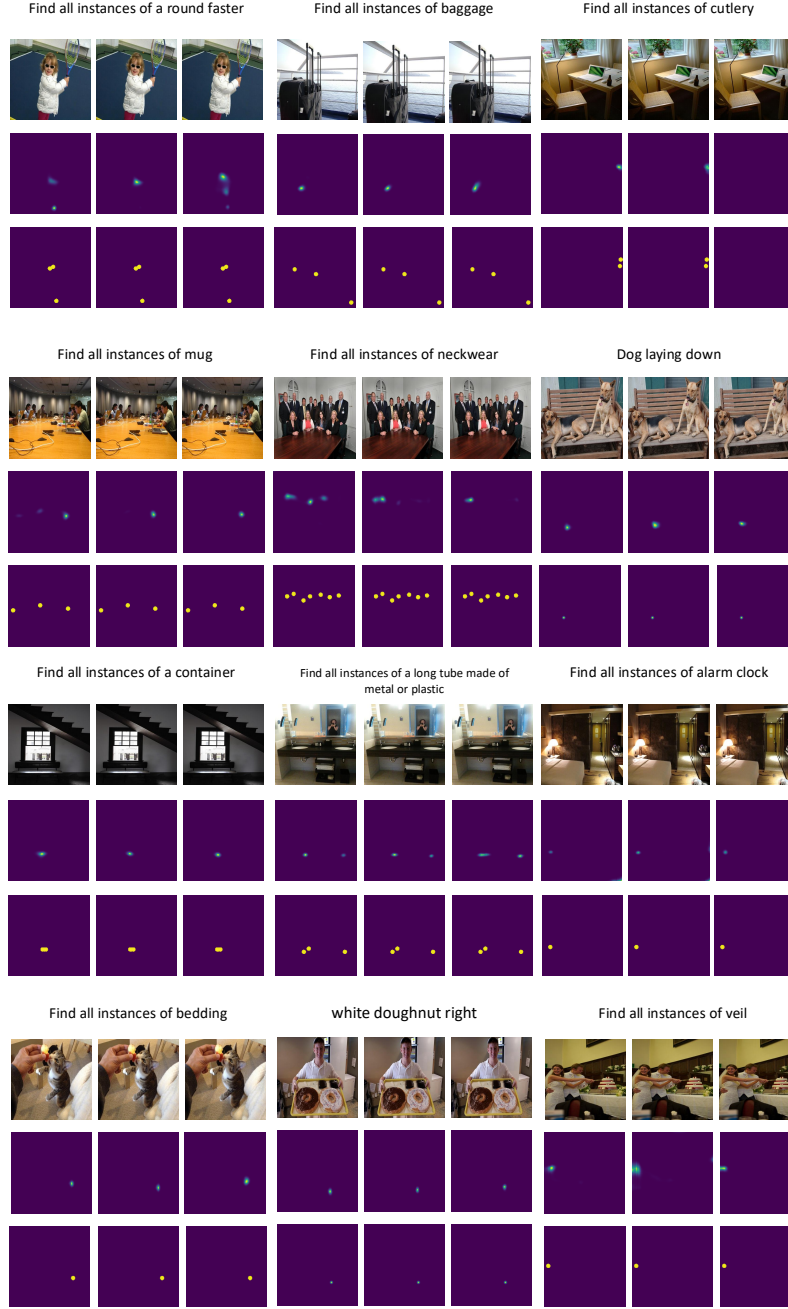


Figure 14: **Visualization of BridgeVLA’s Prediction on Pre-training Dataset after Fine-tuning.** To simulate the multi-view inputs during fine-tuning, we repeat the input image three times and feed them into the fine-tuned model to generate heatmaps. For each sample, the first row shows the input image; the second row shows the heatmap prediction; the third row shows the ground truth.