2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

# Toward Embodiment Equivariant Vision-Language-Action Policy

Anzhe Chen[1], Yifei Yang[1], Zhenjie Zhu[1], Kechun Xu[1], Zhongxiang Zhou[2], Rong Xiong[1,2], Yue Wang[1]

***Abstract*— Vision-language-action policies learn manipulation skills across tasks, environments and embodiments through large-scale pre-training. However, their ability to generalize to novel robot configurations remains limited. Most approaches emphasize model size, dataset scale and diversity while paying less attention to the design of action spaces. This leads to the configuration generalization problem, which requires costly adaptation. We address this challenge by formulating cross-embodiment pre-training as designing policies equivariant to embodiment configuration transformations. Building on this principle, we propose a framework that (i) establishes a embodiment equivariance theory for action space and policy design, (ii) introduces an action decoder that enforces configuration equivariance, and (iii) incorporates a geometry-aware network architecture to enhance embodiment-agnostic spatial reasoning. Extensive experiments in both simulation and real-world settings demonstrate that our approach improves pre-training effectiveness and enables efficient fine-tuning on novel robot embodiments. Our code is available at https://github.com/hhcaz/e2vla**

## I. Introduction

Vision-language-action (VLA) policies have attracted increasing attention in recent years. These policies are typically pre-trained on large-scale cross-embodiment datasets that span diverse tasks and environments, and then fine-tuned for specific embodiments and downstream tasks. While such pre-training has shown effectiveness, it often requires the target embodiment to share similar robot configurations with those seen during pre-training. Consequently, when faced with novel embodiments, the policy tends to fail, leading to considerable additional training costs for adaptation.

Ideally, a VLA policy should exhibit a certain degree of zero-shot generalization to unseen robot configurations (e.g., variations in the coordinate definitions of the robot base and end-effector) after pre-training. Such capability would enable efficient fine-tuning with minimal additional data and training. However, existing approaches [1], [2], [3], [4], [5] to VLA pre-training focus primarily on scaling up parameters, datasets and task diversity, while paying limited attention to the design of neural architectures and action spaces. leaving the configuration generalization problem unresolved. Without configuration generalization, the policy may tend to learn the embodiment-specific knowledge during pre-training, which hinders the sharing of task-level knowledge and reduces the overall effectiveness of cross-embodiment pre-training.

[1]Zhejiang University. [2]Zhejiang Humanoid Robot Innovation Center. Yue Wang is the corresponding author: wangyue@iipc.zju.edu.cn. This work was supported by the National Natural Science Foundation of China (Grant 62522317 and Grant U24A20128), the Zhejiang Provincial Natural Science Foundation of China (Grant LD25F030001), and the Research and Development Project of Zhejiang Province (Grant 2025C01205 (SD2)).

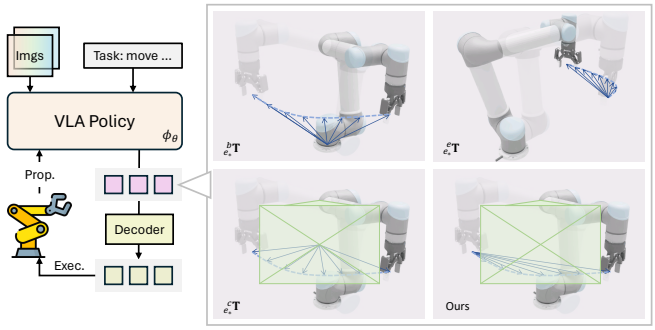


Fig. 1: Different action space design for VLA policy learning. We learn the relative end-effector motion projected in camera space.

A key requirement for cross-embodiment pre-training, therefore, is to prevent networks from hard-memorizing embodiment-specific details. This motivates the question: *Is there a framework to unifies action space design across embodiments to learn task-level knowledge but not embodiment-specific kinematics details?*

We draw inspiration from equivariance, where neural networks are endowed with symmetry properties by design rather than through learning from data. For example, $\mathrm{E}(2)$-equivariant CNN [6] generalizes this property to rotations and reflections in the Euclidean plane. It is applied in medical imaging, satellite imagery where objects may appear at arbitrary orientations. $\mathrm{SE}(3)$-Transformers [7] and Equivariant Graph Neural Networks [8] ensure equivariance under 3D rotations and translations, which are applied in protein structure prediction, molecular dynamics modeling and 3D perception tasks. Compared with data augmentation, equivariant models encode inductive biases more efficiently, leading to reduced model size and improved generalization to unseen transformations.

We hypothesize that the limited generalization ability of existing VLA policies stems from their use of joint-space actions or end-effector pose in base (Fig. 1), which makes the overall policy non-equivariant to embodiment configuration transformations and thus leads to overfitting to specific embodiments during pre-training. Therefore, we propose to formulate cross-embodiment learning as the design of policy equivariant to embodiment configuration transformations. Such a formulation enables systematic generalization to novel embodiments and makes pre-training more effective. Our main contributions are as follows:

- Policy equivariance theory that unifies action space design across embodiments;
- An action decoder that is equivariant to embodiment

configuration transformations;
- A network architecture that enhances geometric reasoning within the equivariant policy;
- Simulation and real-world validation of effective pre-training and efficient fine-tuning with our approach.

## II. Related Works

**VLA Policies.** Vision-language-action (VLA) policies represent a key step toward unifying perception, language understanding, and action execution within a single framework. Classic policies such as RT-1 [1], ACT [9], and Octo [10] adopt transformer-based architectures trained on diverse robot trajectories across multiple environments and tasks. More recent approaches leverage vision-language models (VLMs) pretrained on internet-scale data, adopting paradigms similar to large language models (LLMs). For instance, RT-2 [2] and OpenVLA [5] treat actions as tokens and employ next-token prediction to generate robot behaviors. While these methods improve generalization and instruction-following capabilities through enhanced language understanding and visual grounding, their inference speed remains insufficient for real-time robotic systems.

To address this, dual-system architectures have been proposed. RoboDual [11], Fis-VLA [12], and OpenHelix [13] decouple reasoning and control by running a slow reasoning module alongside a fast action expert, thereby combining the strengths of both. In addition, diffusion-based methods have been explored to better capture multi-modal human demonstrations. Diffusion Policy [14], together with flow-based action generation methods [3], [4], [15], [16], has become a dominant trend in recent VLA research.

Since robot action datasets remain relatively small compared to internet-scale vision-language corpora, several works have introduced auxiliary tasks to leverage large-scale image and video data. SuSIE [17] generates subgoal images through image editing, while the GR series [18], [19], [20] exploit video generation pre-training for manipulation. Other methods avoid explicit future image prediction and instead generate features, such as keypoint motions [21] or latent visual embeddings [22], to better utilize internet-scale supervision.

**Action Space.** Open-source robot datasets [23] contain heterogeneous action representations, posing challenges for cross-embodiment pre-training. RDT-1B [3] directly predicts high-dimensional raw action vectors containing mixed quantities such as joint positions, end-effector poses, and deltas. $\pi_0$ [4] learns joint-space or end-effector actions for most datasets, but a fixed index in the action vector may correspond to different physical meanings across robots (e.g., index-6 representing gripper width in a 6-DoF arm versus a wrist joint in a 7-DoF arm). DexVLA [15] addresses this by employing multiple action heads tailored for different embodiments.

Several works attempt to design unified action spaces for cross-embodiment learning. For example, [24] proposes egocentric actions applicable to both navigation and manipulation robots, but requires manual alignment across datasets to ensure consistency in action semantics. ATM [21] predicts pixel-level motions and employs an inverse dynamics model to map them to executable actions. UniVLA [22] predicts future latent features of observations and derive task-centric action representations with a latent action model. While effective, these approaches rely on over-parameterized image-space motion representations and require training an additional inverse dynamics model for each new embodiment.

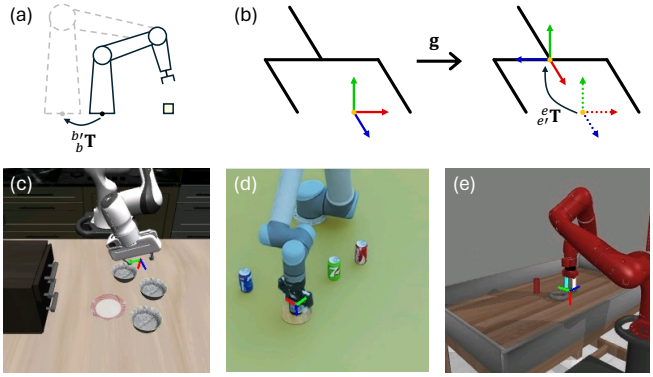


Fig. 2: Schematics of the pose coordinate definition and group transformation of embodiment base and end-effector. (a) Embodiment with different base definition should have similar relative pose trajectory between end-effector pose and the objects when completing similar tasks. (b) Different coordinate definitions with respect to grippers with similar topologies. Some define the origin at the center of fingertip, and some may defines it at the install link of gripper. The orientations may also be different. (c)(d)(e) Example coordinate definitions visualized from different datasets with different embodiments.

## III. Embodiment Equivariance

### A. Problem Formulation

**Definitions.** We define embodiment configuration as the combinition of its end-effector pose under base and camera pose under base: $\mathbf{m} \in \mathcal{M} = \{[{}^{b}_{e}\mathbf{T}, {}^{b}_{c}\mathbf{T}]\}$, and define action as the desired end-effector pose under base: $\mathbf{a} \in \mathcal{A} = \{{}^{b}_{e_*}\mathbf{T}\}$. Given image observation $\mathcal{I}$ and language instruction $\mathcal{L}$, a VLA policy $\pi_\theta$ maps:

$$\mathbf{a} = \pi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \tag{1}$$

Let $\mathcal{G} = \{[{}^{b'}_{b}\mathbf{T}, {}^{e}_{e'}\mathbf{T}]\}$ denote the embodiment transformation group that transforms the existing robot base and end-effector definitions to the new ones (Fig. 2). Given $\mathbf{g} \in \mathcal{G}$, we have:

$$\begin{aligned} \mathbf{g} \circ \mathbf{m} &= [{}^{b'}_{b}\mathbf{T}\,{}^{b}_{e}\mathbf{T}\,{}^{e}_{e'}\mathbf{T},\ {}^{b'}_{b}\mathbf{T}\,{}^{b}_{c}\mathbf{T}] = [{}^{b'}_{e'}\mathbf{T},\ {}^{b'}_{c}\mathbf{T}] \\ \mathbf{g} \circ \mathbf{a} &= {}^{b'}_{b}\mathbf{T}\,{}^{b}_{e_*}\mathbf{T}\,{}^{e}_{e'}\mathbf{T} = {}^{b'}_{e'_*}\mathbf{T} \end{aligned} \tag{2}$$

where $\circ$ denotes the group action operator.

**Embodiment Equivariant Policy**. We desire the policy to be inherently equivariant to configuration transformations rather than learning through data:

$$\pi_\theta(\mathbf{g} \circ \mathbf{m}, \mathcal{I}, \mathcal{L}) = \mathbf{g} \circ \pi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \tag{3}$$

With equivariance ensured, the policy can focus on task-level knowledge rather than embodiment-specific details, enabling more effective pre-training and more efficient fine-tuning.

### B. Architecture Design

Equivariance is challenging to achieve with purely learning-based methods. To address this, we decompose the policy into two parts: a learning based policy $\phi_\theta$ that is invariant to configuration transformation and an analytical decoder $\mathcal{D}$ that is equivariant to configuration transformation:

$$\pi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) = \mathcal{D}(\mathbf{m}, \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L})) \tag{4}$$

where the following constraints are desired:

$$\begin{aligned} \phi_\theta(\mathbf{g} \circ \mathbf{m}, \mathcal{I}, \mathcal{L}) &= \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \\ \mathcal{D}(\mathbf{g} \circ \mathbf{m}, \phi_\theta) &= \mathbf{g} \circ \mathcal{D}(\mathbf{m}, \phi_\theta) \end{aligned} \tag{5}$$

**Lemma 1.** *If $\mathcal{D}$ is equivariant to $\mathcal{G}$ and $\phi_\theta$ is invariant to $\mathcal{G}$, then $\pi_\theta$ is equivariant to $\mathcal{G}$.*

**Proof.** This result follows directly from the constraints in Eq. 5. □

A naive design that satisfies these constraints is:

$$\begin{aligned} \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) &= \phi_\theta(\mathcal{I}, \mathcal{L}) \\ \mathcal{D}([{}^b_c\mathbf{T}, {}^b_e\mathbf{T}], \phi_\theta) &= {}^b_c\mathbf{T}\ \phi_\theta\ {}^b_c\mathbf{T}^{-1}\ {}^b_e\mathbf{T} \end{aligned} \tag{6}$$

**Theorem 2.** *If the network and decoder satisfy Eq. 6, the policy is equivariant to configuration transformations.*

**Proof.** In Eq. 6, $\phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) = \phi_\theta(\mathcal{I}, \mathcal{L})$ means the neural network discards all the embodiment information, therefore $\phi_\theta$ is obviously invariant to $\mathcal{G}$. For decoder $\mathcal{D}$, we have:

$$\begin{aligned} \mathcal{D}(\mathbf{g} \circ \mathbf{m}, \phi_\theta) &= {}^{b'}_c\mathbf{T}\ \phi_\theta\ {}^{b'}_{c'}\mathbf{T}^{-1}\ {}^{b'}_e\mathbf{T} \\ &= {}^{b'}_b\mathbf{T}({}^b_c\mathbf{T}\ \phi_\theta\ {}^b_c\mathbf{T}^{-1}\ {}^b_e\mathbf{T})\ {}^e_{e'}\mathbf{T} \\ &= \mathbf{g} \circ \mathcal{D}(\mathbf{m}, \phi_\theta) \end{aligned} \tag{7}$$

Therefore, $\mathcal{D}$ is equivariant to $\mathcal{G}$. Finally, we have:

$$\begin{aligned} \pi_\theta(\mathbf{g} \circ \mathbf{m}, \mathcal{I}, \mathcal{L}) &= \mathcal{D}(\mathbf{g} \circ \mathbf{m}, \phi_\theta(\mathbf{g} \circ \mathbf{m}, \mathcal{I}, \mathcal{L})) \\ &= \mathbf{g} \circ \mathcal{D}(\mathbf{m}, \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L})) \\ &= \mathbf{g} \circ \pi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \end{aligned} \tag{8}$$

Therefore, the policy $\pi_\theta$ is equivariant to $\mathcal{G}$. □

Note discarding all the embodiment configuration in $\phi_\theta$ is a naive design that yields poor performance. We will discuss how to preserve embodiment configurations while maintaining invariance in section IV. Given the decoder design in Eq. 6, we would see the learning objective $\mathbf{y}$ of $\phi_\theta$ is:

$$\phi_\theta \to \mathbf{y} = {}^b_c\mathbf{T}^{-1}\ {}^b_{e_*}\mathbf{T}\ {}^b_e\mathbf{T}^{-1}\ {}^b_c\mathbf{T} = {}^c_{e_*}\mathbf{T}\ {}^e_c\mathbf{T} \tag{9}$$

It is reasonable to expect a configuration invariant policy $\phi_\theta$ to predict an also configuration invariant objective $\mathbf{y}$.

### C. Trade-off for Positioning Precision

In real-world dataset, the end-effector poses are reliable as they are directly read from robot SDKs. However, the cameras are often not well calibrated, introducing errors in camera extrinsic. While the decoder design in Eq. 6 achieves full equivariance, its learning objective $\mathbf{y}$ is sensitive to such calibration errors, especially in the translation components. We could reformulate the translation component of the original learning objective to:

$$\mathbf{y}^t = {}^c_{e_*}\mathbf{R}\ {}^e_c\mathbf{t} + {}^c_{e_*}\mathbf{t} = {}^c_e\mathbf{R}({}^e_{e_*}\mathbf{R} - \mathbf{I})\ {}^e_c\mathbf{t} + {}^c_e\mathbf{R}\ {}^e_{e_*}\mathbf{t} \tag{10}$$

In Eq. 10, we decompose $\mathbf{y}_t$ into relative end-effector pose related components (accurate ${}^e_{e_*}\mathbf{R}$ and ${}^e_{e_*}\mathbf{t}$) and camera extrinsic related components (inaccurate ${}^c_e\mathbf{R}$ and ${}^e_c\mathbf{t}$). We notice that in most cases we have $\|{}^e_c\mathbf{t}\| > \|{}^e_{e_*}\mathbf{t}\|$, and the extrinsic calibration error will be amplified by $\|{}^e_c\mathbf{t}\|$, yielding large translation error which may overwhelm the norm of ground truth learning objective $\mathbf{y}^t$ itself and make the pre-training less effective. Therefore, we remove the ${}^e_c\mathbf{t}$ related item, and refine the decoder and the corresponding learning objective to:

$$\begin{aligned} \mathcal{D}_r([{}^b_c\mathbf{T}, {}^b_e\mathbf{T}], \phi_\theta) &= \begin{bmatrix} {}^b_c\mathbf{R}\phi^R_\theta\ {}^b_c\mathbf{R}^{-1}\ {}^b_e\mathbf{R} & {}^b_c\mathbf{R}\phi^t_\theta + {}^b_e\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \\ \phi_\theta \to \mathbf{y}_r &= \begin{bmatrix} {}^c_{e_*}\mathbf{R}\ {}^e_c\mathbf{R} & {}^c_e\mathbf{R}\ {}^e_{e_*}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \end{aligned} \tag{11}$$

The revised decoder in Eq. 11 remains equivariant to $\mathcal{G}$ except for pure end-effector translations transformation group $\mathcal{T} = \{{}^e_{e'}\mathbf{t}\}$, but achieves much greater robustness in practice. All subsequent experiments are based on this revised design.

### D. Discussion of Existing Policy Learning

Beyond our proposed formulation, several alternative action spaces are commonly used in VLA policies:

$$\begin{aligned} &\text{BE:} && \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \to {}^b_{e_*}\mathbf{T}, && \mathcal{D}(\mathbf{m}, \phi_\theta) = \phi_\theta \\ &\text{EE:} && \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \to {}^e_{e_*}\mathbf{T}, && \mathcal{D}(\mathbf{m}, \phi_\theta) = {}^b_e\mathbf{T}\ \phi_\theta \\ &\text{CE:} && \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \to {}^c_{e_*}\mathbf{T}, && \mathcal{D}(\mathbf{m}, \phi_\theta) = {}^b_c\mathbf{T}\ \phi_\theta \\ &\text{IM:} && \phi_\theta(\mathbf{m}, \mathcal{I}, \mathcal{L}) \to \mathcal{I}_*, && \mathcal{D}(\mathbf{m}, \phi_\theta) = \psi_\theta(\mathcal{I}, \phi_\theta) \end{aligned} \tag{12}$$

Denoting $\mathcal{G}_b = \{{}^{b'}_b\mathbf{T}\}$ and $\mathcal{G}_e = \{{}^e_{e'}\mathbf{T}\}$ separately as base transformation group and end-effector transformation group, we could draw the following conclusions:

**1)** BE predicts the absolute end-effector pose in the base frame. The decoder is invariant to $\mathcal{G}$, forcing the policy to learn equivariance through data.

**2)** EE predicts the relative end-effector pose, which is adopted by OpenVLA [5] and $\pi_0$ [4] when pre-training on a subset of OXE dataset [23]. The decoder is equivariant to $\mathcal{G}_b$, however, is neither equivariant nor invariant to $\mathcal{G}_e$. Therefore, the policy is not equivariant to $\mathcal{G}$.

**3)** CE predicts the end-effector pose in camera frame, which is adopted by RISE [25]. The decoder is equivariant to $\mathcal{G}_b$ and invariant to $\mathcal{G}_e$, which requires the $\phi_\theta$ to be invariant to $\mathcal{G}_b$ and equivariant to $\mathcal{G}_e$. While the former can be easily achieved by reformulating $\mathbf{m}$ to ${}^c_e\mathbf{T}$, the latter is not guaranteed by network structure but learning through data.

**4)** IM predicts the dense/sparse future image space features (e.g., keypoints in ATM [21] or latent embeddings in Uni-VLA [22]). The encoder is invariant to $\mathcal{G}$, however, the decoder is a neural network without guaranteed equivariance.

These comparisons highlight the limitations of existing formulations and motivate our design of a theoretically grounded, embodiment transformation equivariant policy.

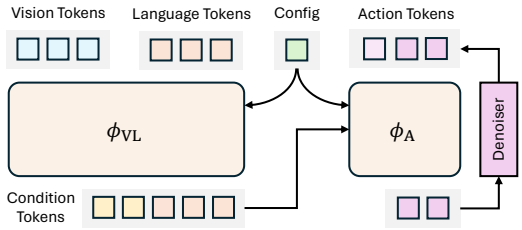


Fig. 3: Schematics of model $\phi$. Model $\phi$ is composed of $\phi_{\text{VL}}$ and $\phi_{\text{A}}$. $\phi_{\text{VL}}$ generate geometric-aware and task-related vision-language condition tokens. $\phi_{\text{A}}$ generate actions with conditions from $\phi_{\text{VL}}$.

## IV. Geometry Utilization

In the previous design (Eq. 6), we discard all the embodiment configurations information in neural network modeling. However, learning 3D geometry solely from multi-view images is difficult. Embodiment configurations provide valuable cues for 3D understanding, which can improve positioning precision and robustness to viewpoint variations. Directly injecting camera extrinsics or embodiment poses as tokens, however, breaks the required invariance. We therefore seek strategies to incorporate geometric information while preserving invariance to configuration transformations.

### A. Relative Camera Pose Embedding

Following GTA [26] and PRoPE [27], we embed camera intrinsic and relative camera pose into the attention mechanism of visual tokens (feature dimension $= d$):

$$\begin{aligned}
&[\mathbf{q}_i;\mathbf{k}_i;\mathbf{v}_i] = [\mathbf{W}_q;\mathbf{W}_k;\mathbf{W}_v](\mathbf{c}_i + \mathbf{W}_p\mathbf{p}_i) \\
&\mathbf{q}'_i = \sigma({}^c_b\mathbf{T}_i^\top)\mathbf{q}_i,\ \ \mathbf{k}'_i = \sigma({}^b_c\mathbf{T}_i)\mathbf{k}_i,\ \ \mathbf{v}'_i = \sigma({}^b_c\mathbf{T}_i)\mathbf{v}_i \\
&\mathbf{o}_i = \sigma({}^c_b\mathbf{T}_i)\sum_j \frac{\exp(\mathbf{q}_i'^\top\mathbf{k}'_j/\sqrt{d})}{\sum_k \exp(\mathbf{q}_i'^\top\mathbf{k}'_k/\sqrt{d})}\mathbf{v}'_j
\end{aligned} \tag{13}$$

where $\sigma(\mathbf{T})$ compose the pose matrix $\mathbf{T}$ for $d/4$ times to build a large block-diagonal matrix:

$$\sigma(\mathbf{T}) = \begin{bmatrix} \mathbf{T} & & \\ & \ddots & \\ & & \mathbf{T} \end{bmatrix}_{d\times d} \tag{14}$$

$\mathbf{p}_i$ is the normalized camera place coordinates of $i$-th image token $\mathbf{c}_i$, and ${}^b_c\mathbf{T}_i$ is its associated camera pose. We adopt the additive positional embedding for intrinsic and rotary positional embedding for extrinsic. This design allows geometry-aware self-attention without violating invariance constraints.

### B. Actions as Positional Embedding

We generate actions with diffusion. At each diffusion timestep $k$, we add positional embedding to the noisy action tokens when cross attending to condition tokens:

$$\begin{aligned}
\mathbf{q}^a &= \mathbf{W}_q(\mathbf{W}_y\hat{\mathbf{y}}_k + \mathbf{W}_p\mathbf{p}^a_k) \\
\mathbf{p}^a_k &= \mathcal{P}({}^b_c\mathbf{T}^{-1}\ \mathcal{D}(\mathbf{m}, \hat{\mathbf{y}}_k)) \\
\mathbf{o}^a &= \text{scale_dot_product}(\mathbf{q}^a, \mathbf{k}, \mathbf{v})
\end{aligned} \tag{15}$$

where $\mathcal{P}$ projects the 3D position of future noisy action to the image space. By converting embodiment configurations into positional encodings in image space, projected future actions provide feedback during recurrent denoising. This explicitly links actions to image patch coordinates, thereby improving positioning precision.

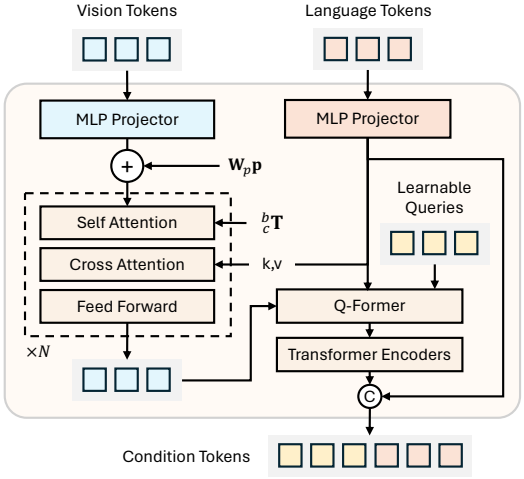


Fig. 4: Schematics of model $\phi_{\text{VL}}$. The initial vision tokens and language tokens are extracted from fronzen DINOv2 and SigLIP models. Image self attention with relative camera pose embedding is adopted to enhance geometry understanding. Image-language cross attention is adopted to enhance instruction following. Q-Former is adopted to compress task-related vision features for efficient action generation.

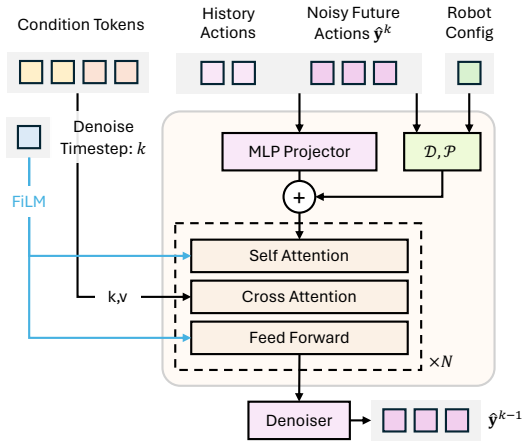


Fig. 5: Schematics of model $\phi_{\text{A}}$. We use cross attention to inject vision-language conditions and FiLM to inject denoise timestep.

### C. Network Structure

Following standard VLA policy design, the policy network is decomposed into a vision-language encoder $\phi_{\text{VL}}$ and an action generator $\phi_{\text{A}}$ conditioned on $\phi_{\text{VL}}$.

In $\phi_{\text{VL}}$, we fuse the DINOv2 [28] features and SigLIP [29] image features with a MLP, followed by repeated image self-attention and image-text cross-attention. Relative camera pose embedding is injected into all the self-attention layers of vision features. A one-layer Q-Former [30] with 64 learnable queries is used to compress task-relevant features for action generation.

In $\phi_{\text{A}}$, action tokens are processed by self-attention and cross-attention with $\phi_{\text{VL}}$ outputs. Action positional embed-

TABLE I: Datasets used in pre-training and fine-tuning. Datasets with notation * are self-collected. The LIBERO dataset are regenerated with original action replayed and failed demonstrations removed.

| Stage | Dataset | Episodes | Cameras | Robot |
|---|---|---|---|---|
| Pre-training | Droid | 78,544 | 3 | Franka |
| | Maniskill | 30,123 | 2 | Franka |
| | Meta-world | 2,500 | 3 of 7 | Sawyer |
| | Pick Place Cans* | 2,000 | 2 | UR5 |
| | Open Drawer* | 2,000 | 1 | UR5 |
| | Open Microwave* | 2,000 | 1 | UR5 |
| Fine-tuning, Simulation | LIBERO-Spatial | 438 | 2 | Franka |
| | LIBERO-Object | 452 | | |
| | LIBERO-Goal | 421 | | |
| | LIBERO-Long | 376 | | |
| Fine-tuning, Real-world | Pick Place Pepper* | 10 | 2 | Piper |
| | Pick Place Carrot* | 10 | | |
| | Pick Place Eggplant* | 10 | | |
| | Table Storage* | 30 | | |

TABLE II: Success rate (%) and total parameters (B) of fine-tuned policies on LIBERO benchmark.

| Method | Spatial | Goal | Object | Long | Avg | Param |
|---|---|---|---|---|---|---|
| OpenVLA | 84.7 | 88.4 | 79.2 | 53.7 | 76.5 | 7 |
| $\pi_0$ | 96.8 | **98.8** | 95.8 | 85.2 | 94.15 | 2.3 |
| UniVLA | 96.5 | 96.8 | 95.6 | 92.0 | 95.2 | 8 |
| Ours | **97.6** | 93.7 | **99.8** | **94.4** | **96.38** | **0.4** |

TABLE III: Zero-shot and few-shot results on pick place task with a new embodiment. Our method achieves consistent success rate when transferring to the new embodiment.

| Configuration | $\pi_0$ | BE | CE | EE | Ours |
|---|---|---|---|---|---|
| UR5 (original) | 82 | 80 | 72 | **98** | **98** |
| Fanuc (zero-shot) | 0 | 0 | 0 | 0 | **94** |
| Fanuc (fine-tune) | 22 | 50 | 52 | 92 | **98** |

dings are injected once before the first self-attention layer. We use FiLM [31] to inject denoise time step $k$ into self-attention layers and feed forward layers. The final denoised learning objective $\hat{\mathbf{y}}_0$ is decoded to ${}^{b}_{e_*}\hat{\mathbf{T}}$ with $\mathcal{D}$ for execution.

Excluding the frozen vision-language backbones, our action expert consists total 122M trainable parameters, which is approximately 1/3 of the action expert size in $\pi_0$ [4].

### D. Implementation Details

**Dataset.** We pre-train our policy on a real-world dataset (Droid [32]) and several simulation datasets (ManiSkill [33], Meta-world [34] and self collected simulation data using IsaacSim). The policy is fine-tuned on LIBERO [35] for simulated benchmark, and self collected datasets for real-world evaluation. Details of datasets are listed in Table. I.

**Training.** We use AdamW as the optimizer with learning rate 1e-4 and weight decay 1e-2. We use linear learning rate warmup for 20k iterations in pre-training and 2k iterations in fine-tuning. We train all the models with batch size 32 in mixed precision (float32 and bfloat16). Pre-training runs for 600k iterations, which costs around 55 hours on single RTX 4090. We use 100-step DDPM in training and 20-step DDIM in inference.

**Learning Objective.** As shown in Eq. 11, the learning objective $\hat{\mathbf{y}}_r$ of policy $\phi_\theta$ lies on the $\mathrm{SE}(3)$ manifold. We use 6D representation [36] of rotation component. We additionally regress the normalized gripper width ($-1$ = fully closed and $1$ = fully open). The total action dimension is 10 (3 for translation, 6 for 6D rotation and 1 for gripper).

## V. Simulation Experiments

### A. Performance on LIBERO Benchmark

After pre-training, we fine-tune our model on LIBERO [35] simulation benchmarks, including LIBERO-(Spatial, Goal, Object and Long). Since the original datasets lack camera extrinsics and intrinsics, we follow OpenVLA [5] to replay the ground truth actions and record this information. Failed demonstrations are removed before fine-tuning. We compare our fine-tuned policy with recent VLA baselines. Results in Table. II show our policy achieves competitive or superior performance while requiring substantially lower fine-tuning cost due to more effective pre-training. Specifically, our policy reaches its reported results with only 40k iterations (5 GPU hours) fine-tuning, whereas $\pi_0$ [4] requires $\sim$34 GPU hours, and OpenVLA [5] / UniVLA [22] require over 300 GPU hours.

### B. Ablation Study of Different Action Space Design

We also pre-train three policies with the same neural network structure as ours but with different decoder and learning objectives. Success rates versus fine-tuning iterations are shown in Fig. 6.

These results highlight that our action space yields the most efficient fine-tuning, since it preserves consistent action representation across tasks regardless of embodiment configurations. We notice EE action space and ours are both in the relative action space, achieving higher success rate than the absolute action space (BE and CE). This is because relative action spaces produce approximately zero-centered action distributions, whereas absolute spaces suffer from mean-shift when transferring across datasets, making fine-tuning harder and reducing positioning precision. We also observe that the naive variant of our method (discarding all embodiment configuration information) performs worst, indicating that embodiment information is essential for trajectory positioning precision.

### C. Zero/Few-shot to New Embodiment Configuration

We further evaluate cross-embodiment generalization in a simulated pick-place task. We first fine-tune all the policies to reach their best performance with sufficient demonstrations collected by UR5. We then replace the embodiment from UR5 to Fanuc (Fig. 7), which is never seen in the pre-training stage of our policy and $\pi_0$. We also shift the base pose and rotate the definition of end-effector coordinates. We measure zero-shot and few-shot performance on this new embodiment.

As shown in Table III, our method maintains a high success rate even under zero-shot transfer to the Fanuc arm. Although appearance differences between embodiments

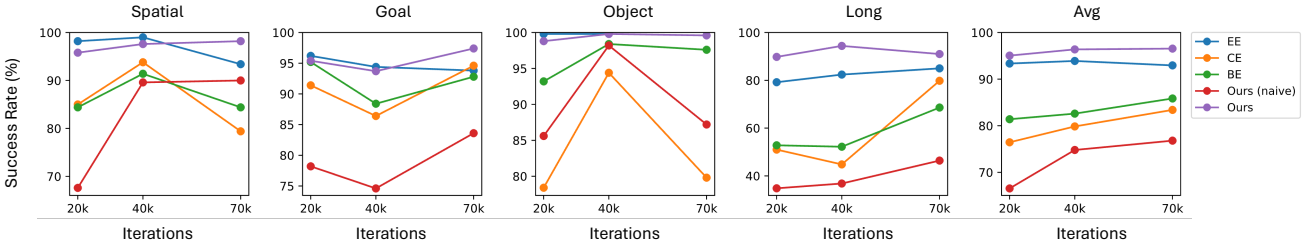


Fig. 6: Performance of policies with different action space on LIBERO benchmark. We evaluate policies fine-tuned for 20k, 40k and 70k iterations. Our method achieves the highest success rate on the average. All the methods except for the naive version of our implementation utilize the geometric information metioned in section IV, which shows the embodiment configuration information is important to the performance.

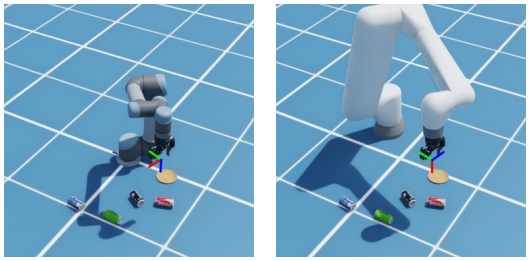

Fig. 7: Pick place task in simulation. The policies are first fine-tuned with sufficient data (1000 demonstrations) generated using UR5 (left), and zero-shot evaluated on the Fanuc (right), and finally fine-tuned with little data (50 demonstrations) generated using Fanuc (right). We shift both the robot base and end-effector definitions. The pose of third-person camera is randomized in demonstration and evaluation.

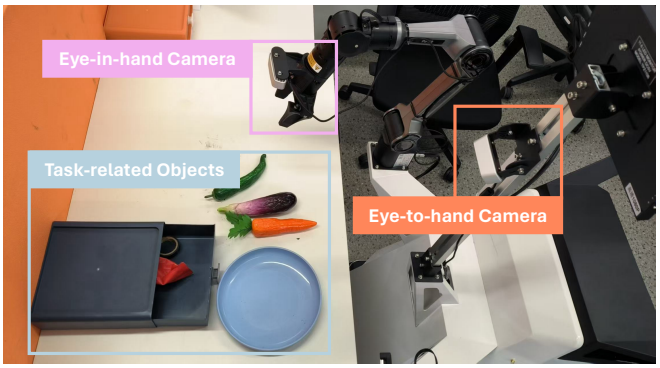


Fig. 8: Data collection and policy evaluation platform in real-world. We use images from two calibrated cameras as observations.

cause slight degradation, our equivariant design still generalizes effectively. Other action spaces completely fail in zero-shot settings. With few-shot fine-tuning, our method and EE action space recover to the comparable performance as on UR5, while others struggle to close the gap. We also found the joint space action performs the worst when adapting to new embodiment with few demonstrations.

## VI. Real-world Experiments

We conduct real-world evaluations on two tasks, pick-place and table storage, using the Agilex Cobot platform equipped with Piper arms (Fig. 8). For each task, we collect 30 teleoperated demonstrations. Both our policy and $\pi_0$ are fine-tuned on this data for 40k iterations with a batch size of 32. Notably, the Agilex Cobot configuration is unseen during our pre-training, while $\pi_0$ was pre-trained partially on mobile Trossen and mobile ARX, which share similar embodiment configuration with Agilex Cobot. In all experiments, $\pi_0$ is fine-tuned with joint-space actions, consistent with its pre-training setup for the mobile Trossen/ARX. Sampled successful roll-outs are visualized in Fig. 10.

### A. Pick-Place Task

**Task Description.** The prompt is: *Pick up the* $\{object\}$ *and place it on the plate*, where the object is selected from pepper, carrot and eggplant. The example initial state and the desired final state is visualized in Fig. 9a. We evaluate each policy for 60 trials. For the first 30 trials, there exists only the plate and the desired object. For the rest 30 trials, we place all the objects on the table acting as distractors.

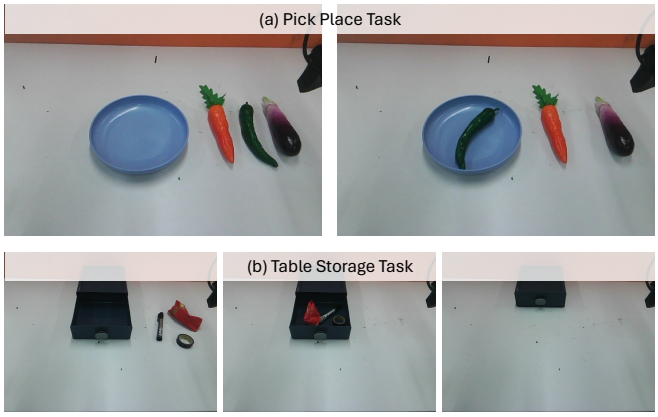


Fig. 9: Tasks in real-world experiments. (a) The initial state (left) and the desired final state (right) of pick place task. The example prompt is: *pick up the pepper and place it on the plate*. (b) Three stages of completing the table storage task. Left to middle: pick all the items on the table and place them in the box. Middle to right: push to close the box.

**Result Analysis.** As shown in Table. IV, both $\pi_0$ and our policy achieve high success rate without distractors. However, $\pi_0$ degrades substantially when distractors are introduced, reflecting its weaker grounding between language and action given limited data for fine-tuning. In contrast, our policy maintains nearly perfect performance. Fig. 11 visualizes how distractors influence the trajectory planned by $\pi_0$, while ours remains almost unaffected.

### B. Table Storage Task

**Task Description.** The prompt is: *Put the items on the table into the storage box and close the box*. The robot needs

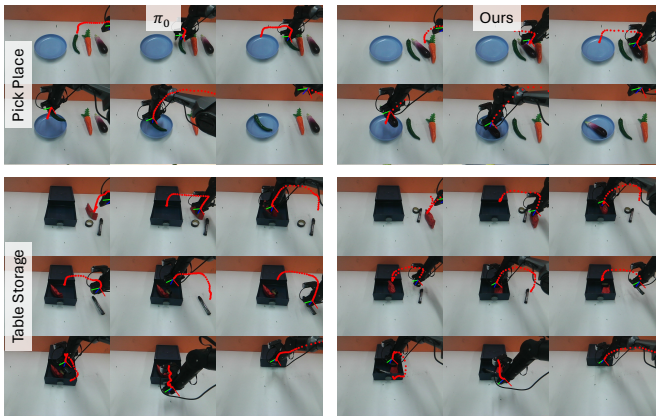


Fig. 10: Example success roll-outs of $\pi_0$ and our policy on pick place task and stable storage task.

TABLE IV: Success rate of real-world pick place task.

| Distractor | Method | Pepper | Carrot | Eggplant | Avg |
|---|---|---|---|---|---|
| ✗ | $\pi_0$ | 7/10 | 9/10 | 10/10 | 26/30 |
| | Ours | 10/10 | 10/10 | 10/10 | 30/30 |
| ✓ | $\pi_0$ | 4/10 | 4/10 | 3/10 | 11/30 |
| | Ours | 9/10 | 10/10 | 10/10 | 29/30 |

to pick up all the items (including a marker pen, tape and snack) on the table, place them in the box, and finally push the box closed. The sample initial state, intermediate state and desired final state is visualized in Fig. 9b. We evaluate the policies for 40 trials. For the first 30 trials, we only perturb the position of the box and the items. For the rest 10 trials, we perturb the pose of robot base, resulting in different camera viewpoints (Fig. 12).

**Result Analysis.** As completing the table storage requiring multiple pick-place and push attempts, we report the success rate of each step. Results (Table. V) show that our policy consistently outperforms $\pi_0$, particularly under base pose disturbance. The robustness comes from representing relative end-effector motions in the camera frame, whereas $\pi_0$ overfits to embodiment-specific dynamics and fails to close the box.

**Failure Case Analysis.** As shown in Fig. 13, we found the most common issue arises from inaccurate Z-axis localization using the eye-in-hand camera, especially when grasping slender objects like the pen. The inaccurate XY-axis localization may cause collision between the gripper and the tape. $\pi_0$ frequently fails at the box-closing step due to its reliance on joint-space actions, which cannot generalize well with pose disturbance. Our policy occasionally fails under unreachable poses or kinematic singularities, though it is generally more robust overall.

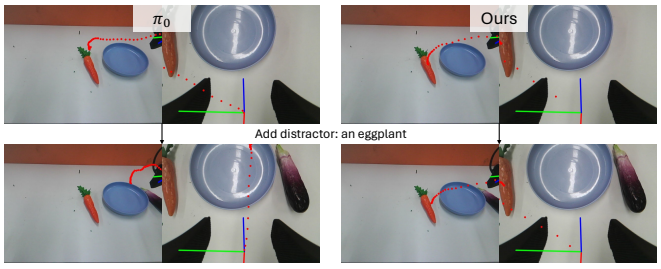


Fig. 11: Effect of distractors on planned trajectories (shown as red dots). The prompt is: *pick up the carrot and place it on the plate*. Both $\pi_0$ and our policy plan correct trajectories without distractors. However, When an eggplant is placed beside the plate, $\pi_0$ is easily distracted, while ours remains robust.

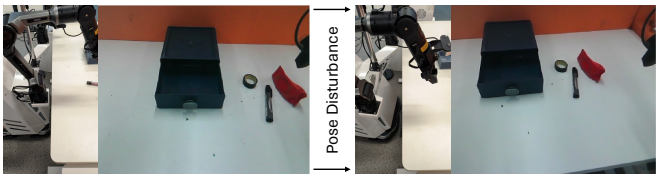


Fig. 12: Robot base pose disturbance is applied in the last 10 trials evaluation of table storage task.

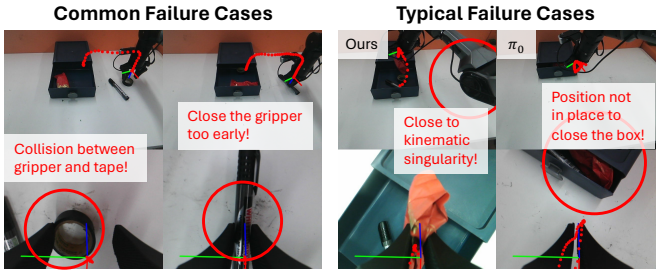


Fig. 13: Common and typical failure cases of our policy and $\pi_0$.

## VII. Limitations

Although our action space and network architecture design fully leverage the relative pose representation in camera space for efficient policy fine-tuning, it requires camera extrinsic and intrinsic recorded. In practice, many large-scale real-world robotic datasets provide high-quality images, detailed language annotations, and diverse environments, but often lack reliable camera calibration, or even doesn't record camera parameters. This limitation currently prevents internet large-scale pre-training of our policy. However, we have seen more researchers realizing the importance of camera parameters in data collection and policy learning. In addition, emerging approaches such as VGGT [37] demonstrate the potential of recovering pseudo camera poses from images, which may help alleviate this constraint in future work.

## VIII. Conclusion

In this paper, we propose an action space invariant to embodiment configuration transformation and the corresponding action decoder equivariant to the transformation. Extensive

TABLE V: Success rate of real-world table storage task.

| Disturbance | Method | Pen | Snack | Tape | Box | Overall |
|---|---|---|---|---|---|---|
| ✗ | $\pi_0$ | 27/30 | 28/30 | 24/30 | 22/30 | 20/30 |
| | Ours | 29/30 | 28/30 | 26/30 | 27/30 | 24/30 |
| ✓ | $\pi_0$ | 6/10 | 8/10 | 5/10 | 0/10 | 0/10 |
| | Ours | 9/10 | 10/10 | 10/10 | 9/10 | 9/10 |

simulation experiments on the LIBERO benchmark demonstrated that our policy achieves state-of-the-art performance with significantly reduced fine-tuning cost. Moreover, our approach enables efficient transfer to unseen embodiments, achieving high success rates even under zero- and few-shot settings. Real-world experiments further validated the effectiveness and robustness of our method in manipulation tasks with distractors and embodiment perturbations. Our proposed action space and architecture design is a feasible choice for effective cross-embodiment pre-training and efficient fine-tuning.

## REFERENCES


[1] 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," in *RSS*, 2023. [Online]. Available: https://doi.org/10.15607/RSS.2023.XIX.025

[2] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid *et al.*, "RT-2: vision-language-action models transfer web knowledge to robotic control," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 229. PMLR, 2023, pp. 2165–2183.

[3] 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," in *ICLR*. OpenReview.net, 2025.

[4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "$\pi_0$: A vision-language-action flow model for general robot control," 2024. [Online]. Available: https://arxiv.org/abs/2410.24164

[5] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong *et al.*, "Openvla: An open-source vision-language-action model," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 270. PMLR, 2024, pp. 2679–2713.

[6] M. Weiler and G. Cesa, "General e(2)-equivariant steerable cnns," in *NIPS*, 2019, pp. 14 334–14 345.

[7] F. Fuchs, D. E. Worrall, V. Fischer, and M. Welling, "Se(3)-transformers: 3d roto-translation equivariant attention networks," in *NIPS*, 2020.

[8] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) equivariant graph neural networks," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 9323–9332.

[9] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *RSS*, 2023. [Online]. Available: https://doi.org/10.15607/RSS.2023.XIX.016

[10] D. Ghosh, H. R. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo *et al.*, "Octo: An open-source generalist robot policy," in *RSS*, 2024. [Online]. Available: https://doi.org/10.15607/RSS.2024.XX.090

[11] Q. Bu, H. Li, L. Chen, J. Cai, J. Zeng, H. Cui, M. Yao, and Y. Qiao, "Towards synergistic, generalized, and efficient dual-system for robotic manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2410.08001

[12] 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," 2025. [Online]. Available: https://arxiv.org/abs/2506.01953

[13] C. Cui, P. Ding, W. Song, S. Bai, X. Tong, Z. Ge, R. Suo, W. Zhou, Y. Liu, B. Jia *et al.*, "Openhelix: A short survey, empirical analysis, and open-source dual-system vla model for robotic manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2505.03912

[14] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *RSS*, 2023. [Online]. Available: https://doi.org/10.15607/RSS.2023.XIX.026

[15] J. Wen, Y. Zhu, J. Li, Z. Tang, C. Shen, and F. Feng, "Dexvla: Vision-language model with plug-in diffusion expert for general robot control," 2025. [Online]. Available: https://arxiv.org/abs/2502.05855

[16] J. Wen, M. Zhu, Y. Zhu, Z. Tang, J. Li, Z. Zhou, C. Li, X. Liu, Y. Peng, C. Shen *et al.*, "Diffusion-vla: Generalizable and interpretable robot foundation model via self-generated reasoning," 2025. [Online]. Available: https://arxiv.org/abs/2412.03293

[17] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine, "Zero-shot robotic manipulation with pretrained image-editing diffusion models," 2023. [Online]. Available: https://arxiv.org/abs/2310.10639

[18] H. Wu, Y. Jing, C. Cheang, G. Chen, J. Xu, X. Li, M. Liu, H. Li, and T. Kong, "Unleashing large-scale video generative pre-training for visual robot manipulation," in *ICLR*. OpenReview.net, 2024.

[19] C.-L. Cheang, G. Chen, Y. Jing, T. Kong, H. Li, Y. Li, Y. Liu, H. Wu, J. Xu, Y. Yang *et al.*, "Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2410.06158

[20] C. Cheang, S. Chen, Z. Cui, Y. Hu, L. Huang, T. Kong, H. Li, Y. Li, Y. Liu, X. Ma *et al.*, "Gr-3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2507.15493

[21] C. Wen, X. Lin, J. I. R. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel, "Any-point trajectory modeling for policy learning," in *RSS*, 2024. [Online]. Available: https://doi.org/10.15607/RSS.2024.XX.092

[22] Q. Bu, Y. Yang, J. Cai, S. Gao, G. Ren, M. Yao, P. Luo, and H. Li, "Univla: Learning to act anywhere with task-centric latent actions," 2025. [Online]. Available: https://arxiv.org/abs/2505.06111

[23] 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," in *ICRA*. IEEE, 2024, pp. 6892–6903. [Online]. Available: https://doi.org/10.1109/ICRA57147.2024.10611477

[24] J. H. Yang, C. Glossop, A. Bhorkar, D. Shah, Q. Vuong, C. Finn, D. Sadigh, and S. Levine, "Pushing the limits of cross-embodiment learning for manipulation and navigation," in *RSS*, 2024. [Online]. Available: https://doi.org/10.15607/RSS.2024.XX.093

[25] C. Wang, H. Fang, H.-S. Fang, and C. Lu, "Rise: 3d perception makes real-world robot imitation simple and effective," in *IROS*, 2024, pp. 2870–2877.

[26] T. Miyato, B. Jaeger, M. Welling, and A. Geiger, "GTA: A geometry-aware attention mechanism for multi-view transformers," in *ICLR, May 7-11, 2024*. OpenReview.net, 2024.

[27] R. Li, B. Yi, J. Liu, H. Gao, Y. Ma, and A. Kanazawa, "Cameras as relative positional encoding," 2025. [Online]. Available: https://arxiv.org/abs/2507.10496

[28] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *Trans. Mach. Learn. Res.*, vol. 2024, 2024.

[29] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *ICCV*. IEEE, 2023, pp. 11 941–11 952. [Online]. Available: https://doi.org/10.1109/ICCV51070.2023.01100

[30] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. C. H. Hoi, "Instructblip: Towards general-purpose vision-language models with instruction tuning," in *NeurIPS*, 2023.

[31] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*. AAAI Press, 2018, pp. 3942–3951. [Online]. Available: https://doi.org/10.1609/aaai.v32i1.11671

[32] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis *et al.*, "DROID: A large-scale in-the-wild robot manipulation dataset," in *RSS*, 2024. [Online]. Available: https://doi.org/10.15607/RSS.2024.XX.120

[33] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations," 2021. [Online]. Available: https://arxiv.org/abs/2107.14483

[34] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 100. PMLR, 2019, pp. 1094–1100.

[35] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "LIBERO: benchmarking knowledge transfer for lifelong robot learning," in *NeurIPS*, 2023.

[36] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *CVPR*, 2019, pp. 5738–5746.

[37] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, "Vggt: Visual geometry grounded transformer," in *CVPR*, 2025.