3DS-VLA: A 3D Spatial-Aware Vision Language Action Model for Robust Multi-Task Manipulation

Xiaoqi Li[1,2], Liang Heng[1], Jiaming Liu[3], Yan Shen[1,2], Chenyang Gu[3], Zhuoyang Liu[3], Hao Chen[4], Nuowei Han[1], Renrui Zhang[4], Hao Tang[3], Shanghang Zhang[3], Hao Dong[1,2]
[1]CFCS, School of Computer Science, Peking University, [2]PKU-Agibot Lab, [3]State Key Laboratory of Multimedia Information Processing, School of Computer Science, Peking University [4]CUHK
xl3062@columbia.edu

Abstract: Recently, 2D vision-language-action (VLA) models have made significant strides in multi-task manipulation. However, these models struggle to reason about 3D spatial relationships from 2D image inputs. Although an increasing number of 3D imitation learning approaches explicitly integrate 3D information, they face challenges such as the lack of generalized 3D pretrained models due to the limited availability of large-scale 3D datasets. Meanwhile, existing policies typically focus on the perception-to-action learning paradigm, lacking an explicit understanding of the spatial and temporal relationships between the robot and its environment. To address this, we propose 3DS-VLA, which enhances pretrained 2D vision-language models (VLMs) with comprehensive 3D awareness, enabling the prediction of robust end-effector poses. Specifically, we enable the 2D vision encoder of the VLMs to encode both 2D images and *3D spatial observation* by introducing a 2D-to-3D positional alignment mechanism. This allows 3DS-VLA to leverage the large-scale pre-trained knowledge of the VLM for effective reasoning in complex 3D robotic environments. Furthermore, to better understand the spatiotemporal relationship between 3D observations and robot behavior, we guide the model to learn the introduced sequential *3D spatial constraints*, which describe the relationship between affordance-relevant objects and robotic actions. Experiments in simulation and real world demonstrate that 3DS-VLA outperforms previous state-of-the-art policies and showcase its generalizable capabilities across multi-task, multi-embodiment, and diverse environmental settings.

Keywords: Vision-Language-Action, Robotic Manipulation, Imitation Learning

1 Introduction

The main objective of vision-language imitation learning policies is to execute actions by taking into account visual observations and language conditions. Recently, driven by the rapid advancement of 2D vision-language models (VLMs) [1, 2, 3, 4] in general scenes, a range of 2D vision-language-action (VLA) methods have been introduced for action generation [5, 6, 7, 8, 9, 10, 11]. However, since robots operate in a complex 3D world, they face challenges in perceiving 3D geometry and reasoning about spatial context solely from 2D image observations [12, 13, 14].

In contrast, 3D imitation learning approaches integrate 3D geometric information for policy learning. Some methods [15, 16, 17, 18, 19, 14, 20, 21, 22] train 3D vision-language models (e.g., PointNet [23] and BERT [24])from scratch or by fine-tuning them for action prediction. However, unlike 2D policy models that have access to large-scale datasets, the scarcity of large-scale 3D data limits these methods' scalability in complex robotic environments. Other methods attempt to leverage 2D pre-trained models for 3D imitation learning, either by using 2D models to encoder multi-view images that are projected from 3D data [25, 26, 27], which inevitably leads to spatial information loss during the 3D-to-2D transformation, or by lifting 2D image features extracted by

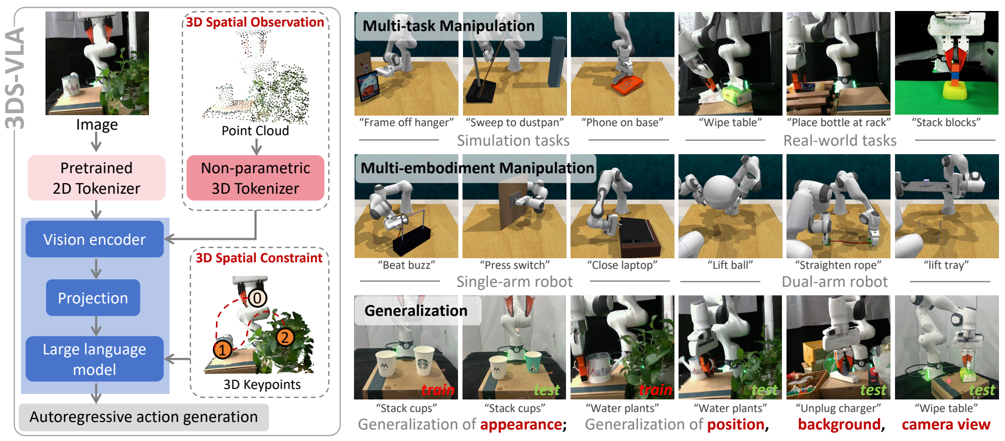


Figure 1: **3DS-VLA** achieves comprehensive 3D spatial awareness by encoding 3D spatial observations with a pretrained 2D vision-language model and establishing 3D spatial constraints to facilitate spatial-temporal reasoning. It demonstrates generalizable capabilities across tasks, embodiments, and environmental settings.

2D models into 3D latent spaces with depth map [28, 29, 30, 31], which are incapable of capturing the rich geometric information embedded in raw point clouds. Moreover, most methods learn a direct mapping from perception-to-action [14], predicting end-effector poses only conditioned on vision and language inputs. Yet, robotic manipulation requires intricate environmental interactions, and such methods [32, 33, 34, 35, 36] often lack a broader understanding of the robot's action with its surroundings in terms of spatial and temporal. All these limitations lead us to consider: "*How can we build a robust VLA model that incorporates comprehensive 3D spatial awareness?*"

To address the above challenges, as shown in Fig. 1 (left), we propose **3DS-VLA**, which equips pretrained 2D vision-language models (2D VLMs) with 3D spatial awareness for robust action generation. Firstly, to enhance the model's understanding of ***3D spatial observations***, we process both point clouds and images using a shared 2D visual encoder. The image processing pipeline remains unchanged, while we introduce a non-parametric 3D tokenizer before the vision encoder to directly convert point cloud data into 3D tokens without introducing additional computational overhead. In addition, we design a 2D-to-3D positional alignment mechanism that geometrically aligns each 3D token with the pretrained 2D positional embedding (PE) pointing to the same spatial region. Through this alignment, the 3D tokens are spatially encoded using pretrained 2D PEs with corresponding spatial and semantic knowledge, enabling them to be interpreted by the VLMs and thereby enhancing the model's ability to reason about 3D spatial context. Additionally, to improve the understanding of the relationship between the environment and robot action, we introduce ***3D spatial constraints***, represented as sequential 3D keypoints in Cartesian coordinates. These keypoints correspond to entities like end effector and objects. The constraints construct dynamic affordance conditions from sequential keypoints, explicitly encoding "where" and "when" the robot should interact with the environment. We then propose a text-based keypoint formulation for training, enabling 3DS-VLA to reason about these constraints and predict actions robustly.

3DS-VLA surpasses state-of-the-art VLA methods and 3D imitation policies across 26 single- and dual-arm tasks in RLBench [37, 38] and 10 real world tasks. Our contributions are as follows: 1) We propose **3DS-VLA**, equipping pretrained 2D VLMs with comprehensive 3D awareness for robust end-effector pose prediction. 2) We introduce a 2D-to-3D positional alignment mechanism to process ***3D spatial observation***, enabling 3DS-VLA to leverage pretrained 2D VLM knowledge for 3D spatial understanding. 3) We establish ***3D spatial constraints*** for policy learning, enabling 3DS-VLA to reason about the spatial-temporal relationships between robot and environment.

2 Related work

Vision Language Action Models. Vision-Language-Action (VLA) models have advanced rapidly to enable VLMs to predict actions. Methods like RT2 [8, 39, 10, 40, 41, 42, 43, 44] discretize 7D

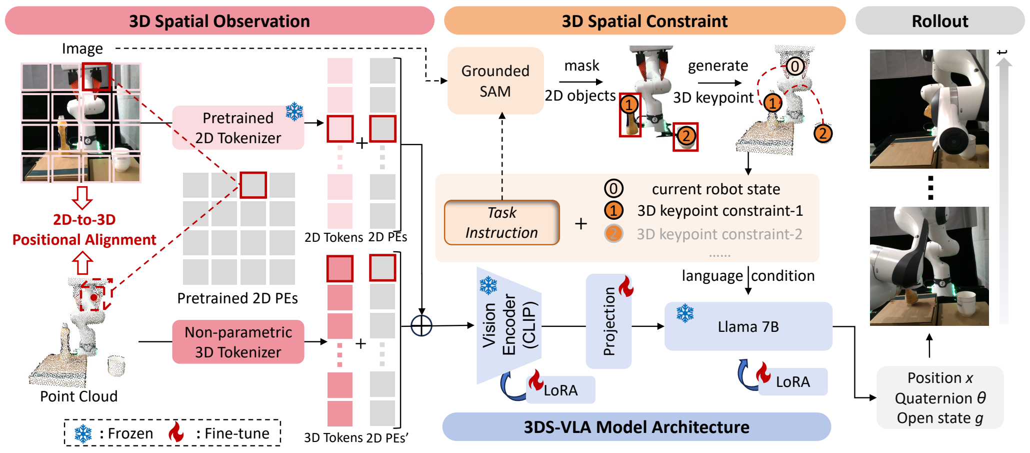


Figure 2: **Model Architecture.** Given the current observation, task instruction, and keypoint constraints, 3DS-VLA predicts the next-frame pose. It incorporates 3D spatial observations and 3D spatial constraints to enhance 3D spatial awareness. The first component uses a 2D visual encoder to encode both the 2D image and the 3D point cloud. The second component guides the model to follow 3D constraint priors between the robot and the environment.

actions into tokens, while others treat action prediction as continuous regression, using MLPs [9, 11] or diffusion transformers [45, 46, 47]. We adopt autoregressive generation for its flexibility and scalability to both single- and dual-arm tasks within a single model. Moreover, most VLA models [9, 45, 48, 7, 49, 50, 51] rely solely on 2D images, ignoring 3D geometric information. To address this, we enable the 2D visual encoder to process both 2D images and 3D point clouds, enhancing the model's 3D spatial awareness.

3D Imitation Learning. One category of works [52, 53, 54, 16, 55, 56, 15, 29, 57, 29, 58, 18] encodes 3D information using 3D visual encoders to predict end-effector poses. However, lacking sufficient 3D data, these methods struggle to fully utilize pretrained vision-language models for 3D robotic scene reasoning. Alternatively, some methods leverage 2D pre-trained models for 3D understanding. Works like [28, 25, 26] project 3D point clouds into multi-view images which are then encoded by 2D model, resulting in spatial information loss during 3D-to-2D transformation. Others [29, 30, 59] process 2D images via 2D visual encoders and lift features into a 3D latent space with depth map, which still fails to directly perceive raw 3D geometric information. Moreover, these approaches map scenes directly to actions, overlooking robot-environment relationships. To address this, we propose 3DS-VLA, a framework that enables 2D pre-trained VLMs to explicitly integrate 2D and 3D observation and reason about 3D spatial constraints.

3 Method

3.1 Task Formulation and Model Architecture

Given a dataset $\mathcal{D} = \{\tau_1, \ldots, \tau_N\}$ of N expert demonstrations, each demonstration τ is paired with a task description l and consists of visual observations $\mathcal{O} = \{o_1, \ldots, o_T\}$, robot state $\mathcal{R} = \{r_1, \ldots, r_T\}$, and actions $\mathcal{A} = \{a_1, \ldots, a_T\}$ over T frames. The robot state r and action a are defined by the end-effector position $x \in \mathbb{R}^3$, rotation quaternion $\theta \in \mathbb{R}^4$, and gripper status $g \in \mathbb{R}^1$. Following Rekep [60, 7, 61], we extract 3D keypoints $\mathcal{K} = \{k_1, \ldots, k_T\}$ using external models. The objective of policy model π is to learn action generation in SE(3) space: $\pi : (o_t, l, k_t, r_t) \rightarrow \hat{a}_{t+1}$. It takes visual inputs $o_t = \{i_t, p_t\}$, where i_t is the image and p_t is the point cloud, while language l, keypoints k_t, and robot state r_t are provided as structured textual inputs. The model supports the output of 7 or 14-DoF end-effector pose for single or dual arms and generates the predicted action $\hat{a}_{t+1}$ autoregressively, supervised by the ground-truth action a_{t+1} under cross-entropy loss. Following previous autoregressive-based VLA methods [39, 10, 41], the output of the end-effector position x and rotation quaternion θ is discretized into 200 bins as integer values. For the open status g, we use a binary representation, with 0 for closed and 1 for open.

As shown in Fig. 2, we enable parameter-efficient fine-tuning (PEFT) to adapt a pretrained vision-language model (e.g., LLaMA-Adapter [62]) into a policy model. The model π consists of a 2D visual encoder, LLM (LLaMA) [63], a cross-modality projection module [62], and LoRA adapters [64]. LoRA adapters are inserted into multiple linear layers of the vision encoder and LLM. During imitation learning, we fine-tune only the LoRA adapters and projection module, freezing all other pretrained 2D VLM parameters. To enhance 3D spatial observation (see Sec. 3.2), 2D images and 3D point clouds are first tokenized and encoded using pretrained 2D positional embeddings (PEa), then fused and processed by the shared visual encoder, CLIP[65]. A cross-modality projection module then maps visual tokens into LLaMA's word embedding space. To facilitate spatial-temporal reasoning (see Sec. 3.3), we formulate them as language condition to LLaMA, which then generates SE(3) actions conditioned on both visual and language input. As shown in Appendix. 7.1, with flexible autoregressive prediction, our model supports both single- and dual-arm tasks without architectural modifications.

3.2 3D Spatial Observation

Motivation. Current 3D imitation learning policies fall into two categories. Some [15, 16, 17, 18, 19, 14] use 3D visual encoders to encode 3D information but struggle to generalize due to limited large-scale 3D training data. Others leverage 2D pretrained models, either by projecting 3D data into multi-view images [25, 26, 27], causing spatial information loss, or by lifting 2D features into 3D latent spaces [28, 29, 57, 30], which cannot directly process 3D point clouds. To address these limitations, we aim to leverage the *pretrained* 2D visual encoder of the VLM to *directly* encode 3D point clouds, harnessing large-scale pretrained knowledge to enhance robotic 3D spatial reasoning.

2D and 3D Tokenizer. For 2D image, we follow CLIP and use its pretrained 2D tokenizer, which partitions the image into patches, flattens them, and projects them into 2D tokens $\{T_{2D}^j\}_{j=1}^n$, where n is the number of tokens and each token has 1024 channels. For 3D input, we generate a single-view point cloud with P points (e.g., $P = 2048$) using the depth map and camera parameters. To avoid introducing computational overhead, we introduce a non-parametric 3D tokenizer to transform the low-dimensional point cloud into high-dimensional 3D tokens. Specifically, we first apply Farthest Point Sampling (FPS) [23] to downsample points, then use k-Nearest Neighbors (kNN) to group k nearest neighbors (e.g., $k = 16$) for each center point. Each center and its neighbors are concatenated and max-pooled for local aggregation [66, 67]. After three iterations of this process, we obtain high-dimensional 3D tokens $\{T_{3D}^i\}_{i=1}^n$, matching the number and channels of 2D tokens.

2D-to-3D Positional Alignment. For 2D tokens, we follow CLIP and use its original 2D positional embeddings (PEs) $\{PE_{2D}^j\}_{j=1}^n$ to encode, preserving pretrained model's spatial reasoning ability. For 3D tokens, naive solutions are to either introduce newly initialized 3D PEs or directly reuse the pretrained 2D PEs $\{PE_{2D}^j\}_{j=1}^n$ to encode. However, the former lacks CLIP's pretrained knowledge, hindering its spatial understanding, while the latter leads to semantic misalignment, as the 3D token and the 2D positional embedding at the same index refer to different spatial regions. Therefore, we propose a 2D-to-3D positional alignment mechanism that allows the original 2D PEs, which are interpretable to pretrained models, to encode semantically aligned 2D and 3D tokens. This design is motivated by the fact that PEs serve as the only positional indicators in Transformer models due to the permutation-invariant nature of self-attention, which treats tokens identically regardless of order [68]. Specifically, since each 3D token T_{3D}^i is aggregated from a set of 3D points, we first unproject its center point to 2D image coordinates using the camera parameters. We then identify the corresponding 2D image patch onto which it projects and align it with the associated 2D token T_{2D}^j. As CLIP uses PE_{2D}^j to encode T_{2D}^j, we assign the same PE_{2D}^j to encode 3D token T_{3D}^i, as both tokens originate from the same image region and thus share consistent spatial and semantic context. The 2D PEs eocoded 2D tokens and 3D tokens are concatenated and then processed by the 2D vision encoder. By fine-tuning CLIP with LoRA on such spatially aligned representations, 3DS-VLA enables leveraging large-scale pretrained 2D VLM knowledge for reasoning both 2D and 3D information effectively without introducing additional computational costs.

3.3 3D Spatial Constraint

Motivation. Previous VLA models [11, 41, 10, 47] map observations to end-effector poses, but often overlook the understanding of constraints that govern the interaction between the robot and its environment. These relationships are shaped by both spatial and temporal constraints [60, 14, 69]. For instance, in Fig. 2, for *"pouring from the bottle into the cup"*, after grasping the bottle, the agent must determine *where* to pour (near the cup) and *when* to pour (when slightly above the cup). Therefore, we aim to incorporate these constraints during imitation learning, facilitating spatial-temporal reasoning for pose prediction.

3D Keypoint Extraction. Keypoint-based representations for object affordances enable adaptation to various robots and tasks, including single- and dual-arm configurations [60]. To model spatial constraints, we use task-specific 3D keypoints corresponding to scene entities. Following Rekep [7, 70], we extract keypoints by inputting an RGB image and object phrase described in task description into a vision foundation model (Grounded SAM [71]) to obtain object masks. We select the center point on object mask and project it into 3D world coordinates as object-level affordance information. Keypoint sequences follow their order in the language instruction. As shown in Fig. 2, we obtain *keypoint1-bottle* and *keypoint2-cup*, while *keypoint0* represents the current end-effector position.

3D Constraints Formulation. After generating 3D keypoints, instead of directly using them as task goals [60], we propose a text-based formulation to integrate these constraints into the VLA model as input conditions. We sequentially input the next keypoint to guide the model on *where* to act. For instance, in Fig. 2, at the initial state, the robot follows *keypoint1-bottle* as the spatial constraint. Once within a sufficient proximity (e.g., 5cm) of *keypoint1-bottle*, it automatically receives *keypoint2-cup* as next target. To understand *when* to act, we define temporal constraints as the relationship between the current robot state (e.g., *keypoint0*) and the keypoint spatial constraint (e.g., *keypoint1*), incorporating them into the language input. Therefore, by introducing spatial and temporal constraints as dynamic affordance conditions, 3DS-VLA integrates spatiotemporal context into the imitation learning process and leverages the reasoning capability of large language models (LLMs) to determine where and when to interact. Furthermore, to ensure that 3DS-VLA is robust to noises in 3D keypoint positions, we randomly add noise to the selected keypoints in each training sample, encouraging the model to tolerate keypoint perturbations. In Sec. 4.3, We empirically show how well the model can handle keypoint noise during inference.

4 Experiment

4.1 Training and Inference

Following previous settings [25, 15], we assume that the model should predict an action specified by a target end-effector pose and gripper state in the next keyframe. The keyframes represent important or bottleneck steps of the gripper during task execution, such as a pre-pick, grasp, or place pose. We simultaneously train on demonstrations from the single-arm simulator RLBench [37, 72] and the dual-arm simulator RLBench2 [38]. The fine-tuning stage trains on 2,400 demonstrations and runs for 10 epochs, taking approximately 8 hours on an NVIDIA RTX A100 GPU, achieving a 5Hz inference speed (similar to VLA baselines), excluding the keypoint extraction pre-processing since it is performed prior to execution.

4.2 Evaluation on RLBench and Dual-arm RLBench

Setup. We adopt 21 tasks from single-arm RLBench [37, 72] and 5 tasks of the same coordinate type from dual-arm RLBench [38], featuring variations such as different colors or object pose. The input RGB-D images, with a resolution of 336 $\times$ 336, are captured by a single camera mounted at the front of the robot. To ensure that objects are visible from this fixed front-view camera, we adjust

Table 1: Single-Arm Multi-Task Performance on RLBench of 21 tasks.

Models	Avg. ↑	Stack Blocks	Open Microwave	USB out Computer	Rubbish in Bin	Close Laptop	Toilet Down	Beat Buzz	Water Plant	Wine at Rack	Sweep Dustpan
RVT2	0.62±0.47	0.52	**0.84**	**0.96**	0.80	0.92	0.92	**0.58**	0.24	0.72	0.20
3DA	0.60±0.35	0.56	**0.84**	**0.96**	0.82	0.23	**0.96**	0.50	**0.42**	0.54	0.35
3D-L	0.64±0.14	0.56	0.60	0.88	**0.96**	**0.96**	**0.96**	0.32	0.16	0.44	0.28
DP3	0.64±0.22	0.56	0.72	0.90	0.88	**0.96**	0.88	0.44	0.39	0.56	0.32
OpenVLA	0.43±0.41	0.12	0.04	0.32	0.52	0.88	**0.96**	0.08	0.12	0.32	0.36
CogACT	0.55±0.36	0.28	0.20	0.54	0.82	0.92	0.88	0.24	0.32	0.56	**0.60**
Ours	**0.66**±0.32	**0.60**	0.68	0.88	0.88	**0.96**	**0.96**	0.52	0.36	**0.88**	0.16

Models	Close Box	Lamp On	Unplug Charger	Close Fridge	Press Switch	Phone on Base	Umbralla Out	Change Clock	Frame off Hanger	Knife on Board	Weighing Scales
RVT2	0.84	0.92	0.00	**1.0**	0.52	0.48	0.56	0.08	0.68	0.56	0.64
3DA	0.62	0.65	0.33	0.92	**0.62**	**0.96**	0.71	0.22	0.35	**0.69**	0.60
3D-L	**0.92**	0.92	**0.56**	0.96	0.60	**0.96**	0.78	0.40	**0.76**	0.60	0.04
DP3	**0.92**	0.84	**0.56**	0.92	0.56	0.52	0.72	0.32	0.62	0.52	0.42
OpenVLA	0.44	0.26	0.08	0.88	0.56	0.48	0.56	0.24	0.56	0.08	0.32
CogACT	0.60	0.56	0.12	0.88	0.64	0.80	0.56	**0.48**	0.60	0.52	0.44
Ours	0.88	**0.96**	**0.56**	0.92	0.56	0.60	**0.80**	0.32	0.52	0.63	**0.84**

the workspace bounding box accordingly. We train and test our methods using the same dataset as the baselines, with 100 demonstrations per task for training and 25 demonstrations for testing.

Baselines. In the single-arm setting, we compare our method against two types of baselines: 3D imitation learning and VLA methods, both using only the single front view. The first category includes: 1) **RVT2** (RSS 2024) [26]: A transformer-based model on RLBench that encodes virtually projected images for 3D object manipulation. 2) **3D Diffusion Policy (DP3)** (RSS 2024) [18]: A visual imitation learning method leveraging diffusion-based policies for expressive 3D representations. 3) **3D Diffuser Actor (3DA)** (CoRL 2024) [29]: A model that processes 2D images through a visual encoder, lifts extracted features into a 3D latent space, and integrates diffusion-based 3D scene representations for action prediction. 4) **3D-Lotus (3D-L)** (ICRA 2025) [73]: The latest accepted SOTA on RLBench before the submission deadline, utilizing foundation models for task planning and object grounding. The second category includes: 5) **OpenVLA** [11]: A VLA model that takes an image and task description as input, using an MLP head for action generation. 6) **CogAct** [47]: A SOTA 7B VLA model that separates cognitive and action capabilities while employing diffusion transformers for action generation. In the dual-arm setting, as the above baselines do not support dual-arm manipulation, we follow PerAct2 [38] and compare against: 1) **RVT-LF**: Two Robotic View Transformer (RVT) [25] models in a leader-follower architecture. 2) **PerAct-LF**: Two Perceiver Actor networks [15] in a leader-follower architecture. 3) **PerAct2** [38]: A single bimanual Perceiver Actor network. Following the leader-follower architecture implementation [38], the output of one network serves as input to the other, with both actions executed sequentially.

Table 2: Dual-Arm Multi-Task Performance on RL-Bench2 of 5 tasks

Bimanual Task	RVT-LF	Peract-LF	Peract2	Ours
Lift ball	0.17	0.40	0.50	**0.71**
Lift tray	0.06	0.14	0.01	**0.22**
Straightn rope	0.03	0.21	0.24	**0.52**
Pick laptop	0.03	0.11	0.12	**0.14**
Push box	0.52	0.57	0.07	**0.70**
AVG.	0.18	0.29	0.22	**0.46** ±0.30

Table 3: The effectiveness of each proposed component.

Row ID	3D Spatial Constraint	3D Tokens	Aligned PEs	AVG. Score
1	✗	✗	✗	0.41
2	✓	✗	✗	0.62
3	✓	✓	✗	0.60
4	✓	✓	✓	**0.66**

Analysis. As shown in Tab. 1, in the single-arm setting, our method surpasses all baselines by at least 4% average success rate. Compared with 3D imitation learning methods, 3DS-VLA leverages the pretrained knowledge from 2D VLMs and boosts its 3D awareness, enabling effective multi-task learning. Compared with 2D VLA methods, we observe frequent failures during the critical final stage of 3D contact. This stems from their reliance on single-view 2D images without explicit 3D geometric understanding, which is essential for precise action prediction. In contrast, our approach improves the robot's spatial understanding by enhancing 3D observation and incorporating 3D spatial constraints. Furthermore, as shown in Tab. 2, in the dual-arm setting, our method outperforms all baselines by a significant margin. This demonstrates our model's robustness and potential for generalization across different control modalities, enabling strong spatial reasoning for dual-arm collaboration and coordinated actions. Additionally, we perform an extra experiment where we first fine-tune the pretrained VLM on the OXE dataset [74], which only takes 2D images as input, and then continue finetuning on single-arm simulated dataset, achieving 68% success rate. Since our method introduces 3D observations in a non-parametric manner, it can fully leverage existing 2D datasets for pretraining.

4.3 Ablation Study

Does each component work? Starting with Row1 in Tab. 3, we use a model with the same architecture as ours but only take the image, robot state, and task description as input, directly outputting actions. In Row2 (w/ 3D spatial constraint), incorporating keypoint constraints as structured language input establishes spatial-temporal associations, improving performance by 19%. In Row3 (w/o Aligned PEs), 3D tokens are extracted from the point cloud but encoded using the original order of 2D PEs instead of the proposed aligned PEs, leading to a 2% drop. This demonstrates that misaligned positional embeddings disrupt spatial reasoning due to inconsistencies in 3D spatial representation. In Row4 (3DS-VLA), using aligned 2D PEs to encode 3D tokens enhances performance to 0.66, validating the effectiveness of our 2D-to-3D positional alignment mechanism.

The robustness of 3DS-VLA when handling noise. During inference, we randomly add noise to the input keypoint constraints, with a maximum deviation of 10 cm in x, y, and z directions, including the circumstance that the keypoints are on the cup, instead of the handle that the robot should grasp, or on the table near the target object. This results in an average success score of 0.63, showing our model can tolerate noise to some extent.

The effectiveness of 3DS-VLA when handling both single- and dual-arm tasks. We compare Ours with Ours-s, where Ours is trained on multi-embodiment tasks (single-arm and dual-arm), while Ours-s are trained exclusively on single-arm tasks. Both Ours and Ours-s achieve the same average success rate of 0.66 on single-arm tasks, demonstrating that our model can effectively handle different embodiments within a unified training pipeline, without requiring architectural modifications.

4.4 Evaluation in Real-world

Setup. Our method is evaluated across 10 tasks on the Franka Research 3 (FR3) robot with a 3D-printed UMI gripper [75]. We use a RealSense L515 camera to obtain real-world visual observations from the front view. Expert demonstrations are collected via human teleoperation. For each task, 50 demonstrations are collected in diverse object poses. We fine-tune the model with 10 epochs using pretrained weights obtained from simulation training. For each task, we train an agent and evaluate it over 10 trials with diverse object poses. The success rate is used as the evaluation metric. The green light on the Franka Research 3 indicates that the robot is in execution mode with Franka Control Interface (FCI) enabled.

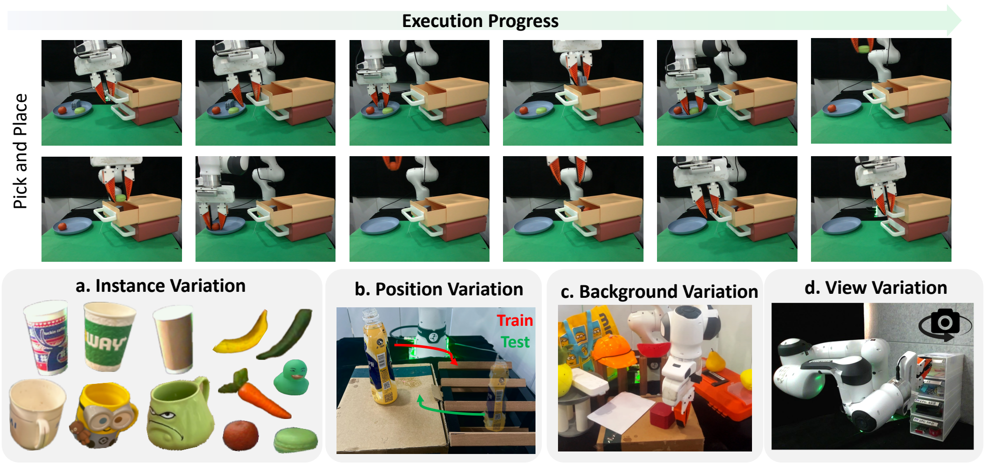


Figure 3: Demonstrations of execution process and four types of generalization settings.

Table 4: We compare 3DS-VLA with baselines on 10 real-world tasks and evaluate its robustness across test settings that vary from the training dataset domain. * denotes long-horizon tasks.

Models	Avg. Success ↑	Stack Cup	Pour Water	Pick Place*	Stack Block*	Water Plants	Bottle at Rack	Slide Box	Unplug Charger	Wipe Table	Open Drawer
DP3	0.49	**0.70**	**0.50**	**0.50**	0.30	0.40	0.50	0.50	0.40	0.50	**0.60**
CogAct	0.50	0.30	0.40	0.35	0.30	0.60	0.40	**0.60**	**0.70**	**0.60**	0.50
Rekep	0.41	0.40	**0.50**	0.30	0.10	0.60	0.60	0.50	0.40	0.55	0.10
Ours	**0.54**	0.40	**0.50**	0.40	**0.40**	**0.70**	**0.80**	**0.60**	0.40	**0.60**	**0.60**
Ablation		*a.Instance Var.*			*b.Position Var.*			*c.Background Var.*		*d.View Var.*	
Ours	-	0.30	0.40	0.40	0.30	0.40	0.50	0.60	0.40	0.50	0.60

Quantitative results. We compare our method with the two best-performing methods in simulator experiments: the 3D imitation learning method 3D Diffusion Policy (DP3) and the VLA-based method CogAct. Additionally, we include the baseline Rekep [60], which leverages constraints on semantic keypoints to specify desired relationships between robot arms, object parts, and other agents in a training-free manner. Since Rekep's open-source implementation is not well-suited for RLBench, we compare it with only in real-world experiments. As shown in Table 4, our method outperforms all baselines, demonstrating superior interaction in 3D environments. Although both our method and Rekep use external visual models for keypoint extraction, Rekep treats semantic keypoints as subgoal objective for trajectory planning. This makes the pipeline prone to failure if the underlying models are inaccurate—for example, if GroundingDINO [71] misses critical keypoints on the cup handle that needs to be grasped, or if GPT-4 overlooks essential sub-steps, such as forgetting to lift a block before placing it, resulting in collisions.

Generalization. 3DS-VLA demonstrates strong generalization abilities in real world. We categorize these generalization abilities into 4 aspects in Fig. 3 and present quantitative results in Tab. 4: **1) Instance variation**: We evaluate across a diverse set of unseen instances that differ in color, size, and appearance compared to the training data. We demonstrate that 3DS-VLA effectively handles a wide range of everyday objects across the tasks listed in Group-A of Tab. 4, which we attribute to the inherent geometric richness and spatial continuity provided by point clouds. **2) Position Variation**: We shift the relative position of objects to evaluate the model's ability to handle positional changes, thereby testing its capability in trajectory prediction. For example, during training, the model learns to stack the block on the left onto the block on the right, while in testing, the task is to stack the block on the right onto the block on the left. As shown in group-b of Tab. 4, thanks to the spatial constraints that encode the relationship between the robot and its environment, our model is capable of adapting to such variations despite the change in relative positions. **3) Background Variation:** We demonstrate that 3DS-VLA is capable of handling tasks in complex and cluttered environments in Tab. 4 group-c. To illustrate this, we test the "slide box" and "unplug charger" tasks with randomly set backgrounds, without additional training. Remarkably, the model achieves similar accuracy as it does in clean background settings. Since we establish associations between the robot and its environment through structured text input, our model learns to focus on task-relevant objects while disregarding irrelevant background disturbances. **4) View Variation:** In group-d of Tab. 4, we test our model from different camera view angles and observe that it is capable of handling significant changes in view variation.

Please refer to Appendix for more details: Section 7.2 for visualization of tasks in RLBench and real world and Section 7.3 for discussion of failure cases.

5 Conclusion

We propose 3DS-VLA, which enhances pretrained 2D vision-language models (VLMs) with comprehensive 3D awareness and the ability to predict end-effector poses. Our approach involves enriching 3D spatial observations to enable a shared 2D visual encoder to process both 3D and 2D data while establishing 3D spatial constraints that encode the spatial-temporal relationships between the robot and its environment for reliable policy learning. Extensive experiments, both in simulation and in the real world, demonstrate the promising performance of 3DS-VLA.

6 Limitations

First, 3DS-VLA requires camera calibration and depth information, as is the case with all 3D policies. Secondly, similar to most VLA methods, it tends to be slower in inference on average compared to policies that do not utilize large language model. However, this can be mitigated by recent techniques that increase the output action chunks, or applying mechanisms like KV-cache [76] to speed up transformer-based models. In addition, the current keypoint sequence is determined using a heuristic approach. For more complex tasks in the future, planning models such as GPT [77] can be leveraged to determine the task-specific order of keypoints.

Acknowledge

This work was supported by the National Youth Talent Support Program (8200800081), National Natural Science Foundation of China (62376006), and National Natural Science Foundation of China (62476011).

References

[1] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[2] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[3] J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[4] X. Li, J. Xu, M. Zhang, J. Liu, Y. Shen, I. Ponomarenko, J. Xu, L. Heng, S. Huang, S. Zhang, et al. Object-centric prompt-driven vision-language-action model for robotic manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 27638–27648, 2025.

[5] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[6] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[7] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[8] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[9] X. Li, M. Liu, H. Zhang, C. Yu, J. Xu, H. Wu, C. Cheang, Y. Jing, W. Zhang, H. Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023.

[10] X. Li, M. Zhang, Y. Geng, H. Geng, Y. Long, Y. Shen, R. Zhang, J. Liu, and H. Dong. Manipllm: Embodied multimodal large language model for object-centric robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18061–18070, 2024.

[11] J. Liu, M. Liu, Z. Wang, L. Lee, K. Zhou, P. An, S. Yang, R. Zhang, Y. Guo, and S. Zhang. Robomamba: Multimodal state space model for efficient robot reasoning and manipulation. *arXiv preprint arXiv:2406.04339*, 2024.

[12] H. Zhu, Y. Wang, D. Huang, W. Ye, W. Ouyang, and T. He. Point cloud matters: Rethinking the impact of different observation spaces on robot learning. *arXiv preprint arXiv:2402.02500*, 2024.

[13] S. Chen, R. Garcia, I. Laptev, and C. Schmid. Sugar: Pre-training 3d visual representations for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18049–18060, 2024.

[14] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

[15] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[16] S. Chen, R. Garcia, C. Schmid, and I. Laptev. Polarnet: 3d point clouds for language-guided robotic manipulation. *arXiv preprint arXiv:2309.15596*, 2023.

[17] M. Liu, X. Li, Z. Ling, Y. Li, and H. Su. Frame mining: a free lunch for learning robotic manipulation from 3d point clouds. *arXiv preprint arXiv:2210.07442*, 2022.

[18] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy. *arXiv preprint arXiv:2403.03954*, 2024.

[19] J. Huang, S. Yong, X. Ma, X. Linghu, P. Li, Y. Wang, Q. Li, S.-C. Zhu, B. Jia, and S. Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023.

[20] X. Li, J. Liu, N. Han, L. Heng, Y. Guo, H. Dong, and Y. Liu. 3dwg: 3d weakly supervised visual grounding via category and instance-level alignment. *arXiv preprint arXiv:2505.01809*, 2025.

[21] Y. Jia, J. Liu, S. Chen, C. Gu, Z. Wang, L. Luo, X. Li, P. Wang, Z. Wang, R. Zhang, et al. Lift3d policy: Lifting 2d foundation models for robust 3d robotic manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17347–17358, 2025.

[22] M. Zhang, X. Li, J. Xu, K. Zhou, H. Bae, Y. Shen, C. Xiong, and H. Dong. Sr3d: Unleashing single-view 3d reconstruction for transparent and specular object grasping. *arXiv preprint arXiv:2505.24305*, 2025.

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[24] J. D. M.-W. C. Kenton and L. K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota, 2019.

[25] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.

[26] A. Goyal, V. Blukis, J. Xu, Y. Guo, Y.-W. Chao, and D. Fox. Rvt-2: Learning precise manipulation from few demonstrations. *arXiv preprint arXiv:2406.08545*, 2024.

[27] W. Wang, Y. Lei, S. Jin, G. D. Hager, and L. Zhang. Vihe: Virtual in-hand eye transformer for 3d robotic manipulation. *arXiv preprint arXiv:2403.11461*, 2024.

[28] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[29] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[30] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR, 2022.

[31] K. Zheng, X. Chen, O. C. Jenkins, and X. Wang. Vlmbench: A compositional benchmark for vision-and-language manipulation. *Advances in Neural Information Processing Systems*, 35: 665–678, 2022.

[32] X. Li, Y. Wang, Y. Shen, P. Iaroslav, H. Lu, Q. Wang, B. An, J. Liu, and H. Dong. Imagemanip: Image-based robotic manipulation with affordance-guided next view selection. *arXiv preprint arXiv:2310.09069*, 2023.

[33] S. Huang, I. Ponomarenko, Z. Jiang, X. Li, X. Hu, P. Gao, H. Li, and H. Dong. Manipvqa: Injecting robotic affordance and physically grounded information into multi-modal large language models. *arXiv preprint arXiv:2403.11289*, 2024.

[34] R. Xu, Y. Shen, X. Li, R. Wu, and H. Dong. Naturalvlm: Leveraging fine-grained natural language for affordance-guided visual manipulation. *arXiv preprint arXiv:2403.08355*, 2024.

[35] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021.

[36] Y. Ju, K. Hu, G. Zhang, G. Zhang, M. Jiang, and H. Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2025.

[37] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[38] M. Grotz, M. Shridhar, Y.-W. Chao, T. Asfour, and D. Fox. Peract2: Benchmarking and learning for robotic bimanual manipulation tasks. In *CoRL 2024 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*, 2024.

[39] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[40] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[41] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[42] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[43] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[44] H. Chen, J. Liu, C. Gu, Z. Liu, R. Zhang, X. Li, X. He, Y. Guo, C.-W. Fu, S. Zhang, et al. Fast-in-slow: A dual-system foundation model unifying fast manipulation within slow reasoning. *arXiv preprint arXiv:2506.01953*, 2025.

[45] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[46] Z. Hou, T. Zhang, Y. Xiong, H. Pu, C. Zhao, R. Tong, Y. Qiao, J. Dai, and Y. Chen. Diffusion transformer policy. *arXiv preprint arXiv:2410.15959*, 2024.

[47] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[48] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[49] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *Advances in Neural Information Processing Systems*, 36:25081–25094, 2023.

[50] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

[51] Y. Chen, W. Cui, Y. Chen, M. Tan, X. Zhang, D. Zhao, and H. Wang. Robogpt: an intelligent agent of making embodied long-term decisions for daily instruction tasks. *arXiv preprint arXiv:2311.15649*, 2023.

[52] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on Robot Learning*, pages 284–301. PMLR, 2023.

[53] Y. Jia, J. Liu, S. Chen, C. Gu, Z. Wang, L. Luo, L. Lee, P. Wang, Z. Wang, R. Zhang, et al. Lift3d foundation policy: Lifting 2d large-scale pretrained models for robust 3d robotic manipulation. *arXiv preprint arXiv:2411.18623*, 2024.

[54] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 2023.

[55] W. Yuan, A. Murali, A. Mousavian, and D. Fox. M2t2: Multi-task masked transformer for object-centric pick and place. *arXiv preprint arXiv:2311.00926*, 2023.

[56] S. James and P. Abbeel. Coarse-to-fine q-attention with learned path ranking. *arXiv preprint arXiv:2204.01571*, 2022.

[57] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[58] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[59] A. Simeonov, A. Goyal, L. Manuelli, L. Yen-Chen, A. Sarmiento, A. Rodriguez, P. Agrawal, and D. Fox. Shelving, stacking, hanging: Relational pose diffusion for multi-modal rearrangement. *arXiv preprint arXiv:2307.04751*, 2023.

[60] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[61] F. Liu, K. Fang, P. Abbeel, and S. Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[62] R. Zhang, J. Han, C. Liu, P. Gao, A. Zhou, X. Hu, S. Yan, P. Lu, H. Li, and Y. Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[63] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[64] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[65] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[66] R. Zhang, L. Wang, Z. Guo, Y. Wang, P. Gao, H. Li, and J. Shi. Parameter is not all you need: Starting from non-parametric networks for 3d point cloud analysis. *arXiv preprint arXiv:2303.08134*, 2023.

[67] X. Zhu, R. Zhang, B. He, Z. Guo, J. Liu, H. Xiao, C. Fu, H. Dong, and P. Gao. No time to train: Empowering non-parametric networks for few-shot 3d scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3838–3847, 2024.

[68] Y. Tang, R. Zhang, J. Liu, Z. Guo, B. Zhao, Z. Wang, P. Gao, H. Li, D. Wang, and X. Li. Any2point: Empowering any-modality large models for efficient 3d understanding. In *European Conference on Computer Vision*, pages 456–473. Springer, 2025.

[69] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

[70] S. Wu, Y. Zhu, Y. Huang, K. Zhu, J. Gu, J. Yu, Y. Shi, and J. Wang. Afforddp: Generalizable diffusion policy with transferable affordance. *arXiv preprint arXiv:2412.03142*, 2024.

[71] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[72] E. Rohmer, S. P. Singh, and M. 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.

[73] R. Garcia, S. Chen, and C. Schmid. Towards generalizable vision-language robotic manipulation: A benchmark and llm-guided 3d policy. *arXiv preprint arXiv:2410.01345*, 2024.

[74] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[75] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[76] S. Ge, Y. Zhang, L. Liu, M. Zhang, J. Han, and J. Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.

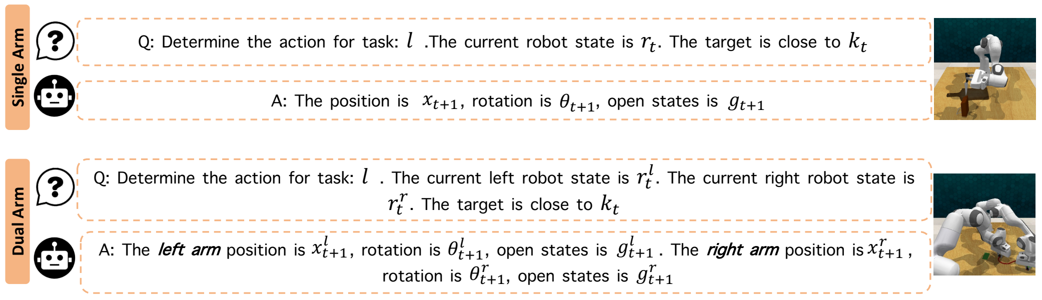
[77] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[78] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2025.

7 Appendix

7.1 Input and Output Pairs

We illustrate the input language prompt and answer format in Fig. 4. Single-arm and dual-arm tasks share the same format, with the latter including additional information for the second arm. This allows a single model to handle both tasks without any architectural modifications.



Q: Determine the action for task: l .The current robot state is r_t. The target is close to k_t

A: The position is x_{t+1}, rotation is θ_{t+1}, open states is g_{t+1}

Q: Determine the action for task: l . The current left robot state is r_t^l. The current right robot state is r_t^r. The target is close to k_t

A: The ***left arm*** position is x_{t+1}^l, rotation is θ_{t+1}^l, open states is g_{t+1}^l . The ***right arm*** position is x_{t+1}^r , rotation is θ_{t+1}^r, open states is g_{t+1}^r

Figure 4: **Question-Answer Pair.** We encode the current robot pose and target keypoint constraints as 3D spatio-temporal constraints in the model's language input. Both single-arm and dual-arm settings share this formulation, differing only in prompt and supervision.

7.2 RLBench and Real-world Tasks Overview

The simulation tasks in Fig.5 follow previous works, while we provide a more detailed explanation of the real-world tasks and their success conditions.

Qualitative results. As shown in Fig.6, we visualize the manipulation process of our method on real-world tasks. Whether the task demands precise positioning, complex rotation, or semantic understanding, 3DS-VLA can accurately predict the sequence of 7-DoF end-effector poses. For example, in the *Stack Cup* task, our method first ensures an exact grasping position at the edge of the cup, then precisely lifts and moves it above the target cup, eliminating any positional deviation. The real-world execution video can be found in the supplementary material.

1. Pour water: This task requires the robot to first grasp the bottle, then rotate it to a position slightly above the cup, and tilt it to perform the pouring action.

2. Slide box: The robot needs to slide the box using the side of the end-effector, moving it to the designated white space area.

3. Unplug charger: The robot needs to grasp the charger and then pull it out to a certain height without slipping.

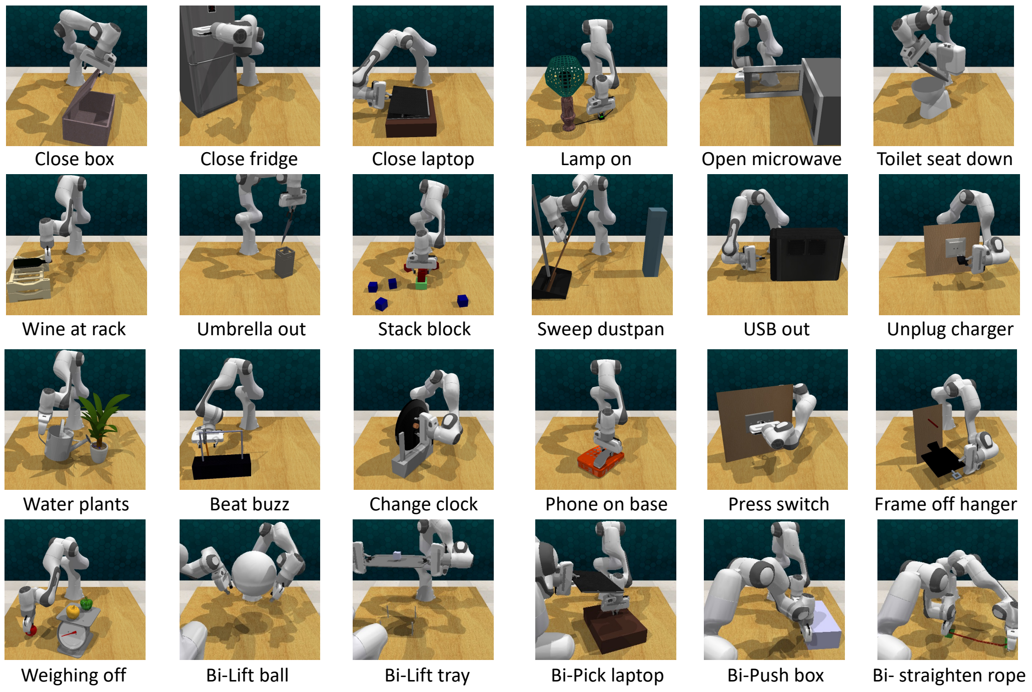


Figure 5: **Visualization of simulation tasks.** We conduct on both single-arm and dual-arm simulation tasks.

4. Bottle at rack: The robot needs to grasp the bottle, ensuring that the grasp pose is not a simple top-down approach, as this would hinder the completion of subsequent tasks. Considering joint limitations, the first grasp must be in a backward direction, followed by rotation to place the bottle on the rack.

5. Stack cup: The robot needs to grasp a cup and place it on top of another one. Since the cup fits inside the other, this task requires higher precision in rotation prediction compared to the previous one.

6. Water plant: The robot needs to grasp the handle of the object, a specific part, and then pour water into the plant.

7. Wipe desk: The robot needs to pick up a tissue and use it to wipe the stain on the table, ensuring the stain is completely covered.

8. Open drawer: The robot needs to open the top drawer by rotating the end-effector, rather than using a top-down pose. Due to the position of the robot's base and the front camera's field of view, the drawer is placed at a sharp angle, requiring more accurate rotation prediction to open it.

9. Stack Block: The robot is required to stack a total of three blocks. It grasps one block at a time and places it on top of the previously stacked block.

10. Pick and place: The robot needs to open the drawer, sequentially pick up the charger, macaron, and tomato, place them into the drawer, and then close it.

7.3 Failure Cases

We categorize the failure cases into two types: 1) *Pose precision*, where the predicted end-effector poses are too imprecise to meet the success criteria. 2) *Keypoint generation*, where the external model fails to detect the object.

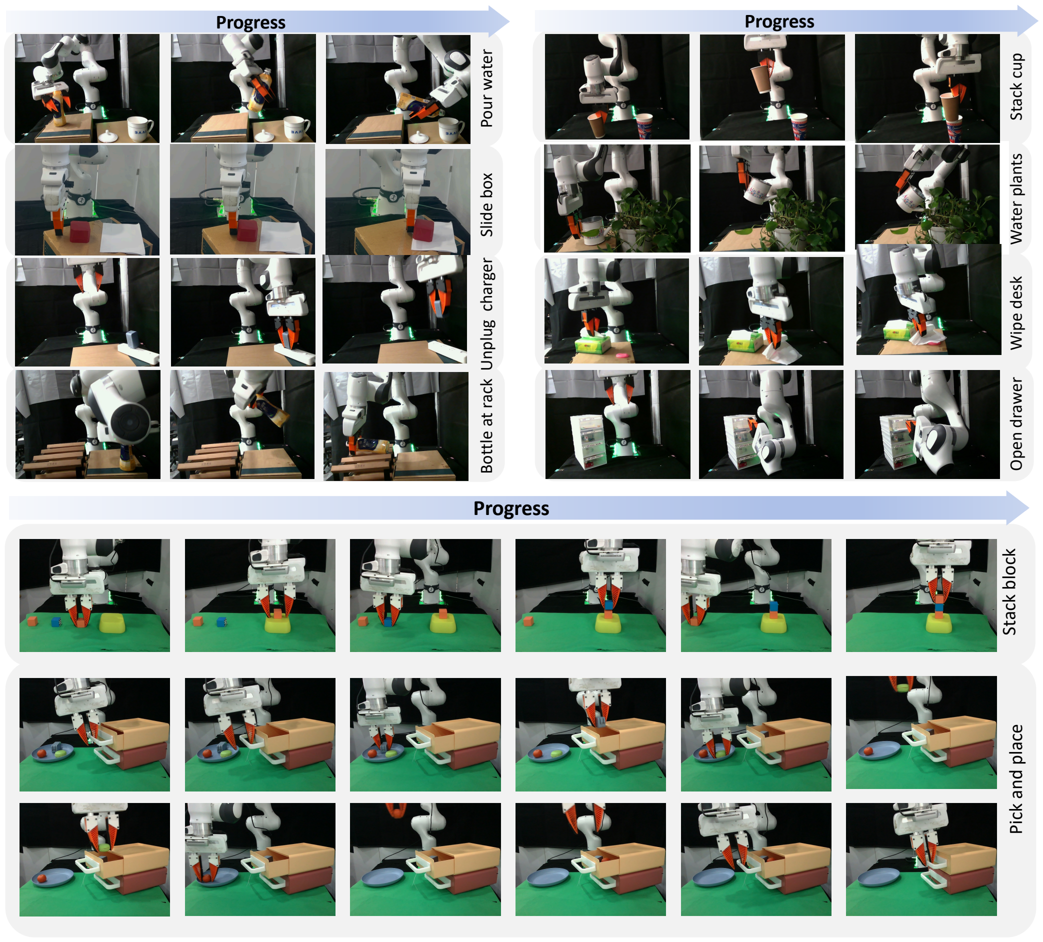


Figure 6: **Visualization of real-world tasks.** The tasks are shown in key-frame flow.

The primary failure mode is the imprecise prediction of end-effector poses. This is especially common in tasks that require high precision, such as stacking blocks, where the blocks must be placed stably on top of one another.

Incorrect keypoint generation is another significant failure mode, particularly in simulations. For instance, Grounding SAM [78] may fail to detect objects due to unrealistic rendering in the simulator. Additionally, the grounding results can be influenced by the presence of the robot arm under the front-view camera, as it is not a typical object encountered during the pretraining of Grounding SAM. Although the model can tolerate a certain degree of keypoint noise, severe errors—such as providing a keypoint on a banana when the intended target is an apple—can still lead to task failure.

Another common failure occurs in dual-arm setting, where collaboration between the two arms is required. Since the model predicts the actions of both arms based on the current object state and if one arm makes contact with the object, the object's state changes. As a result, the previously predicted action for the other arm may no longer be valid.