

SAVLA: Symmetry-Aware Vision-Language-Action Models for Robotic Manipulation

Abstract—Vision-language-action (VLA) models have become the dominant paradigm for language-conditioned robot manipulation. However, although images and language instructions inherently encode geometric information, VLAs acquire their spatial competence purely from demonstrations. As a result, they are reliable only within the range of scene poses that the demonstrations cover. We propose SAVLA, an end-to-end symmetry-aware VLA model for robust and data-efficient policy learning. Our approach keeps the pretrained vision-language backbone entirely frozen while combining it with an equivariant flow-matching action head and a learned canonicalizer. The head decomposes its state, action, and conditioning inputs into invariant and equivariant channels, and preserves this typing throughout all of its layers. The canonicalizer transforms oblique-view images into a canonical frame and rotates the geometric conditions consistently. We evaluate our model on LIBERO. Compared with the GR00T N1.5 baseline, SAVLA improves the success rate averaged over all four LIBERO suites by 5.1 points and increases the mean success rate under rotation on LIBERO-Goal from 41.5% to 90.4%.

I. INTRODUCTION

Vision-language-action (VLA) models, which couple a large pretrained vision-language model with a learned action head, have quickly become the dominant recipe for language-conditioned robot manipulation. VLAs provide a unified framework for mapping visual observations and natural-language instructions to robot actions [1]–[4]. However, standard VLAs encode no explicit architectural prior for rotational symmetry in visuomotor control [5]–[7], and consequently remain sensitive to controlled changes in object layout, camera viewpoint, and robot initial state [8]. Recent work has enhanced VLA spatial reasoning by injecting egocentric 3D position encodings and adaptive spatial action grids [9], but the resulting policy is not explicitly constrained to be rotational equivariant. Other works build geometric symmetry directly into generative robot policies using point-cloud observations [10], [11], but their reliance on 3D point clouds limits their direct integration with VLAs built primarily on pretrained RGB encoders.

In this work, we propose SAVLA, a symmetry-aware VLA that combines a frozen, non-equivariant vision-language backbone with an equivariant action head. SAVLA treats the output features from the frozen backbone as invariant and combines them with equivariant geometric conditions in an equivariant flow-matching action head. It further canonicalizes oblique-view RGB images while applying the corresponding rotation to the geometric conditions.

Our main contributions are as follows:

- 1) **An equivariant action head.** We design a flow-matching action head that is equivariant.

- 2) **A typed interface.** We introduce an interface that generates the invariant and equivariant conditions for the action head, allowing it to be driven by a backbone that is not equivariant.
- 3) **An end-to-end symmetry-aware VLA.** Together with a canonicalization head that maps the oblique view into a canonical frame, these components form an end-to-end symmetry-aware VLA.

We evaluate SAVLA on LIBERO. Compared with the GR00T N1.5 baseline, SAVLA achieves a 5.1-point improvement in average success rate averaged across four LIBERO suites and a substantial gain under rotation, with the mean success rate on LIBERO-Goal increasing from 41.5% to 90.4%. Code is available at <https://anonymous.4open.science/r/SAVLA-48D0/>.

II. RELATED WORK

A. Vision-Language-Action Models

Early VLAs, including RT-1, RT-2, and OpenVLA, generate discretized action tokens autoregressively [1], [2], [12]. More recent systems adopt continuous action modules: OpenVLA-OFT introduces parallel action decoding and action chunking [5]. Octo provides a closely related frozen-trunk, replaceable-head factorization [13]. π_0 , SmolVLA, and GR00T employ diffusion or flow-matching action experts [3], [4], [6], [7]. Specifically, GR00T N1 [4] formalizes the split as a dual-system architecture: a vision-language model provides slow semantic reasoning, while a diffusion-transformer action head denoises action chunks at control rate, conditioned on the backbone’s tokens. GR00T N1.5 [7] upgrades that backbone to a SigLIP2 vision encoder paired with a Qwen3-1.7B language model, using the hidden features from the 12th layer. SAVLA is built on GR00T N1.5. EquiVLA [14] combines token-level frame averaging with a steerable flow-matching action head. In contrast, SAVLA canonicalizes oblique-view images and conditions its $SO(3)$ -equivariant action head on approximately invariant VLM features and equivariant geometric reference points.

B. Diffusion and Flow-Matching Policies

Behavior cloning with a unimodal regression loss cannot represent the multimodality of demonstration data. Diffusion Policy [15] addresses this by generating action chunks with a denoising diffusion model [16], and becomes the standard action head for imitation learning. Flow matching [17] offers a closely related but simpler formulation, learning a velocity field that transports noise to data. With the optimal-transport

path, the resulting trajectories are close to straight, allowing accurate sampling in few integration steps.

A parallel line of work equips such generative policies with symmetry. Equivariant Diffusion Policy [18] constrains the denoising network of a diffusion policy to be $\text{SO}(2)$ -equivariant, reporting substantial gains in the low-data regime. EquiBot [10] extends the construction to $\text{SIM}(3)$ equivariance on point-cloud observations, covering scale in addition to rotation and translation. Several works target the full $\text{SE}(3)$ group. Diffusion-EDFs [19] formulates pick-and-place as bi-equivariant denoising generative modeling on $\text{SE}(3)$. RiEMann [20] learns $\text{SE}(3)$ -equivariant manipulation directly from unsegmented point clouds at near real-time rates. ET-SEED [11] lifts the equivariance from single poses to whole trajectories.

C. Canonicalization

An alternative to building equivariance into the architecture is to normalize the input instead. Learned canonicalization [21] trains a small network to predict a canonical pose. The input is mapped into that frame before the backbone, and the output is mapped back afterwards. Probabilistic symmetrization [22] generalizes the idea to sampled group elements. In robot manipulation, Canonical Policy [23] learns a canonical 3D representation for an $\text{SE}(3)$ -equivariant visuomotor policy.

The effectiveness of equivariance depends on the accuracy of the canonicalizer: errors in the estimated pose propagate directly to the prediction. Architectural equivariance offers the opposite trade-off: it guarantees equivariance by construction, but requires equivariant design throughout the network, which is impractical for a multi-billion-parameter pretrained backbone. SAVLA deliberately combines the two approaches at different stages of the pipeline. Canonicalization normalizes the input image for the frozen backbone, whose features are subsequently treated as invariant, while the estimated scene-rotation is also used to rotate the geometric conditions of the equivariant action head (Section IV-B).

III. PRELIMINARIES

Problem setup: We consider language-conditioned manipulation. At each control step t , the policy receives an observation $o_t = (I_t^{3p}, I_t^w, s_t, \ell)$, where I_t^{3p} is the third-person image, I_t^w is the wrist image, s_t is the proprioceptive state of the arm, and ℓ is the language instruction. The policy then generates a chunk of future actions $(a_t, \dots, a_{t+T_p-1})$, where T_p is the horizon.

Equivariance and typed features: A group G acts on a vector space through a representation ρ , which assigns to each $R \in G$ an invertible linear map $\rho(R)$. A map f is equivariant if transforming its input by a group element produces the correspondingly transformed output: $f(\rho_{\text{in}}(R)x) = \rho_{\text{out}}(R)f(x)$; invariance is the special case: $f(\rho_{\text{in}}(R)x) = f(x)$. Any finite-dimensional representation of a compact group can be decomposed into a direct sum of irreducible representations (irreps), which define the basic transformation types of the features. The trivial irrep ρ_0 , for

which $\rho_0(R) = 1$, is a scalar unchanged by rotation (a type-0 track); the standard irrep ρ_1 , for which $\rho_1(R) = R$, is a vector in $\mathbb{R}^3$ that co-rotates with the scene (a type-1 track). Higher-order irreps exist but are not used here. It is worth noting that, while the action head is exactly $\text{SO}(3)$ -equivariant, the canonicalizer operates on gravity-axis rotations $R_z(\theta)$, with $R_z(\theta) \cong \text{SO}(2) \subset \text{SO}(3)$. Accordingly, our end-to-end equivariance concerns this subgroup.

Flow matching: The action chunk is generated by a conditional model trained with flow matching [17], which learns a velocity field that transports noise toward the data distribution along near-straight paths. Given a ground-truth chunk A_t , a noise sample $\epsilon \sim \mathcal{N}(0, I)$ and a flow time $\tau \in [0, 1]$, we form the interpolant $Z_\tau = \tau A_t + (1 - \tau)\epsilon$ and train $v_\phi(Z_\tau, S_t, \tau, \mathcal{C}_t)$ to predict the constant velocity $A_t - \epsilon$ that connects the two endpoints:

$$\mathcal{L} = \mathbb{E}_{A_t, \epsilon, \tau} \|v_\phi(Z_\tau, S_t, \tau, \mathcal{C}_t) - (A_t - \epsilon)\|^2, \quad (1)$$

where S_t is the typed state representation derived from s_t and $\mathcal{C}_t$ is the condition of the flow velocity field. At inference we sample $Z_0 = \epsilon \sim \mathcal{N}(0, I)$ and integrate the learned velocity field with K steps,

$$Z_{k+1} = Z_k + \frac{1}{K} v_\phi(Z_k, S_t, \tau_k, \mathcal{C}_t), \quad \tau_k = k/K, \quad (2)$$

then Z_K is the predicted action chunk.

Let ρ_A , ρ_S , and ρ_C be the rotation representations on action chunks, states, and conditions. For any $R \in \text{SO}(3)$, the design target is

$$\begin{aligned} v_\phi(\rho_A(R)Z_\tau, \rho_S(R)S_t, \tau, \rho_C(R)\mathcal{C}_t) \\ = \rho_A(R)v_\phi(Z_\tau, S_t, \tau, \mathcal{C}_t). \end{aligned} \quad (3)$$

IV. METHOD

In this section, we will describe the design of SAVLA. Our method builds upon VLA policies that combine a pretrained vision-language backbone with a flow-matching action head [7]. Starting from this architecture, our key insight is that equivariance can be introduced without modifying the frozen, non-equivariant backbone. We treat the backbone tokens as type-0 conditions, construct type-1 geometric conditions from a set of fixed reference points. By building the flow-matching velocity network entirely from symmetry-preserving operations, we make every integration update equivariant by construction. To handle RGB observations captured from an oblique viewpoint, SAVLA uses a learned canonicalizer that maps the third-person image into a canonical view and rotates the geometric conditions consistently. Figure 1 gives an overview of the full architecture.

A. Equivariant Action Head

The flow-matching sampler starts from an isotropic Gaussian action chunk $Z_0 \sim \mathcal{N}(0, I)$ and transforms it into Z_K through K explicit Euler updates. The distribution of Z_0 is invariant under $\rho_A(R)$. Given this invariant initialization, if the learned velocity field satisfies Eq. (3), the resulting

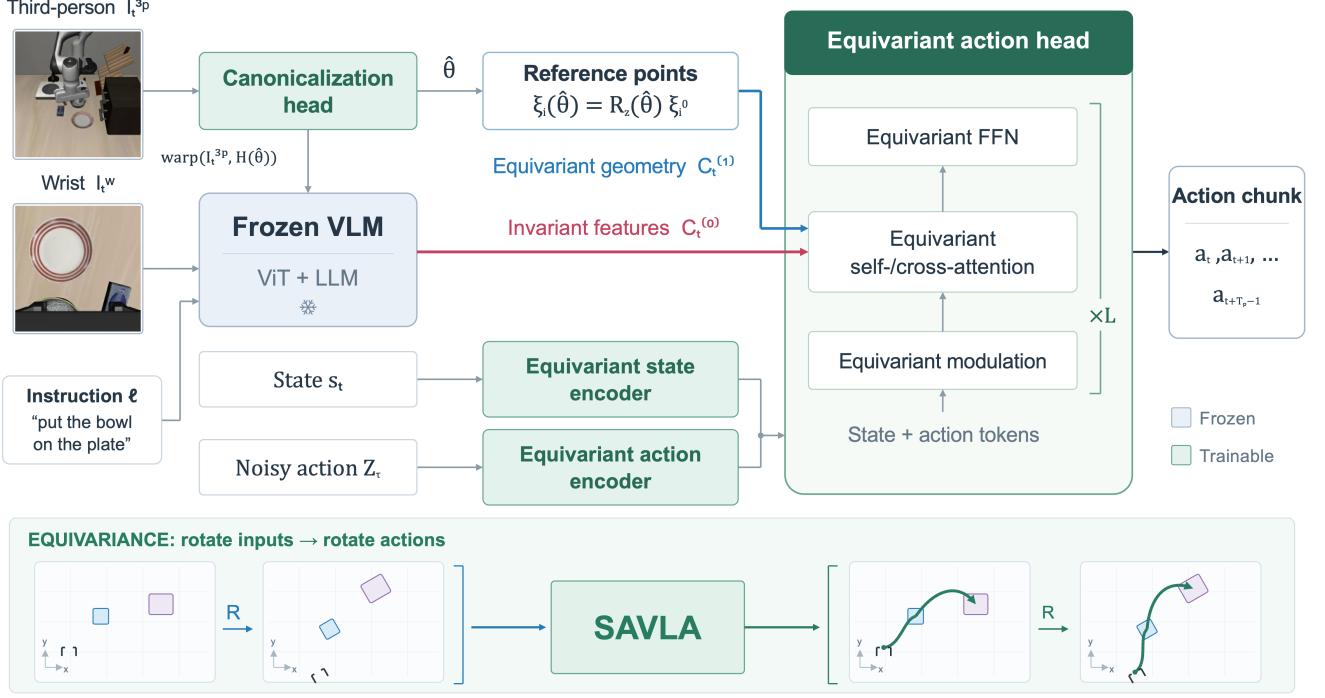


Fig. 1. SAVLA architecture. The canonicalization head estimates the scene rotation $\hat{\theta}$ from the third-person image, constructs the canonical view $\text{warp}(I_t^{3p}, H(\hat{\theta}))$ for the frozen VLM, and rotates the fixed reference points. Equivariant encoders embed the state and the noisy action. Equivariant flow-matching head outputs the action chunk.

flow preserves equivariance throughout the integration. Let $Z'_k = \rho_A(R)Z_k$ be the corresponding rotated trajectory, then $Z'_{k+1} = \rho_A(R)Z_k + \frac{1}{K}v_\phi(\rho_A(R)Z_k, \rho_S(R)S_t, \tau_k, \rho_C(R)C_t) = \rho_A(R)[Z_k + \frac{1}{K}v_\phi(Z_k, S_t, \tau_k, C_t)] = \rho_A(R)Z_{k+1}$. (4)

The rest of this section describes how each layer makes the velocity field satisfy Eq. (3).

To construct such an equivariant velocity field, each hidden token in the velocity network is split into a scalar track $h \in \mathbb{R}^{C_0}$ and a vector track $V = (v_1, \dots, v_{C_1}) \in \mathbb{R}^{C_1 \times 3}$, where C_0 and C_1 denote the numbers of type-0 and type-1 channels. Following the definitions in Section III, h is invariant under rotation, whereas every $v_c \in \mathbb{R}^3$ co-rotates with the input.

Action and state representation. Before describing the internal layers, we first specify how the state and action are mapped to these two transformation tracks. The raw 8-D state contains end-effector position p , absolute orientation as an axis-angle vector, and two gripper joint positions s_{grip} . We center position about the workspace rotation center c_{ws} , convert orientation to a rotation matrix $R_{\text{ee}} = [r_1 \ r_2 \ r_3] \in \text{SO}(3)$, and use its first two columns r_1, r_2 . We also append the known gravity direction $g_{\text{grav}} = (0, 0, 1)$. Thus the typed state is $[p - c_{\text{ws}}, r_1, r_2, g_{\text{grav}}] \oplus s_{\text{grip}}$, with signature $4\rho_1 \oplus 2\rho_0$. The action $a_t = (\Delta p_t, \omega_t, u_t^{\text{grip}})$ comprises a world-frame position increment, an incremental axis-angle rotation vector and a scalar gripper command. It has signature $2\rho_1 \oplus \rho_0$.

The ρ_0 components are embedded by ordinary MLPs, while ρ_1 components are embedded by Vector Neuron [24] $V' = WV$, which operate only on the channel dimension, leaving the three spatial components unchanged.

The resulting state and action embeddings form the input tokens of the velocity network. They are processed by L repeated blocks, each consisting of a normalization and modulation module, an attention module, and an FFN module. Next, we will show how these operations preserve the scalar and vector transformation types.

Normalization and modulation: Each block first normalizes and modulates its input while maintaining the transformation behavior of each track. The scalar track uses LayerNorm, whereas the vector track uses an equivariant RMS norm: let $q = \text{RMS}_{m'}(\|v_{m'}\|)$ be its invariant scale, then the normalized vector feature is $\tilde{v}_m = v_m/q$. For modulation, we use an equivariant modulation that changes the magnitude of each feature vector without altering its direction. Let $g = (g_1, \dots, g_{C_1}) \in \mathbb{R}^{C_1}$ be the vector of gates produced by an ordinary MLP of the flow-timestep embedding. For each channel m , the modulated vector output is

$$v'_m = (1 + g_m) \tilde{v}_m. \quad (5)$$

The scalar track uses standard modulation. Let $\gamma_0, \beta_0 \in \mathbb{R}^{C_0}$ be the modulation parameters produced by an MLP of the flow-timestep embedding. The modulated scalar output

is:

$$h' = \text{LN}(h) \odot (1 + \gamma_0) + \beta_0, \quad (6)$$

Equivariant attention: The action head uses self-attention among the input tokens (state tokens + action tokens) and cross-attention from the input tokens to the condition tokens. The state and action tokens are typed by their respective encoders, and the condition tokens are typed by the interface of Section IV-B. The Q/K/V projections are applied per track: an ordinary linear layer on the scalar part, a Vector Neuron layer on the vector part. Each query and key carries both a scalar and a vector component. A rotation acts on these features through a block-diagonal orthogonal transformation, which preserves the inner products $\langle q, k \rangle$ and hence leaves the attention weights unchanged. We sum them into a single attention weight:

$$\begin{aligned} \text{score}_{ij} &= \frac{\langle q_i^s, k_j^s \rangle}{\sqrt{d_s}} + \lambda_h \sum_m \frac{\langle q_i^{V,m}, k_j^{V,m} \rangle}{\sqrt{3d_v}}, \\ \alpha_{ij} &= \text{softmax}_j(\text{score}_{ij}), \end{aligned} \quad (7)$$

where $d_s = C_0/N_h$, $d_v = C_1/N_h$ are the scalar-channel and vector-channel widths per head, N_h is the number of attention heads, $m = 1, \dots, d_v$, and λ_h is a learnable parameter for head h . Since α_{ij} is invariant, aggregation with the same α on both tracks returns an invariant scalar output and an equivariant vector output: an invariant-weighted sum of equivariant values is equivariant. In self-attention both sides are input tokens; in cross-attention the keys and values come from the condition. Multi-head splitting operates along the channel dimension. Since scalar and vector channels are partitioned across heads, each head remains equivariant.

Equivariant FFN: After attention, each token is processed by a pointwise FFN. The scalar track uses a standard MLP with GELU, but applying the same nonlinearity to the vector track would break equivariance: $\text{GELU}(Rv) \neq R \text{GELU}(v)$. To apply a GELU on the vector track without breaking equivariance, we use an equivariant neural network to produce a per-channel direction. Let W_k be the equivariant neural network, then $k_c = (W_k V)_c$ is the direction for channel c , and $\hat{k}_c = k_c / \|k_c\|$ is the corresponding unit vector. Each feature vector is decomposed into components parallel and perpendicular to $\hat{k}_c$, and only the parallel part passes through the GELU while the perpendicular component remains unchanged:

$$v'_c = \text{GELU}(\langle v_c, \hat{k}_c \rangle) \hat{k}_c + (v_c - \langle v_c, \hat{k}_c \rangle \hat{k}_c). \quad (8)$$

Since $k_c \mapsto Rk_c$ under rotation and R preserves both $\|k_c\|$ and the inner product $\langle v_c, \hat{k}_c \rangle$, the parallel and perpendicular components in Eq. (8) transform equivariantly under R . The complete FFN applies this activation between the up- and down-projections.

Cross-track interaction: Although the two tracks follow different transformation rules, they can still exchange information through symmetry-preserving cross-track operations: vector $\rightarrow$ scalar through the per-channel invariant norms $\text{Inv}(V)$, concatenated into the scalar track at every block;

scalar $\rightarrow$ vector through invariant coefficients, which rescale equivariant quantities without altering their directions.

Since each layer preserves the transformation law, the entire head satisfies Eq. (3).

An equivariant action head alone, however, is insufficient to make the complete VLA equivariant, because the frozen vision-language backbone has no prescribed transformation law under rotations. Sections IV-B and IV-C therefore introduce a typed conditioning interface and a learned canonicalizer to extend the head-level equivariance guarantee to approximate end-to-end rotational equivariance.

B. Symmetry-aware Conditions

The equivariance guarantee of Section IV-A assumes that the condition obeys the transformation law. Since the frozen VLM does not impose such a transformation law on its output tokens, we define a typed interface that provides the action head with invariant and equivariant conditioning tracks. The construction below assumes that the canonicalizer provides a canonical third-person image and a scene-rotation estimate $\hat{\theta}$; Section IV-C describes how these quantities are obtained.

Invariant tracks: Since the the third-person image is canonicalized, with the wrist image and language instruction remaining invariant to scene rotations, the resulting VLM features are approximately invariant to such rotations. Let N_c denote the total number of condition tokens and D_0 the scalar-channel width. Accordingly, the scalar conditioning track is represented as $C_t^{(0)} \in \mathbb{R}^{N_c \times D_0}$.

Equivariant tracks: A set of fixed reference points $X = \{\xi_i^0\}_{i=1}^{N_{3p}}$ is precomputed by back-projecting the ViT patch centers through the camera onto the tabletop plane, where N_{3p} is the number of third-person visual tokens. Each ξ_i^0 is rotated by the scene rotation estimate $\hat{\theta}$ from the canonicalizer:

$$\xi_i(\hat{\theta}) = R_z(\hat{\theta}) \xi_i^0, \quad (9)$$

where $R_z(\hat{\theta}) \in \text{SO}(2)$ is the rotation by $\hat{\theta}$ about the gravity axis. Each third-person visual token is therefore paired with an equivariant location. Aligning the vector conditioning track with the scalar one, we define

$$(C_t^{(1)})_j = \begin{cases} \xi_i(\hat{\theta}), & \text{if token } j \text{ is the } i\text{th} \\ & \text{third-person visual token} \\ \mathbf{0}, & \text{if token } j \text{ is a wrist-camera} \\ & \text{or language token} \end{cases} \quad (10)$$

Together, the two tracks form the complete condition of the flow velocity field, $C_t = (C_t^{(0)}, C_t^{(1)})$. At interface, we define its group action as $\rho_C(R)C_t = (C_t^{(0)}, RC_t^{(1)})$.

C. Canonicalization

Section IV-B requires the third-person observation to be mapped to an approximately rotation-invariant canonical view, while geometric reference points co-rotate with the scene. Canonicalization satisfies these requirements by using the same estimated rotation to warp the image and transform the reference points. In most manipulation setups, the third-person camera observes the table from an oblique rather than a top-down viewpoint. Under an oblique view, rotating the

scene about the gravity axis is not an in-plane image rotation, and naively rotating the image about its center misaligns the scene. To present the frozen backbone with a canonical view, we therefore warp the image by the homography induced by the dominant tabletop plane [25]. Let $H(\theta)$ denote the homography from the canonical image to the image obtained after a scene rotation θ :

$$H(\theta) = K_{\text{cam}} \left(R_\theta + \frac{1}{d} t_\theta n^\top \right) K_{\text{cam}}^{-1}, \quad (11)$$

where K_{cam} is the camera intrinsic matrix, (R_θ, t_θ) is the relative camera motion equivalent to the scene rotation, and (n, d) parameterizes the tabletop plane in the camera frame. These quantities are computed from the simulator camera specification and table geometry rather than estimated from demonstrations. We denote by $\text{warp}(x, H)$ the inverse-sampling image warp defined by $\text{warp}(x, H)(u) = x(Hu)$, i.e., each output pixel at u samples the source image at the corresponding location Hu . Under this convention, applying $\text{warp}(x_\theta, H(\theta))$ to an observation at rotation θ maps its content back to the canonical view.

To get the scene rotation estimate $\hat{\theta}$, we use a three-layer CNN with global average pooling and a linear head (0.09 million parameters in total) to score $M = 8$ candidates $\theta_k = 2\pi k/M$, for $k = 0, \dots, M-1$. For each candidate, the canonical view is constructed as

$$x_k = \text{warp}(x, H(\theta_k)). \quad (12)$$

Let p_k be the softmax probability of candidate θ_k . The continuous estimate $\hat{\theta}$ is obtained by the circular mean

$$\hat{\theta} = \text{atan2} \left(\sum_k p_k \sin \theta_k, \sum_k p_k \cos \theta_k \right). \quad (13)$$

The scorer is trained in a self-supervised manner using the original demonstrations. Each demonstration frame x is treated as the canonical pose. We sample a rotation offset $\Delta \sim \mathcal{U}(-\pi, \pi)$ and generate a synthetic frame $x_\Delta = \text{warp}(x, H(\Delta)^{-1})$. Let $\hat{\theta}_\Delta$ denote the angle predicted from the synthetically rotated frame x_Δ , while $\hat{\theta}_0$ is the prediction from the original demonstration frame x . We train the scorer with

$$\mathcal{L}_{\text{canon}} = \mathbb{E}_{x, \Delta} \left[(1 - \cos \hat{\theta}_0) + \lambda (1 - \cos(\hat{\theta}_\Delta - \hat{\theta}_0 - \Delta)) \right], \quad (14)$$

In the second term, gradients through $\hat{\theta}_0$ are stopped. To calibrate the scale of the estimated rotation magnitude, we then introduce a scalar gain using $N_{\text{cal}} = 12$ simulator frames rendered at known angles, which are disjoint from the evaluation set. At inference, the canonicalized third-person image $\text{warp}(x, H(\hat{\theta}))$ is passed to the frozen VLM, while the same estimate $\hat{\theta}$ rotates the fixed reference points in Eq. (9).

Consequently, canonicalization makes the third-person image approximately invariant under rotations, while the rotated reference points provide equivariant geometric conditions. Together with the equivariant action head, SAVLA achieves approximate end-to-end rotational equivariance.

V. EXPERIMENTS

Our experiments are designed to evaluate SAVLA along three main dimensions: performance on standard language-conditioned manipulation tasks, robustness to unseen scene rotations, and data efficiency under reduced demonstration budgets. We use all four LIBERO suites as the primary benchmark. First, we assess standard task performance, followed by an evaluation of rotation generalization under rotations. To better understand the rotation result, we compare against rotation augmentation, ablate the equivariant action head and canonicalization module, and use oracle variants to locate the remaining error. In addition, we study data efficiency by reducing the available demonstrations. Finally, to assess its effectiveness in real-world settings, we evaluate SAVLA on three real-world manipulation tasks. Since SAVLA is built on GR00T N1.5, we use a budget-matched GR00T N1.5 as our primary baseline.

For a controlled comparison, SAVLA and the GR00T N1.5 baseline use the same frozen vision-language backbone, demonstrations, and checkpoint-selection procedure, with both action heads trained from scratch. SAVLA uses $L = 16$ blocks, typed widths $C_0 = C_1 = 256$, 8 attention heads, an action horizon $T_p = 16$, and $K = 4$ flow-integration steps. To match the parameter count of SAVLA, we reduce the official GR00T N1.5 flow-matching head to 7 attention heads with a hidden width of 192 and use a single-layer VLM-feature self-attention encoder, yielding 100.8M trainable parameters compared with 103.3M for SAVLA. Both models are optimized with AdamW using a learning rate of 10^{-4} and a batch size of 16. For each suite, checkpoints at {20K, 40K, 80K} steps are screened using the same 200-episode preliminary evaluation, and the best checkpoint is used for reporting. The canonicalizer is trained separately for 3,000 steps using AdamW with a learning rate of 10^{-4} and a batch size of 32.

A. LIBERO Experimental Results

We evaluate SAVLA on all four LIBERO suites. Each model is evaluated on 500 episodes per suite, and we report the mean over three random seeds. As shown in Table I, SAVLA outperforms the GR00T N1.5 baseline on all four suites, increasing the average success rate from 86.5% to 91.6% (+5.1 points). The results indicate that

TABLE I
EXPERIMENTAL RESULTS ON LIBERO. EXTERNAL BASELINE RESULTS ARE TAKEN FROM THE ORIGINAL PAPERS OR SUBSEQUENT PUBLISHED EVALUATIONS.

Method	Spatial	Object	Goal	Long	Average
Diffusion Policy [15]	78.3	92.5	68.3	50.5	72.4
Octo [13]	78.9	85.7	84.6	51.1	75.1
OpenVLA [2]	84.7	88.4	79.2	53.7	76.5
SmolVLA [6]	93.0	94.0	91.0	77.0	88.8
GR00T N1.5	95.3	92.4	91.6	66.7	86.5
SAVLA (ours)	96.7	92.9	97.5	79.3	91.6

TABLE II
SUCCESS RATES (%) ON ROTATION AUGMENTATION VARIANTS.

Method	$-30^\circ, \dots, +30^\circ$	$\pm 35/45^\circ$
GR00T N1.5	41.5	1.9
GR00T N1.5 + rotation aug. (warped)	69.1	24.0
GR00T N1.5 + rotation aug. (re-rendered)	74.5	59.2
SAVLA (ours)	90.4	64.5

the symmetry-aware architecture can improve standard in-distribution manipulation performance. The largest gains occur on Goal and Long, with improvements of 5.9 and 12.6 points. The consistently lower scores on Long across methods reflect its longer, multi-stage task horizons, over which execution errors can accumulate [26]. We exclude some baselines that use substantially larger models or additional pretraining on robot data to improve comparability.

B. Rotation Generalization

Rotation Robustness: We analyze rotation robustness on LIBERO by rotating the entire scene (the table layout, the objects, and the robot end-effector state) about the gravity axis through the workspace center by $\theta \in \{-30^\circ, \dots, +30^\circ\}$ in 5° steps during evaluation. This range introduces unseen scene orientations while keeping the task semantics, robot reachability, and camera visibility comparable to the original benchmark. We implement the scene rotation by orbiting the third-person camera by $-\theta$, transforming the proprioceptive state into the rotated frame, and mapping the predicted actions back to the original simulator frame before execution. Both methods are trained only on the original dataset. At each of the 13 scene rotations, we evaluate 200 episodes using three random seeds. Figure 2 shows that SAVLA substantially improves rotation robustness on Spatial, Object, and Goal, increasing the mean success rates over the 13 angles from 40.5% to 81.8%, 26.0% to 53.9%, and 41.5% to 90.4%, respectively. The smaller improvement on Long (22.7% to 25.4%) may be attributed to its longer, multi-stage horizon, which compounds execution errors beyond what rotational equivariance alone can address. We conduct the following augmentation and ablation studies on LIBERO-Goal as a representative suite.

Rotation Augmentation: A natural alternative to building symmetry into the architecture is to learn it from data. Both augmentation variants consistently transform the third-person image, proprioceptive state, and actions. In the warped variant, we sample $\theta \sim \mathcal{U}(-30^\circ, +30^\circ)$ for each training sample and warp the third-person image using the homography in Eq. (11). In the re-rendered variant, we render the third-person image from rotated camera viewpoints at the same 13 discrete angles used in the rotation experiment. Both variants are trained with the same recipe and budget as the unaugmented baseline.

Table II shows that warped rotation augmentation improves the unaugmented GR00T N1.5 baseline across both reported metrics. The re-rendered variant further improves rotation robustness, achieving a mean success rate of 74.5% across rotations from -30° to 30° and 59.2% at the extrapolation

TABLE III
REAL-ROBOT SUCCESS RATES (%) OVER 20 EPISODES PER TASK.

Method	Pick and place	Block stacking	Block sorting	Avg.
GR00T N1.5	70	60	55	61.7
SAVLA (ours)	75	70	70	71.7

TABLE IV
SUCCESS RATES (%) ON ORACLE VARIANTS.

Variant	$-30^\circ, \dots, +30^\circ$	$ \theta \geq 25^\circ$
SAVLA (estimated rotation angle, warped image)	90.4	82.1
+ ground-truth angle	90.9	83.4
+ true canonical view	97.3	96.8

angles. Nevertheless, SAVLA achieves higher success rates of 90.4% and 64.5% without additional rotated demonstration trajectories, highlighting the advantage of our symmetry-aware design over rotation augmentation in this setting.

C. Ablation Study

To locate where the rotation robustness comes from, we ablate both components. For GR00T N1.5, we apply a full canonicalization wrapper: the same canonicalizer normalizes the third-person image and provides the rotation estimate used to transform the robot state into the canonical frame. The standard action head predicts actions in this frame, which are then transformed back to the execution frame. Conversely, we remove the canonicalization from SAVLA by forcing $\theta = 0$. All variants in this section are evaluated over 200 episodes with three random seeds.

Figure 3 shows that neither component is sufficient on its own: adding canonicalization to GR00T N1.5 reaches 63.5%, while removing canonicalization from SAVLA yields 46.0%, compared with 90.4% for the complete model. The equivariant action head and canonicalization module are therefore jointly necessary for the rotation robustness. We next use two oracle variants to determine which part of the system accounts for its remaining error.

As shown in Table IV, replacing the estimated rotation angle by the ground-truth angle, while leaving the warp untouched, improves the mean success rate across -30° to 30° from 90.4% to 90.9%. With the true canonical view in place of the warped image, the mean rises to 97.3% and remains at 96.8% even when $|\theta| \geq 25^\circ$. These results suggest that the remaining headroom on LIBERO-Goal lies primarily in front-end image geometry rather than scene rotation estimation.

D. Data Efficiency

In addition to rotation robustness, we further investigate whether SAVLA maintains its advantage when fewer demonstrations are available. We re-train the models and the canonicalizer on subsets of the demonstrations (10% and 25% per task, fixed seed, identical subsets for both methods) and evaluate over 200 episodes with three random seeds, while

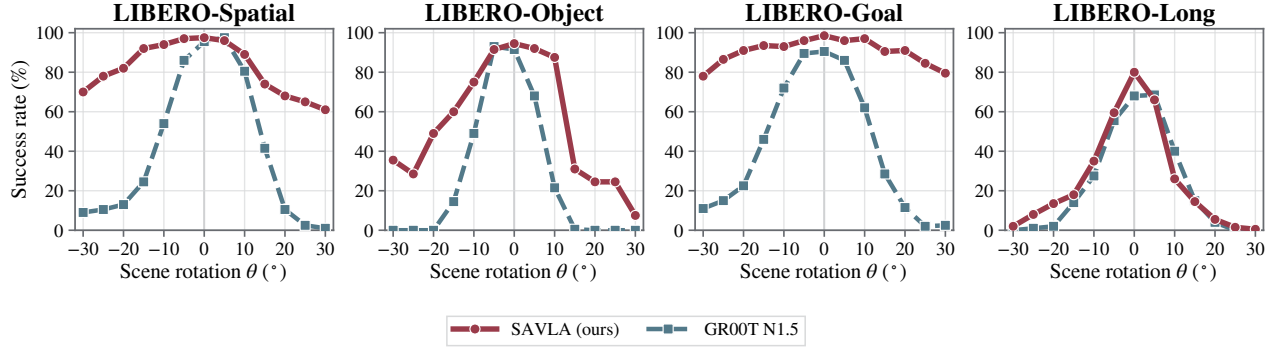


Fig. 2. Rotation-OOD across all four LIBERO suites. Each panel compares SAVLA with GR00T N1.5 over the same scene-rotation sweep. Mean over the 13 angles (SAVLA vs. GR00T N1.5): Spatial 81.8% vs. 40.5%, Object 53.9% vs. 26.0%, Goal 90.4% vs. 41.5%, and Long 25.4% vs. 22.7%.

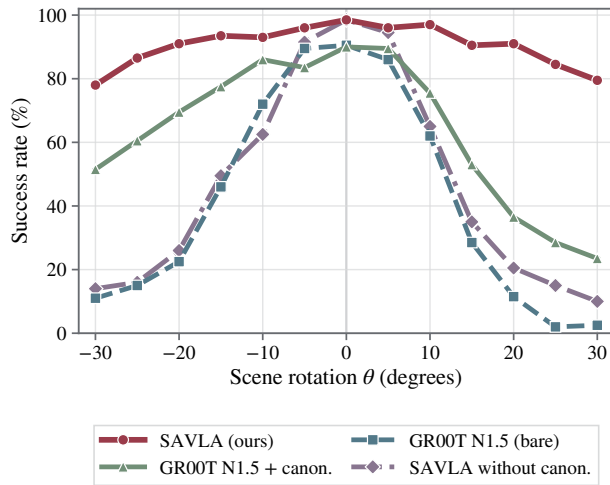


Fig. 3. Ablation on the LIBERO-Goal rotation experiment. Means over the 13 angles: SAVLA 90.4, SAVLA without canonicalization 46.0, GR00T N1.5 + canonicalization (full) 63.5, GR00T N1.5 41.5.

TABLE V

DATA EFFICIENCY ON LIBERO (AVERAGE SUCCESS RATE % OVER THE FOUR SUITES)

Method	10% data	25% data	100% data
GR00T N1.5	48.3	66.1	86.5
SAVLA (ours)	57.1	71.9	91.6
Δ	+8.8	+5.8	+5.1

the 100% result is taken from Table I. As shown in Table V, the gap increases from 5.1 points with the full dataset to 5.8 points at 25% data and 8.8 points at 10% data, showing that the equivariance structure is still beneficial under a low demonstration budget.

E. Real Robot Experiment

To test the real-world applicability, we evaluate SAVLA and GR00T N1.5 on the SO-101 robot across three real-robot manipulation tasks: pick-and-place, block stacking, and block

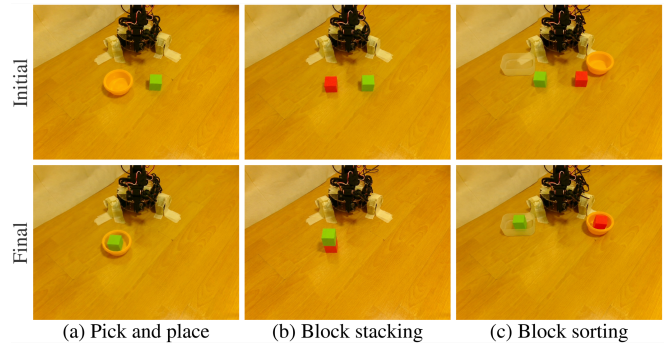


Fig. 4. Initial and final frame of the three real-robot tasks.

sorting. Following the simulation setup, both models use observations from a third-person camera and a wrist camera. Figure 4 shows the initial and final frame of three real-robot tasks. We collect 50 demonstrations per task, yielding 150 demonstrations in total. Both models are trained in a multi-task setting on demonstrations from all three tasks for 40K steps, using a batch size of 16 and a learning rate of 1×10^{-4} . We evaluate each task over 20 episodes and report the success rate.

As shown in Table III, SAVLA outperforms GR00T N1.5 on all three real-world tasks, increasing the mean success rate from 61.7% to 71.7%. The result is consistent with our simulation findings, indicating that SAVLA also improves manipulation performance in real-world settings.

We further evaluate rotation robustness on pick-and-place at $\pm 10^\circ$, $\pm 20^\circ$, and $\pm 30^\circ$, using 20 episodes per angle. We use the same camera-orbit and state/action coordinate transformations as in Section V-B. As shown in Table VI, SAVLA increases the mean success rate across the seven angles from 47.9% to 57.1%, demonstrating stronger rotation robustness in the real-world experiments.

VI. CONCLUSION

In this paper, we propose SAVLA, a symmetry-aware vision-language-action model that builds rotational symmetry into its architecture instead of learning it from demonstrations.

TABLE VI

REAL-ROBOT PICK-AND-PLACE SUCCESS RATES (%) UNDER ROTATION.

Method	-30°	-20°	-10°	0°	10°	20°	30°	Avg.
GR00T N1.5	35	45	55	70	55	45	30	47.9
SAVLA (ours)	45	55	60	75	65	55	45	57.1

On top of a frozen pretrained vision-language backbone, we design an action head that is equivariant by construction, a typed interface that generates invariant and equivariant conditions, and a canonicalization head that brings the oblique view into a canonical frame. Together these components form an end-to-end symmetry-aware VLA. On the four LIBERO suites, SAVLA outperforms GR00T N1.5 by 5.1 points on average, and the gap widens as the demonstration budget shrinks. Under scene rotations on LIBERO-Goal, SAVLA improves the mean success rate from 41.5% to 90.4%.

Several limitations remain. Although the action head itself is equivariant to the full rotation group $SO(3)$, the canonicalization head estimates only rotation about the gravity axis, and our rotation experiments cover only such rotations. Moreover, the performance drop at large angles stems from the approximate canonicalization, whose parallax and field-of-view errors grow with the angle.

IMPACT STATEMENT

This paper presents work whose goal is to advance the field of machine learning and robot manipulation. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

ACKNOWLEDGEMENT

Generative AI tools, including OpenAI Codex, were used solely for language polishing and assistance with code implementation and debugging. All AI-assisted content and code were reviewed and verified by the authors, who take full responsibility for the manuscript and released artifacts.

REFERENCES

- [1] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, *et al.*, “RT-2: Vision-language-action models transfer web knowledge to robotic control,” in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 229, 2023, pp. 2165–2183.
- [2] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, P. Sanketi, *et al.*, “OpenVLA: An open-source vision-language-action model,” in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 270, 2024, pp. 2679–2713.
- [3] K. Black, N. Brown, D. Driess, A. Esmail, M. R. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, *et al.*, “ π_0 : A vision-language-action flow model for general robot control,” in *Robotics: Science and Systems (RSS)*, 2025.
- [4] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, *et al.*, “GR00T N1: An open foundation model for generalist humanoid robots,” *arXiv preprint arXiv:2503.14734*, 2025.
- [5] M. J. Kim, C. Finn, and P. Liang, “Fine-tuning vision-language-action models: Optimizing speed and success,” in *Robotics: Science and Systems (RSS)*, 2025, arXiv:2502.19645.
- [6] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, *et al.*, “SmolVLA: A vision-language-action model for affordable and efficient robotics,” *arXiv preprint arXiv:2506.01844*, 2025.

- [7] NVIDIA GEAR Lab, “GR00T N1.5: An improved open foundation model for generalist humanoid robots,” Technical report and model release, <https://research.nvidia.com/labs/gear/gr00t-n1.5>, 2025, model weights: [nvidia/GR00T-N1.5-3B](https://research.nvidia.com/labs/gear/gr00t-n1.5-3B).
- [8] S. Fei, S. Wang, J. Shi, Z. Dai, J. Cai, P. Qian, L. Ji, X. He, S. Zhang, Z. Fei, J. Fu, J. Gong, and X. Qiu, “LIBERO-Plus: A progressive robustness benchmark for visual-language-action models,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026, pp. 38 574–38 583, arXiv:2510.13626.
- [9] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang, and X. Li, “SpatialVLA: Exploring spatial representations for visual-language-action models,” in *Robotics: Science and Systems (RSS)*, 2025, arXiv:2501.15830.
- [10] J. Yang, Z. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, “EquiBot: SIM(3)-equivariant diffusion policy for generalizable and data efficient learning,” in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 270, 2024, pp. 1048–1068.
- [11] C. Tie, Y. Chen, R. Wu, B. Dong, Z. Li, C. Gao, and H. Dong, “ET-SEED: Efficient trajectory-level SE(3) equivariant diffusion policy,” in *International Conference on Learning Representations (ICLR)*, 2025.
- [12] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, *et al.*, “RT-1: Robotics transformer for real-world control at scale,” in *Robotics: Science and Systems (RSS)*, 2023.
- [13] 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,” in *Robotics: Science and Systems (RSS)*, 2024.
- [14] T.-L. Ha, Q.-T. Nguyen, T.-B. Ho, L. Dinh, M. D. Nguyen, G.-B. Nguyen, *et al.*, “EquiVLA: A general framework for rotationally equivariant vision-language-action models,” *arXiv preprint arXiv:2606.19784*, 2026.
- [15] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, “Diffusion policy: Visuomotor policy learning via action diffusion,” in *Robotics: Science and Systems (RSS)*, 2023.
- [16] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 6840–6851.
- [17] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, “Flow matching for generative modeling,” in *International Conference on Learning Representations (ICLR)*, 2023.
- [18] D. Wang, S. Hart, D. Surovik, T. Kestemur, H. Huang, H. Zhao, M. Yeatman, J. Wang, R. Walters, and R. Platt, “Equivariant diffusion policy,” in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 270, 2024, pp. 48–69.
- [19] H. Ryu, J. Kim, H. An, J. Chang, J. Seo, T. Kim, Y. Kim, C. Hwang, J. Choi, and R. Horowitz, “Diffusion-EDFs: Bi-equivariant denoising generative modeling on SE(3) for visual robotic manipulation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 18 007–18 018.
- [20] C. Gao, Z. Xue, S. Deng, T. Liang, S. Yang, L. Shao, and H. Xu, “RiEMann: Near real-time SE(3)-equivariant robot manipulation without point cloud segmentation,” in *Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 270, 2024, pp. 2164–2182.
- [21] S.-O. Kaba, A. K. Mondal, Y. Zhang, Y. Bengio, and S. Ravanbakhsh, “Equivariance with learned canonicalization functions,” in *International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 202, 2023, pp. 15 546–15 566.
- [22] J. Kim, T. D. Nguyen, A. Suleymanzade, H. An, and S. Hong, “Learning probabilistic symmetrization for architecture agnostic equivariance,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023, pp. 18 582–18 612.
- [23] Z. Zhang, Z. Xu, J. N. Lakamsani, and Y. She, “Canonical policy: Learning canonical 3D representation for SE(3)-equivariant policy,” *arXiv preprint arXiv:2505.18474*, 2025.
- [24] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas, “Vector neurons: A general framework for SO(3)-equivariant networks,” in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 12 180–12 189.
- [25] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.
- [26] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, “LIBERO: Benchmarking knowledge transfer for lifelong robot learning,” in *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, vol. 36, 2023.