GeoAlign-WAM: A 4D World Action Model for Generalizable and Robust Robot Manipulation Using Video Foundation Prior

Zirui Li[1], Yujie Yan[1], Runjia Tan, Lei Yang, Yangang Ren, Ziwei Liu* and Chen Lv*

Author affiliations and mapping to be supplied,

ARTICLE INFO

Keywords:
Robot manipulation
World action models
Video foundation models
Pointmaps
Flow matching
Geometry–action alignment

ABSTRACT

Video foundation models provide temporal priors for robot manipulation, but plausible image prediction does not explicitly constrain the relationship between scene geometry and Cartesian control. Image pixels identify viewing rays rather than unique metric locations, whereas end-effector commands must resolve spatial relationships in the robot workspace. We present GeoAlign-WAM, a geometry-aware world action model that combines a pretrained video prior with pixel-aligned pointmaps in the robot base frame. RGB and pointmap features are encoded into compatible latent grids and concatenated along the spatial width dimension to jointly represent appearance, three-dimensional structure, and temporal evolution. An action expert predicts executable action chunks from the current multimodal context. A lightweight convolutional head reads the predicted future latent to produce an auxiliary action chunk, and a symmetric geometry–action alignment loss penalizes disagreement between the two predictions. Structured attention prevents the action branch from accessing future tokens, allowing the future-generation branch and auxiliary head to be removed at deployment. GeoAlign-WAM achieves 97.4% success on LIBERO, 81.0% on LIBERO-Plus, 83.1% on LIBERO-Occ, and 84.0% across three real-robot tasks. Compared with Fast WAM, mean success increases by 11.9 percentage points under occlusion and 13.3 points on the real robot. LIBERO-Plus results show substantial gains under camera shifts, although weaknesses remain under some language and lighting perturbations.

Project page: https://lzrbit.github.io/GeoAlign-WAM/.

1. Introduction

Robot manipulation requires policies to infer task-relevant spatial relationships from incomplete observations and coordinate motion as contact and visibility change. Hand-designed perception and control pipelines can be effective in well-characterized settings, but often require task-specific engineering. Learning-based approaches connect sensory observations to actions through experience. End-to-end visuomotor learning jointly optimizes perception and control, while subsequent imitation-learning methods model multimodal behavior and temporally extended action sequences [5, 13, 31]. Their reliability beyond the training distribution depends, in part, on which scene properties their representations preserve.

Foundation models offer representations learned from data beyond the available robot demonstrations. Vision–language–action (VLA) models transfer visual and linguistic knowledge to instruction-conditioned control [1, 12, 33]. Such knowledge supports semantic task interpretation, but does not explicitly specify the metric displacement required to approach a handle or place an object in a receptacle. Video foundation models provide complementary priors on temporal appearance, motion, and interaction. World action models (WAMs) adapt these priors by coupling world prediction with action learning [28, 29]. The world objective can supervise aspects of a demonstration that are absent from its action labels.

World objectives defined only on RGB observations or their latents leave the image-to-workspace relationship implicit. Perspective projection allows different metric configurations to produce similar image evidence, while a camera displacement can change image coordinates without requiring the corresponding change in robot-frame action. Joint video and action training may learn these relationships implicitly; however, a shared transformer does not by itself establish calibrated correspondence between image locations and the robot workspace. We term this limitation a *geometric grounding gap*; it does not preclude learning spatial information from images.

*Corresponding authors.
ziwei.liu@ntu.edu.sg (Z. Liu); lyuchen@ntu.edu.sg (C. Lv)
[1]Zirui Li and Yujie Yan contributed equally to this work.

This gap can affect generalization when viewpoints, object appearances, or backgrounds change without altering the required interaction. Partial occlusion further reduces evidence about object boundaries and approach directions. Explicit geometry provides an additional representation of visible spatial structure that may help disambiguate these observations. Its usefulness nevertheless depends on how it informs action selection: geometry alone neither determines hidden surfaces nor guarantees robustness.

Recent spatial WAMs incorporate future RGB-D prediction, as in X-WAM, or transfer geometric supervision through spatial register tokens, as in WAM4D [7, 17]. Building on these developments, we investigate an explicit action-level constraint: *the action inferred from a predicted geometric world representation should agree with the action produced by the deployed policy*. This motivates joint geometric representation learning and action consistency within one architecture.

We introduce GeoAlign-WAM, a 4D world action model with geometry–action alignment. Here, "4D" denotes a time-indexed sequence of three-dimensional scene representations. The model receives RGB observations and pixel-aligned XYZ pointmaps in the robot base frame. A video-compatible encoding and fusion pathway supplies appearance and geometry to the world backbone, while an action expert predicts the control sequence. We adapt the multimodal representation strategy of 4DNeX [4] and the deployment structure of Fast WAM [29] to encourage action consistency in a metric workspace. A small convolutional readout maps the predicted future latent to an auxiliary action chunk. A symmetric alignment loss penalizes its disagreement with the expert's action prediction, with gradients propagated through both branches.

The auxiliary readout is used only during training. The action expert receives current observations and cannot attend to future world tokens. Deployment therefore omits future RGB/XYZ synthesis while retaining parameters trained with world prediction and action alignment. This separates training supervision from information available to the deployed policy. Pointmap sensing and preprocessing remain necessary and must be included in deployment latency.

The contributions of this work are as follows:

1. We formulate a pointmap-conditioned 4D WAM that brings metric scene geometry into the observation context and jointly learns future RGB, future XYZ, and robot actions. The representation preserves pixel correspondence while making its relationship to the robot base frame explicit.
2. We introduce a geometry–action alignment objective that compares an auxiliary action inferred from the predicted future latent with the action expert's prediction. The constraint operates on a common action representation and is compatible with a policy that does not consume future tokens at deployment.
3. We evaluate nominal performance, distribution shifts, and occlusion on LIBERO, LIBERO-Plus, and LIBERO-Occ [6, 16, 19], together with three real-robot tasks. The model reaches 97.4% on LIBERO and improves mean success over Fast WAM by 11.9 percentage points on LIBERO-Occ and 13.3 points on the real robot.

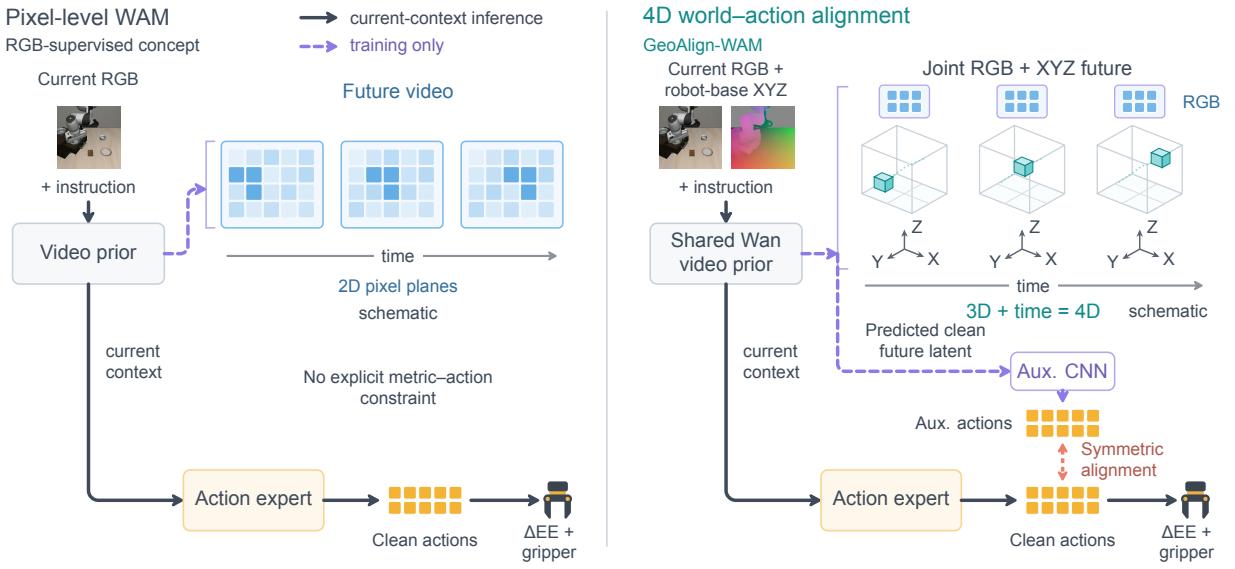


Figure 1: Pixel-level and 4D world supervision. GeoAlign-WAM represents robot-frame geometry over time and aligns future-derived auxiliary actions with the action expert. Dashed paths denote training-only computation.

2. Related works

2.1. End-to-end visuomotor and geometry-aware manipulation

End-to-end manipulation policies map sensory inputs directly to motor commands. Levine et al. jointly learned visual representations and control [13]. Subsequent imitation-learning approaches model multimodal action distributions and temporally extended commands: Diffusion Policy uses iterative action generation, while Action Chunking with Transformers predicts action chunks for fine manipulation [5, 31]. These formulations capture both action ambiguity and the temporal structure of demonstrations.

Geometry-aware policies explicitly represent spatial relationships. PerAct uses a voxel representation for language-conditioned manipulation, and 3D Diffusion Policy combines compact point clouds with generative action modeling [23, 30]. Beyond explicit 3D inputs, representation-alignment methods incorporate supervision from 3D representations into VLA features. ROCKET, for example, uses residual multi-layer alignment to learn spatially aware features [24]. Our focus is the connection between pixel-aligned geometry in a video-derived world representation and action prediction through future-latent supervision.

2.2. Vision–language–action models

VLA models integrate visual perception, language conditioning, and robot action prediction. RT-2 transfers vision–language pretraining to robotic control, and OpenVLA provides an open model and adaptation framework for generalist manipulation [12, 33]. The π_0 model combines a pretrained vision–language backbone with flow-matching action generation [1]; $\pi_{0.5}$ extends this approach toward open-world generalization through co-training on heterogeneous data [21]. OpenVLA-OFT examines how parallel decoding, action chunking, and other fine-tuning choices affect speed and success [10].

VLA and WAM formulations overlap: a language-conditioned policy can use world-prediction objectives, and a WAM can expose the same observation-to-action interface as a VLA. The relevant distinctions concern the learned representation and the information available during action inference. GeoAlign-WAM conditions on language, RGB, and current XYZ, while future-world supervision shapes its video backbone. Its alignment objective connects future features to action predictions without assuming that VLA models cannot infer geometry or dynamics.

2.3. World action models and 4D scene generation

Video-based robot models use visual prediction to supervise action learning. Unified World Models couples video and action diffusion in a shared transformer [32]. DreamZero builds a large-scale WAM around a video prior [28],

while Cosmos Policy fine-tunes a pretrained video model for visuomotor control and planning [11]. Fast WAM studies future prediction during training and removes future-video generation from deployment [29]. We adopt this separation between current-context action inference and future-world supervision, and extend it with metric geometry and an explicit action-consistency objective.

4DNeX models paired RGB and XYZ sequences in video-compatible latent space [4]. We adapt its modality-fusion strategy to robot control, where geometry must retain the coordinate frame and metric scale of the controller. Observed robot-frame pointmaps enter the conditioning context, and an auxiliary readout connects predicted future features to action space. The contribution lies in this representation-to-action connection rather than in the concatenation operation.

Among spatial WAMs, X-WAM jointly models visual, geometric, and robot information through RGB-D prediction with asynchronous denoising [7]. WAM4D uses spatial registers for geometric supervision and removes that branch at inference [17]. Our formulation combines explicit current XYZ input with future-latent-to-action consistency in a video-derived model. Controlled ablations are needed to distinguish the effects of pointmap conditioning, future XYZ prediction, and alignment.

3. Problem formulation

3.1. Observations, world sequences, and action chunks

Consider an imitation-learning dataset $\mathcal{D}$ of synchronized robot demonstrations. At environment time t, the policy receives a language instruction ℓ and an observation

$$o_t = \left\{ I_t^{(v)}, P_t^{(v)} \right\}_{v=1}^{V}, \qquad I_t^{(v)} \in \mathbb{R}^{H_I \times W_I \times 3}, \quad P_t^{(v)} \in \mathbb{R}^{H_I \times W_I \times 3}. \tag{1}$$

Here v indexes the camera view, and $P_t^{(v)}(u)$ gives the metric XYZ coordinates associated with image pixel u in the robot base frame B. RGB and XYZ share a registered image grid. When included in the selected Fast WAM configuration, robot state forms part of the current observation; future state is unavailable to the deployed policy. We omit the view index when an operation applies identically to every view.

The action chunk is $A_t = [a_t, \dots, a_{t+H_A-1}] \in \mathbb{R}^{H_A \times d_a}$. We retain the Fast WAM task interface: Cartesian increments, rotation commands, and gripper commands are controller inputs, not absolute XYZ positions. For the LIBERO interface, $d_a = 7$ and

$$a_t = [(u_t^p)^\top, (u_t^r)^\top, g_t]^\top, \qquad u_t^p, u_t^r \in \mathbb{R}^3, \tag{2}$$

where u_t^p and u_t^r are translational and axis-angle rotation increment commands, respectively, and g_t controls the gripper. Controller scaling converts these commands into pose offsets; their raw values are not metric displacements. Controller-to-base transformations must agree with the pointmap calibration. The normalization operator $\mathcal{N}_A$ maps controller commands to the action expert's training space, and its inverse is applied before execution.

The future world target is

$$W_t = \left\{ I_{t+\delta_k}, P_{t+\delta_k} \right\}_{k=1}^{T}, \qquad 0 < \delta_1 < \cdots < \delta_T, \tag{3}$$

where the offsets δ_k align video sampling with the action interval. The video horizon T need not equal the action horizon H_A. The learned policy is

$$A_t \sim \pi_\theta(\cdot \mid o_t, \ell), \tag{4}$$

and only a prefix is executed before the next observation. World prediction is an auxiliary conditional task learned from the same demonstrations. Under the attention structure defined below, the world branch does not receive candidate actions. It predicts demonstration futures conditioned on the observation and instruction, rather than simulating arbitrary action interventions.

The "4D representation" is the sequence of 3D pointmaps in Eq. (3), each describing visible surfaces in a common coordinate frame. Within-frame RGB–XYZ correspondence does not imply persistent point identities, complete reconstruction of occluded surfaces, or an explicit physical dynamics model.

3.2. Conditional flow-matching training

We formulate future RGB, future XYZ, and action prediction using conditional flow matching [18]. Let Y_m denote a normalized clean target for modality $m \in \{\text{rgb}, \text{xyz}, \text{act}\}$. For the visual modalities, Y_m is an encoded latent sequence; for actions, $Y_{\text{act}} = \mathcal{N}_A(A_t)$. For Gaussian noise $\epsilon_m \sim \mathcal{N}(0, \mathbf{I})$ and flow time $\tau_m \in (0, 1)$, define

$$Y_{m,\tau_m} = (1 - \tau_m)Y_m + \tau_m \epsilon_m, \qquad U_m = \epsilon_m - Y_m. \tag{5}$$

Thus $\tau_m = 0$ is clean data and $\tau_m = 1$ is noise. Environment time t, future-frame index k, and flow time τ_m are distinct quantities. With a velocity predictor $V_{\theta,m}$ and current context C_t, the modality objective is

$$\mathcal{L}^m_{\text{FM}} = \mathbb{E}\left[\frac{\omega_m(\tau_m)}{n_m}\left\|V_{\theta,m}(Y_{m,\tau_m}, \tau_m; C_t, \ell) - U_m\right\|^2_F\right], \tag{6}$$

where the norm is evaluated over valid, non-padded target entries, n_m is their count, and $\omega_m(\tau_m)$ denotes any baseline timestep weight (one for the unweighted objective). For RGB and XYZ, the predictor jointly reads both noisy latent grids through their fusion; the notation in Eq. (6) abbreviates this joint input. Current observation tokens remain clean and are excluded from the future-target loss.

We use a common world noise level τ_w for paired RGB and XYZ and denote the action noise level by τ_a. Their sampling distributions and noise-shift parameters are training hyperparameters. A one-step estimate of the clean target is

$$\widehat{Y}_m = Y_{m,\tau_m} - \tau_m \widehat{U}_m, \qquad \widehat{U}_m = V_{\theta,m}(\cdot). \tag{7}$$

Equation (7) recovers the clean target exactly when the predicted velocity equals U_m; otherwise it gives a one-step estimate. It places the expert and auxiliary predictions in a common action domain, avoiding a comparison between executable actions and flow velocities.

The flow-matching scheduler, world/action time sampling, temporal offsets, and action normalization follow Fast WAM [29]. RGB and XYZ share the world noise level, following the paired-modality formulation of 4DNeX [4]. The auxiliary convolutional neural network (CNN) reads the predicted clean future latent given by Eq. (7).

4. Proposed model

4.1. Overall architecture

GeoAlign-WAM contains a world backbone initialized from a video foundation model, a modality-compatible RGB/XYZ encoding pathway, an action expert, and an auxiliary convolutional action head. Figure 2 separates the components required for deployment from those used for training. The observation encoders produce a clean multimodal context C_t. During training, the world backbone also processes noisy future RGB and XYZ latents and predicts their flow velocities. The action expert predicts action velocity using the current context and noisy actions. The auxiliary head reads the estimated clean future latent and outputs an action chunk in the same normalized space as the expert.

The expert provides the deployed policy and accesses only information available at the decision time. The auxiliary head supplies an action-space readout of predicted future features, coupling the training objectives through a differentiable loss. It does not pass features to the expert, contribute an averaged action, or serve as a second controller at deployment.

The world backbone and action expert form a mixture of transformers with modality-specific transformations and structured cross-branch attention. We retain the Wan2.2-5B backbone and action-expert configuration of Fast WAM (Table 1). Controlled comparisons must hold the backbone and demonstration data fixed to isolate pointmap conditioning and alignment.

4.2. Input initialization and normalization

4.2.1. Metric pointmaps and registration

The geometric input contains metric surface coordinates in the robot base frame. A depth-derived pointmap is obtained through

$$\begin{bmatrix} P^B_t(u) \\ 1 \end{bmatrix} = \mathbf{T}^B_{C,t} \begin{bmatrix} D_t(u)\mathbf{K}^{-1}\widetilde{u} \\ 1 \end{bmatrix}, \qquad \widetilde{u} = (u_x, u_y, 1)^\top, \tag{8}$$

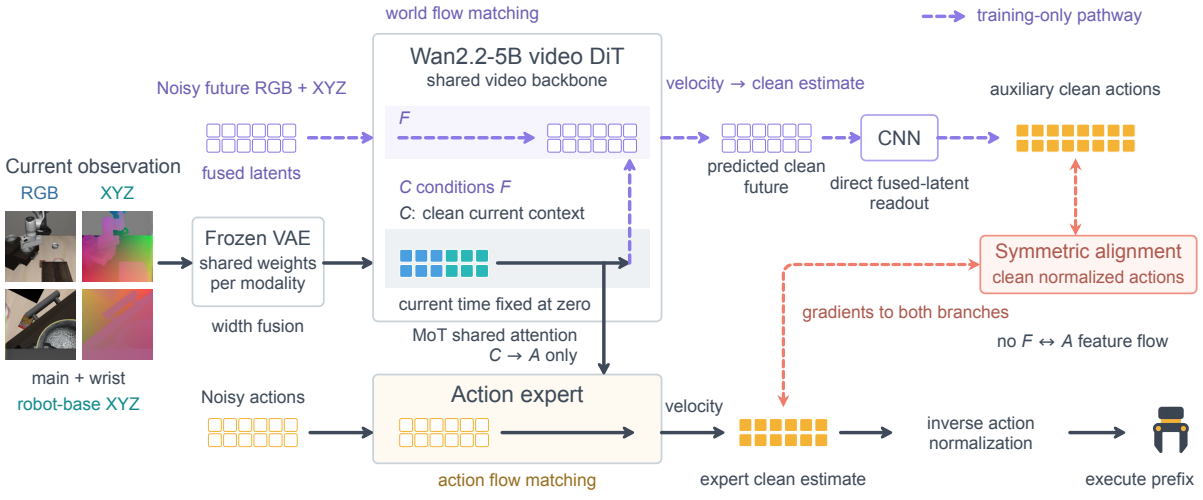


Figure 2: GeoAlign-WAM architecture. The Wan video backbone denoises future RGB/XYZ latents; an auxiliary readout aligns their action prediction with the action expert. Deployment uses only the current-context action path.

where D_t is metric depth, $\mathbf{K}$ is the camera intrinsic matrix, and $\mathbf{T}^{B}_{C,t}$ maps camera coordinates to the robot base. For a wrist camera, this transform depends on the robot configuration. A learned pointmap estimator would likewise need to resolve metric scale and frame alignment before its output could be used in this formulation.

Identical cropping, resizing, and view composition preserve correspondence within each RGB/XYZ pair. RGB images and pointmaps use the same fixed camera order. Although pointmaps and end-effector motion share a base frame, their entries represent different quantities: surface coordinates and controller commands, respectively.

4.2.2. Initialization and metric normalization

The current XYZ input is initialized from observed geometry. Unlike the synthetic sloped-plane initialization used when geometry is unavailable in image-to-4D generation, this preserves measured spatial information. The known-context mask treats valid current RGB and XYZ as conditioning data, while future targets are initialized from noise for generation. Missing or invalid depth must be handled consistently during training and deployment.

To expose XYZ to a video-compatible encoder, let

$$\overline{P} = (P^B - \boldsymbol{b}) \oslash \boldsymbol{s}, \qquad P^B = \overline{P} \odot \boldsymbol{s} + \boldsymbol{b}, \tag{9}$$

where $\boldsymbol{b} \in \mathbb{R}^3$ and $\boldsymbol{s} \in \mathbb{R}^3_{>0}$ are known offset and scale vectors that preserve recovery of the metric frame. A fixed transform estimated from training data satisfies this requirement. Per-frame centering would discard motion and location information unless its offsets were retained and supplied to the model. Online normalization must not use statistics from held-out trajectories or unavailable future frames.

RGB uses the pretrained encoder's input scaling. XYZ additionally undergoes modality-specific latent normalization, defined below. Both action heads share the baseline's controller-specific normalization. In the released Fast WAM LIBERO configuration, this is a per-component affine min/max mapping derived from training data with nominal range $[-1, 1]$. Other robot interfaces require their own controller conventions. Input, latent, and action normalization serve distinct purposes and do not replace geometric calibration.

In our implementation, $\boldsymbol{b}$ is a fixed workspace center and $\boldsymbol{s} = s\mathbf{1}$ an isotropic scale; for the LIBERO experiments $\boldsymbol{b} = (0.531,\ 0.063,\ 0.214)\,\mathrm{m}$ and $s = 0.5793\,\mathrm{m}$. Normalized coordinates are clipped to $[-1, 1]$, and pixels without valid geometry are set to zero, which is the mid-gray value of the 8-bit visualization used in our figures. In simulation, pointmaps are back-projected from rendered metric depth with the simulator's camera intrinsics and camera-to-base transforms; points farther than 2 m from the camera and points more than 5 cm below the table surface are marked invalid. On the real robot, pointmaps are computed from the RGB-D depth images with the calibrated camera poses (Section 6).

4.3. Multimodal encoding

Let $\mathcal{E}_v$ and $\mathcal{D}_v$ be the encoder and decoder of the pretrained video variational autoencoder (VAE). Following the video-compatible encoding principle of 4DNeX, RGB and normalized pointmaps are encoded in separate applications of the VAE [4]. We write $\mathcal{E}_{\text{rgb}}$ and $\mathcal{E}_{\text{xyz}}$ to distinguish the modality pathways; these symbols do not require separately pretrained encoder weights. With frozen VAE parameters, their latent sequences are

$$Z^{\text{rgb}} = \mathcal{E}_{\text{rgb}}(\overline{I}), \qquad Z^{\text{xyz}} = \mathcal{E}_{\text{xyz}}(\overline{P}), \qquad Z^m \in \mathbb{R}^{T_z \times h \times w \times c}. \tag{10}$$

Here T_z, h, w are the compressed temporal and spatial dimensions, and c is the latent channel count. The pointmap branch encodes organized XYZ images, preserving the grid structure required by the video backbone.

For XYZ latent statistics $\boldsymbol{\mu}_{\text{xyz}}$ and $\boldsymbol{\sigma}_{\text{xyz}}$ estimated on the training set, define

$$\widetilde{Z}^{\text{xyz}} = (Z^{\text{xyz}} - \boldsymbol{\mu}_{\text{xyz}}) \oslash (\boldsymbol{\sigma}_{\text{xyz}} + \varepsilon). \tag{11}$$

The statistics are broadcast over the corresponding latent channels. Decoding first inverts this normalization; Eq. (9) then restores robot-frame units. RGB latents retain the pretrained backbone's scaling. Below, both modality latents refer to their normalized forms unless stated otherwise.

Current context must be encoded independently of future frames, either through the baseline's causal observation encoder or by encoding the observation separately. Selecting the first token after non-causal encoding of an entire demonstration would violate this constraint. Training and inference use the same current-context construction, with the diffusion-time embedding fixed at the clean endpoint $\tau_C = 0$ and independent of the sampled future or action noise level.

Language embeddings condition both branches through the baseline's text pathway. The RGB–XYZ fusion and associated positional settings follow 4DNeX [4]. Physical correspondence between paired observations is distinct from their packed token positions.

4.4. Fusion strategies

The final model uses width-wise concatenation of RGB and XYZ latents:

$$Z^F = \mathcal{F}_w(Z^{\text{rgb}}, Z^{\text{xyz}}) = \text{Concat}_w(Z^{\text{rgb}}, Z^{\text{xyz}}) \in \mathbb{R}^{T_z \times h \times 2w \times c}. \tag{12}$$

The operation is applied to both current context and future targets. Each Z^m already packs the main and wrist cameras side by side, following the baseline's two-camera layout, with 224×224 pixels per panel. The four panels appear in the order RGB main, RGB wrist, XYZ main, XYZ wrist, producing a 224×896 frame (Fig. 3). This layout preserves the pretrained latent channel width and permits cross-modal attention. The packed position of an XYZ panel does not define a new camera view or physical image location. Shared rotary indices and explicit packed positions are separate positional-encoding choices.

The following alternatives define the fusion design space, following 4DNeX's modality-fusion analysis [4]:

- Channel fusion: concatenate at matching locations along the feature-channel dimension and adapt the input/output projections to the increased channel count.
- Batch fusion: process modality-labeled samples separately. Without an explicit cross-modal interaction mechanism, batch concatenation alone does not fuse information within a sample.
- Frame fusion: pack modality sequences along the temporal axis, retaining separate modality labels and the original physical timestamps.
- Height fusion: place the latent grids above and below one another while retaining their shared image coordinates.
- Width fusion: use the side-by-side layout of Eq. (12), selected for the final model.

Figure 3 summarizes the layouts; Appendix A specifies their shapes, index mappings, tokenization, and interaction operators. Channel fusion changes projection dimensions, while the other layouts may change token count or attention cost. These differences must be accounted for in controlled comparisons. Concatenation exposes paired modalities to shared computation; the action-consistency objective provides the additional task-level constraint.

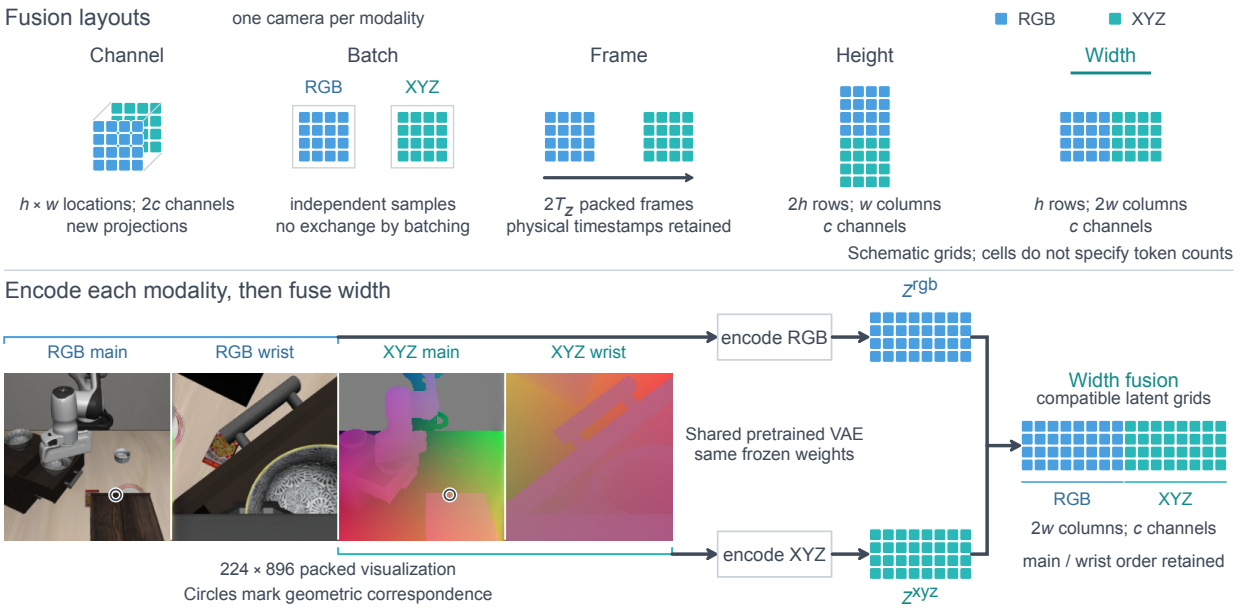


Figure 3: RGB–pointmap fusion layouts and the selected width-wise packing of two camera views. Paired RGB and XYZ retain pixel correspondence; Appendix A gives the formulations.

4.5. Causal world–action transformer

Let C denote clean current RGB/XYZ context tokens, F noisy future fused tokens, and A noisy action tokens. We extend the Fast WAM visibility structure to the multimodal context [29], as illustrated in Fig. 4. For a query-group/key-group ordering (C, F, A), the allowed-attention matrix is

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \tag{13}$$

An entry of one permits all attention edges within the corresponding block. The additive attention mask $\mathbf{M}$ is zero on permitted edges and $-\infty$ elsewhere, giving

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^{\top}}{\sqrt{d}} + \mathbf{M}\right) V. \tag{14}$$

RGB and XYZ may interact within C and within F. The action group reads C and its own chunk, while the future group reads C and future world tokens. Language conditioning is available separately to the permitted branches.

Causality separates available observations from unavailable future information; it does not require autoregressive ordering within jointly generated video or action chunks. Bidirectional attention is therefore permitted within each group. Since C cannot read F or A, additional transformer layers cannot relay future information to the action expert through the context. Preprocessing, normalization, temporal encoding, and feature pooling must obey the same visibility constraint.

The branches share context and training supervision, with their predictions coupled by the alignment loss. They do not define a general action-conditioned transition model. The mask preserves the policy's forward computation when the future branch is removed.

4.6. Geometry–action alignment and training objective

4.6.1. Auxiliary action from the predicted future latent

Equation (7) gives the normalized clean future estimates $\hat{Z}^{\mathrm{rgb}}$ and $\hat{Z}^{\mathrm{xyz}}$. The CNN directly reads their width-fused latent grid:

$$\hat{G} = \mathrm{Concat}_w(\hat{Z}^{\mathrm{rgb}}, \hat{Z}^{\mathrm{xyz}}), \qquad \hat{G} \in \mathbb{R}^{T_z \times h \times 2w \times c}. \tag{15}$$

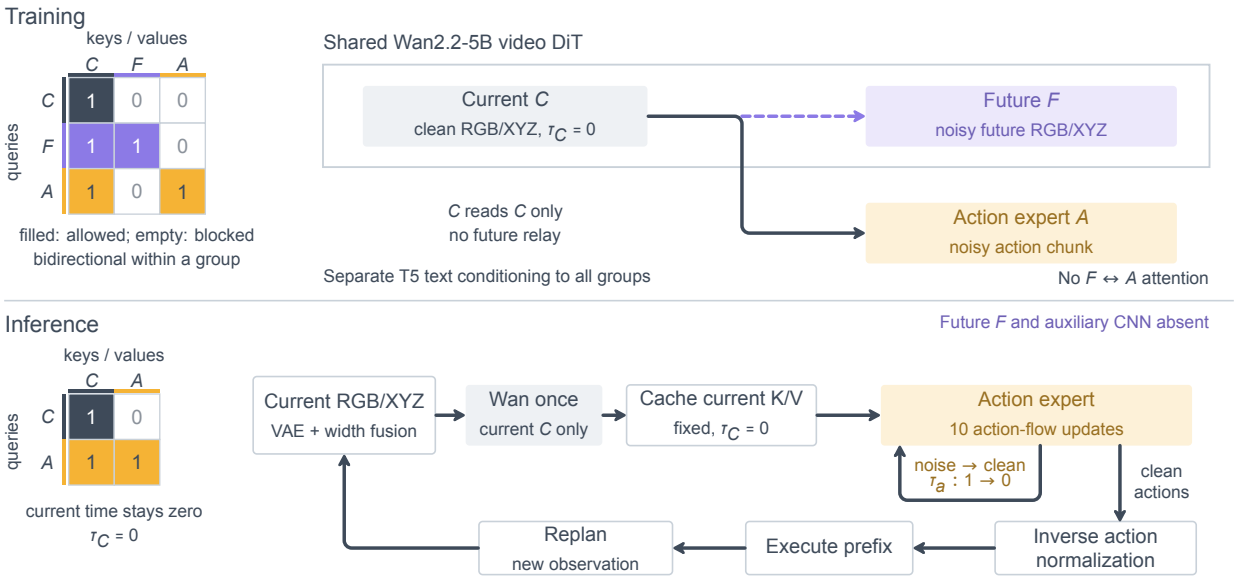


Figure 4: Training and inference attention. Training separates future-world and action tokens; inference removes future tokens and reuses cached current-context keys and values for action denoising.

The layout is retained without unpacking the modalities or concatenating them along channels. A small convolutional head $\mathcal{H}_\psi$ maps the fused future features to the action horizon:

$$\widehat{A}^{\text{aux}} = \mathcal{H}_\psi(\widehat{G}) \in \mathbb{R}^{H_A \times d_a}. \tag{16}$$

Spatial feature extraction and temporal projection accommodate different T_z and H_A. The head reads predicted world features without access to expert hidden tokens or ground-truth actions. The readout does not identify individual XYZ pixels with robot commands.

The expert's clean action estimate under the same flow convention is

$$\widehat{A}^{\text{exp}} = A_{\tau_a} - \tau_a V_{\theta,\text{act}}(A_{\tau_a}, \tau_a; C_t, \ell), \tag{17}$$

where A_{τ_a} is the noisy normalized action chunk. Both estimates share action units, channel order, horizon, and normalization. Comparing Eq. (16) with an action velocity instead would mix different prediction targets.

4.6.2. Symmetric alignment

The geometry–action alignment objective is

$$\mathcal{L}_{\text{align}} = \mathbb{E}\left[\frac{1}{H_A d_a}\left\|\widehat{A}^{\text{aux}} - \widehat{A}^{\text{exp}}\right\|_F^2\right]. \tag{18}$$

Gradients propagate through both predictions. With $\Delta = \widehat{A}^{\text{aux}} - \widehat{A}^{\text{exp}}$, the output gradients are proportional to Δ and $-\Delta$, respectively. The loss therefore updates the future-world readout, the action expert, and any shared trainable parameters. Both predictions correspond to the same demonstrated interval.

The objective penalizes disagreement in action space without guaranteeing geometric or physical consistency. The CNN may rely on appearance rather than XYZ, and two incorrect predictions may agree. Its contribution must therefore be evaluated together with supervised world/action learning through alignment and geometry ablations. In a multimodal demonstration distribution, independently sampled world and action trajectories may also represent different futures. Equation (18) pairs predictions through demonstrated targets and their noisy interpolants; it does not guarantee consistency between arbitrary independently sampled rollouts.

4.6.3. Joint objective

The complete training loss is

$$\mathcal{L} = \lambda_a \mathcal{L}_{\mathrm{FM}}^{\mathrm{act}} + \lambda_r \mathcal{L}_{\mathrm{FM}}^{\mathrm{rgb}} + \lambda_p \mathcal{L}_{\mathrm{FM}}^{\mathrm{xyz}} + \lambda_{\mathrm{align}} \mathcal{L}_{\mathrm{align}}, \tag{19}$$

where the nonnegative weights balance modality supervision and consistency. RGB and XYZ latent-velocity losses supervise demonstrated appearance and geometry, while the action objective supervises demonstrated commands. No additional auxiliary imitation or geometric reprojection loss is included. The 4DNeX-style post-optimization used for offline reconstruction of predicted future pointmaps is a separate fitting procedure (Appendix B); it neither updates the policy nor contributes a training loss.

Alignment couples a high-dimensional future predictor to a smaller action predictor, making its weight relevant to optimization. The component losses are normalized separately in the formulation above. A zero-alignment ablation retains pointmap input and future XYZ learning, whereas a zero-XYZ-loss ablation removes geometric prediction supervision while retaining the input. These interventions test different contributions.

4.7. Efficient inference

At deployment, the model receives current RGB and robot-frame pointmaps, constructs their fused clean context, and encodes that context through the world backbone once. Because C is independent of F and A in Eq. (13), its attention keys and values can be cached for the subsequent action steps. The action expert begins from $A^{(K)} \sim \mathcal{N}(0, \mathbf{I})$ and follows a decreasing flow-time schedule $1 = \tau_K > \cdots > \tau_0 = 0$. The first-order update is

$$A^{(k-1)} = A^{(k)} + (\tau_{k-1} - \tau_k) V_{\theta,\mathrm{act}}(A^{(k)}, \tau_k; C_t, \ell). \tag{20}$$

Following Fast WAM, we use $K = 10$ Euler updates and compute the current-context keys and values once at $\tau_C = 0$. The resulting normalized chunk is converted back to controller commands using $\mathcal{N}_A^{-1}$. The robot executes a prefix and replans with a new observation.

5. Simulation experiments

The simulation study separates nominal task performance, distribution shifts, and occlusion using benchmarks that share the LIBERO task interface. LIBERO evaluates nominal performance, LIBERO-Plus introduces controlled perturbations, and LIBERO-Occ occludes task-relevant objects in the third-person view. The same trained policy and inference settings are used for all three.

5.1. Benchmarks and protocol

LIBERO [19] comprises four suites, Spatial, Object, Goal, and Long, each containing ten language-conditioned tasks performed by a simulated Franka arm. We evaluate each task over 100 episodes from the benchmark initial states, yielding 1,000 episodes per suite. Success rates are reported by suite and across all four suites.

LIBERO-Plus [6] varies camera viewpoint, robot initial state, language instruction, lighting, background texture, sensor noise, and object layout. Its 10,030 task variants are evaluated without retraining policies learned from the original LIBERO demonstrations. Table 3 reports success by suite and perturbation, together with unperturbed LIBERO performance.

LIBERO-Occ [16] introduces occluded counterparts of the four suites. LIBERO objects are placed along the line of sight between the third-person camera and the task-relevant object. A placement is retained if it occludes the target, avoids scene collisions, and permits replay of the original demonstration. Each suite contains ten tasks, evaluated over 100 episodes per task. We report mean success over the resulting 1,000 episodes per suite.

For Noise, we use the original LIBERO-Plus image-corruption operators on the external RGB view. These operators do not directly corrupt the wrist RGB image or simulator depth, from which the pointmaps are constructed. The Noise results therefore evaluate robustness to external-view image corruption with the remaining inputs available.

5.2. Baselines

The simulation comparisons cover three groups:

- Vision–language–action models: OpenVLA [12], OpenVLA-OFT [10], RIPT-VLA [25], π_0 [1], π_0-FAST [20], $\pi_{0.5}$ [21], UniVLA [2], and NORA [8]. The LIBERO-Occ comparison additionally includes the Unified Vision-Language-Action Model [27], also named UniVLA but distinct from Bu et al. [2].

Table 1
Model and experiment configuration. Backbone, action-expert, and baseline optimization settings follow Fast-WAM [29].

Parameter	Setting
Video foundation model	Wan2.2-TI2V-5B
World transformer	30 layers; hidden width 3072; feed-forward width 14336
Action transformer	30 layers; hidden width 1024; feed-forward width 4096
Attention configuration	24 heads, head dimension 128 in both branches
Video latent / patch size	48 channels / $1 \times 2 \times 2$
Text conditioning	T5 features of width 4096; up to 128 tokens
Model size	6.3B parameters
Camera inputs	Main and wrist RGB-D; XYZ expressed in the robot base frame
Panel / packed frame	224×224 / 224×896 pixels
Visual training sequence	9 frames including the observation; 4 control steps apart
Action / proprioception	7 / 8 dimensions in the LIBERO interface
Action chunk / execution	32 predicted actions / first 10 executed before replanning
Action inference	10 Euler updates; current-context keys and values cached once
Control frequency	20 Hz
Optimizer / learning rate	AdamW / 10^{-4}
Weight decay / gradient clipping	0.01 / 1.0
Learning-rate schedule / precision	Cosine annealing / mixed precision
Evaluation step limit	400 for Spatial, Object, and Goal; 700 for LIBERO-10
LIBERO / LIBERO-Occ evaluation	100 episodes per task; 10 tasks per suite
Real-robot demonstrations	50 per task
Real-robot training	2 epochs per policy
Real-robot evaluation	25 trials per reported policy–task pair

- World action models: Cosmos Policy [11], Fast WAM [29], and WorldVLA [3]. Fast WAM is the direct architectural reference: GeoAlign-WAM retains its video backbone, action interface, and deployment procedure while adding pointmap input, future XYZ supervision, and action alignment.
- Geometry-aware policies: $\pi_{0.5}$ with Spatial Forcing [15] or ROCKET alignment [24].

For LIBERO-Occ, Table 4 distinguishes main-camera-only policies from the main-and-wrist-camera inputs of Fast WAM and GeoAlign-WAM.

5.3. Implementation details

GeoAlign-WAM builds on Fast WAM [29], coupling the Wan2.2-5B video diffusion transformer [26] to a smaller action expert through shared attention. The policy observes the third-person (agentview) and wrist cameras. For each camera, rendered depth is back-projected into robot-base pointmaps using Eq. (8) and normalized as in Section 4.2. The pointmaps are paired with registered RGB images. Four 224×224 panels form the 224×896 input described in Section 4.4. The visual training sequence contains the current frame and eight future frames separated by four control steps, covering the 32-step action chunk. Actions follow the seven-dimensional delta end-effector interface of Eq. (2) at 20 Hz. At inference, the expert performs $K = 10$ flow updates, executes the first ten actions, and replans. Episodes begin after 30 settling steps. Following the Fast WAM evaluation configuration, the limit is 400 control steps for Spatial, Object, and Goal, and 700 for LIBERO-10; evaluation uses a fixed random seed. Table 1 summarizes the configuration.

The baseline training recipe and optimization settings follow Fast WAM [29]; RGB–XYZ fusion follows 4DNeX [4]. The added CNN and alignment objective are described in Section 4.6.

5.4. Results

5.4.1. Nominal task performance

Table 2 summarizes LIBERO and LIBERO-Plus performance. GeoAlign-WAM achieves 97.4% mean success on nominal LIBERO, with rates of 98.1%, 98.3%, 96.9%, and 96.3% on Spatial, Object, Goal, and Long, respectively (Table 3). The mean is 0.2 percentage points below Fast WAM's 97.6%. The observed benefit therefore lies primarily in performance under visual shifts and occlusion.

Table 2
Aggregate success rates (%) on LIBERO and LIBERO-Plus [6]. Orig. and Plus denote performance on original LIBERO and overall LIBERO-Plus, respectively; Size denotes model size. Cam.: camera viewpoint; Robot: robot initial state; Lang.: language instruction; Light: lighting; BG: background texture; Noise: sensor noise; Layout: object layout. For GeoAlign-WAM, each perturbation score is the equal-weight mean over four suites; Plus is the equal-weight mean over all 28 suite–perturbation entries.

Method	Size	Orig.	Plus	Cam.	Robot	Lang.	Light	BG	Noise	Layout
VLAs										
$\pi_{0.5}$ [21]	3.3B	96.9	84.6	72.0	76.6	86.5	96.1	95.2	86.7	86.0
OpenVLA-OFT [10]	7B	97.1	69.6	56.4	31.9	79.5	88.7	93.3	75.8	74.3
RIPT-VLA [25]	7B	97.5	68.4	55.2	31.2	77.6	88.4	91.6	73.5	74.2
π_0 [1]	3.3B	91.3	69.3	61.0	40.8	63.7	89.3	84.1	80.1	75.9
π_0-FAST [20]	3.3B	85.5	61.6	65.1	21.6	61.0	73.2	73.3	74.4	68.8
UniVLA [2]	8.5B	95.2	42.9	1.8	46.2	69.5	69.0	81.0	21.2	31.9
NORA [8]	3B	87.9	39.0	2.2	37.0	65.1	45.7	58.6	12.8	62.1
OpenVLA [12]	7B	76.5	15.6	0.8	3.5	23.0	8.1	34.8	15.2	28.5
WAMs										
Cosmos-Policy [11]	2B	98.5	82.4	73.4	63.3	89.3	98.9	83.5	89.3	84.0
Fast-WAM [29]	6B	97.6	50.0	16.4	44.5	68.9	78.2	53.7	37.7	60.7
WorldVLA [3]	7B	79.1	25.0	0.1	27.9	41.6	43.7	17.1	11.0	38.0
Geometry-aware VLAs										
$\pi_{0.5}$ + Spatial Forcing [15]	3.3B	94.0	25.7	0.1	0.3	26.8	66.0	45.9	0.1	59.8
$\pi_{0.5}$ + ROCKET [24]	3.3B	95.3	47.5	30.9	75.6	29.3	69.2	47.0	25.4	62.0
GeoAlign-WAM (ours)	6.3B	97.4	81.0	79.4	72.2	77.7	81.4	95.2	88.6	72.5

5.4.2. Robustness to distribution shifts

Averaging equally over four suites and seven perturbation dimensions, GeoAlign-WAM achieves 81.0% success on LIBERO-Plus (Table 2). The difference from Fast WAM varies by perturbation (Table 3). Under camera shifts, the four suite rates are 73.4%, 94.5%, 95.4%, and 54.3%, exceeding Fast WAM by 59.0, 69.2, 87.3, and 36.6 percentage points, respectively. Gains also occur for robot-state changes in Goal (89.3% versus 24.7%) and for background and sensor-noise changes in Long (87.4% versus 22.8%, and 83.2% versus 28.5%). Nominal benchmark performance alone therefore does not characterize sensitivity to altered observations.

The gains do not extend to every perturbation. Under Object layout changes, GeoAlign-WAM reaches 72.8%, compared with Fast WAM's 73.0%; Goal lighting yields 63.1% versus 75.3%, and Long language changes yield 58.2% versus 60.1%. Several VLA baselines also perform better in language and layout settings. For example, $\pi_{0.5}$ achieves 89.6% under Long language changes and 98.2% under Spatial layout changes, compared with 58.2% and 81.8% for GeoAlign-WAM. The results support improved robustness to specific visual shifts, while leaving substantial variation across perturbation types.

5.4.3. Occlusion robustness

On LIBERO-Occ (Table 4), GeoAlign-WAM achieves 83.1% mean success, compared with 71.2% for Fast WAM, a gain of 11.9 percentage points. Improvements are 9.9, 11.9, 12.0, and 13.9 points on Spatial, Goal, Object, and LIBERO-10, respectively. Both methods use main and wrist cameras, holding view availability fixed. The first five baselines use only the main view, so those comparisons also reflect differences in available observations. Gains over Fast WAM occur across all four suites, including longer task sequences. The case studies illustrate the complementary observations available when one view is occluded.

Table 3
LIBERO-Plus success rates (%) by task suite and perturbation. Orig. denotes unperturbed LIBERO. Cam.: camera viewpoint; Robot: robot initial state; Lang.: language instruction; Light: lighting; BG: background texture; Noise: sensor noise; Layout: object layout. π_0^* denotes the results reported by LIBERO-Plus [6] for the official openpi LIBERO checkpoint.

Method	Orig.	Cam.	Robot	Lang.	Light	BG	Noise	Layout
(a) LIBERO-Spatial								
VLAs								
$\pi_{0.5}$ [21]	98.8	76.6	86.3	96.4	97.9	99.6	92.0	98.2
OpenVLA-OFT [10]	97.6	88.3	40.0	80.5	98.3	97.3	96.3	93.9
RIPT-VLA [25]	97.5	85.4	38.0	99.7	99.7	100.0	92.0	92.3
π_0 [1]	96.8	70.7	49.1	67.9	92.8	95.0	87.7	94.0
π_0^* [6]	96.8	17.8	6.6	58.8	89.7	90.7	90.9	89.1
π_0-FAST [20]	96.4	87.2	26.9	84.2	37.0	97.7	93.2	95.5
UniVLA [2]	96.5	1.1	52.6	83.9	96.6	90.7	15.7	69.5
NORA [8]	92.2	4.3	50.9	63.8	66.8	65.5	12.5	84.6
OpenVLA [12]	84.7	0.0	3.7	27.7	12.3	50.4	12.0	40.7
WAMs								
Cosmos Policy [11]	98.1	83.5	59.7	96.4	99.7	85.3	91.7	95.1
Fast-WAM [29]	98.2	14.4	44.0	69.5	87.3	69.8	35.0	60.5
WorldVLA [3]	85.6	0.0	44.3	46.3	65.1	19.8	11.7	46.1
Geometry-aware VLAs								
$\pi_{0.5}$ + ROCKET [24]	96.4	14.9	18.0	35.4	50.7	45.0	13.7	48.6
GeoAlign-WAM (ours)	98.1	73.4	78.3	95.5	98.0	99.4	84.3	81.8
(b) LIBERO-Object								
VLAs								
$\pi_{0.5}$	98.2	86.4	71.9	91.0	99.0	99.2	96.2	91.1
OpenVLA-OFT	98.4	38.9	25.4	99.0	73.7	97.6	72.3	71.8
RIPT-VLA	97.5	37.9	26.4	80.8	85.9	99.2	68.0	70.1
π_0	98.8	80.1	31.9	75.4	94.3	85.9	87.9	76.2
π_0^*	98.8	22.2	8.3	70.0	90.9	91.1	87.0	76.2
π_0-FAST	96.8	72.0	27.6	71.5	71.0	95.2	93.1	84.5
UniVLA	96.8	0.0	42.2	86.9	25.6	81.5	10.4	27.3
NORA	95.4	0.5	28.4	76.4	25.3	54.8	5.7	55.8
OpenVLA	88.4	0.5	4.5	21.0	1.0	45.2	11.4	22.4
WAMs								
Cosmos Policy	100.0	88.6	61.6	94.1	99.7	96.8	97.4	86.4
Fast-WAM	100.0	25.3	64.1	96.3	97.9	77.8	63.7	73.0
WorldVLA	89.0	0.0	26.4	57.2	20.5	17.3	18.0	53.6
Geometry-aware VLAs								
$\pi_{0.5}$ + ROCKET	98.8	41.4	27.1	44.6	91.6	69.8	34.1	83.6
GeoAlign-WAM (ours)	98.3	94.5	71.8	98.8	100.0	95.8	97.3	72.8
(c) LIBERO-Goal								
VLAs								
$\pi_{0.5}$	98.0	77.2	73.6	70.5	93.2	92.5	87.3	70.4
OpenVLA-OFT	97.9	62.0	25.2	53.2	93.9	92.5	75.2	59.1
RIPT-VLA	97.5	65.7	23.2	45.4	74.2	79.7	71.0	59.8
π_0	95.8	56.6	43.3	43.2	90.3	84.7	82.8	59.8
π_0^*	95.8	12.3	5.6	39.3	84.2	76.5	76.5	44.7
π_0-FAST	88.6	70.8	20.5	47.3	95.3	60.9	69.7	51.6
UniVLA	95.6	3.9	37.9	45.6	89.6	78.3	33.5	22.6
NORA	89.4	2.9	31.1	56.6	60.6	60.5	18.2	53.9
OpenVLA	79.2	2.5	2.7	21.5	9.0	27.1	19.5	25.6
WAMs								
Cosmos Policy	98.2	64.0	64.1	73.4	100.0	85.1	75.2	65.2
Fast-WAM	97.0	8.1	24.7	49.8	75.3	44.5	23.7	48.0
WorldVLA	82.6	0.3	30.6	42.2	68.8	30.3	13.5	47.4
Geometry-aware VLAs								
$\pi_{0.5}$ + ROCKET	96.6	41.2	36.9	27.6	57.0	47.0	28.2	46.8
GeoAlign-WAM (ours)	96.9	95.4	89.3	58.4	63.1	98.2	89.6	59.6
(d) LIBERO-Long								
VLAs								
$\pi_{0.5}$	92.4	49.2	76.1	89.6	93.8	90.3	73.1	85.9
OpenVLA-OFT	94.5	38.7	38.2	87.0	89.4	86.8	63.5	76.9
RIPT-VLA	97.5	34.1	38.4	88.3	93.4	89.3	66.4	79.2
π_0	73.8	38.7	39.9	69.7	79.2	72.3	64.6	77.6
π_0^*	85.2	3.8	3.6	68.4	74.5	69.5	64.4	69.6
π_0-FAST	60.2	33.2	12.0	43.6	91.6	44.6	46.1	47.8
UniVLA	92.0	1.9	53.2	64.2	65.7	74.4	25.4	16.4
NORA	74.6	1.2	39.4	64.0	30.3	54.0	15.1	59.5
OpenVLA	53.7	0.0	3.0	22.2	10.6	19.4	17.6	28.3
WAMs								
Cosmos Policy	97.6	58.9	67.4	94.8	96.4	68.9	91.8	92.9
Fast-WAM	95.2	17.7	45.3	60.1	52.2	22.8	28.5	61.2
WorldVLA	59.0	0.0	12.2	20.6	20.4	1.7	1.6	4.4
Geometry-aware VLAs								
$\pi_{0.5}$ + ROCKET	89.2	25.3	38.4	11.0	77.0	29.4	23.8	71.5
GeoAlign-WAM (ours)	96.3	54.3	49.3	58.2	64.3	87.4	83.2	75.6

5.5. Case studies

Figures 5–7 show three closed-loop rollouts of the LIBERO-Spatial instruction "pick up the black bowl in the top drawer of the wooden cabinet and place it on the plate": a nominal scene, a LIBERO-Plus camera shift, and a LIBERO-Occ occluder. Each figure contains five main-camera keyframes and their base-frame pointmaps, a wrist RGB/pointmap pair before grasping, seven joint-angle trajectories, gripper opening, replanning instants, and the 3D end-effector trajectory. These successful episodes illustrate behavior; aggregate performance is reported in Tables 2–4. Images are rendered offline from recorded simulator states.

5.5.1. Case S1: nominal scene

In the nominal case (Fig. 5), the end effector starts at (0.45, 0.02, 0.27) m in the base frame and descends into the open top drawer, where the bowl lies below the rim. The gripper closes at step 60 ($t = 3.0$ s), reducing the measured finger opening from 80 mm to 5 mm while pinching the bowl rim. The bowl is lifted to $z = 0.32$ m as the arm travels approximately 0.23 m along y_B toward the plate. Success is reached at $t = 7.2$ s with the end effector at (0.72, 0.25, 0.03) m. The weakly textured drawer interior appears in the pointmap as a recessed region, and the bowl's pointmap color changes with its position. Before grasping ($t = 2.25$ s), the wrist view shows the bowl between the fingers; the pointmap distinguishes the drawer, handle, and bowl from the table. Joint trajectories remain smooth across all 15 replanning instants, with consecutive action chunks continuing the motion.

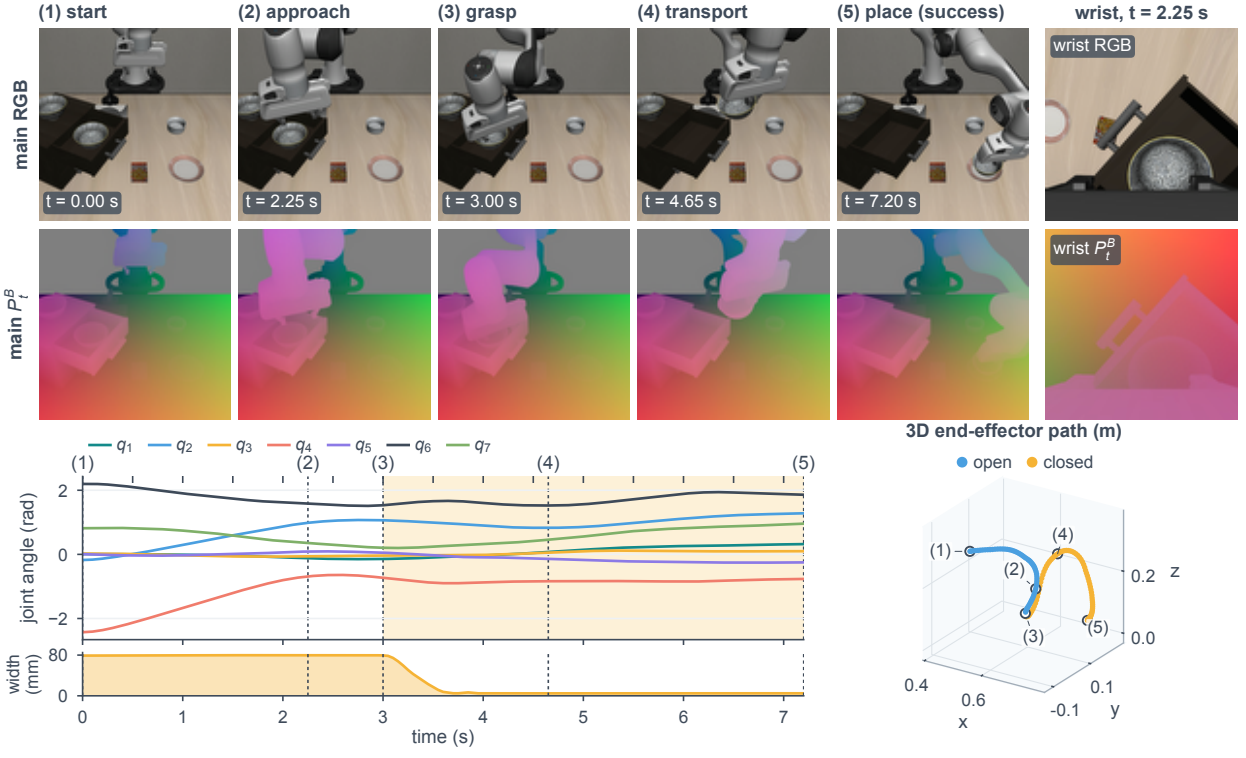


Figure 5: LIBERO-Spatial case (task 4, 144 control steps): main-camera RGB/XYZ keyframes, a wrist view, joint angles and gripper opening, and the 3D end-effector trajectory. XYZ colors encode robot-base coordinates.

5.5.2. Case S2: camera viewpoint shift

In the LIBERO-Plus case (Fig. 6), the main camera moves backward, reducing the apparent size of the robot and objects and revealing the table edges (the inset shows the nominal view). Static surfaces retain their pointmap colors because the coordinates remain in the robot base frame, despite changes in image position and size. The motion closely follows the nominal case: grasping occurs at step 50 ($t = 2.5$ s), transport reaches $z = 0.32$ m, and the episode ends at $t = 6.55$ s with the end effector at (0.73, 0.24, 0.04) m, approximately 2 cm from the nominal endpoint. The unaffected wrist view shows the bowl beneath the gripper in the open drawer before grasping.

Table 4
Success rate (%) on LIBERO-Occ [16]. Views lists the available camera inputs; Avg. is the mean over the four suites.

Method	Views	Spatial	Goal	Object	LIBERO-10	Avg.
UniVLA [27]	Main	61.00	67.40	78.80	21.20	57.10
OpenVLA [12]	Main	39.20	37.40	65.60	20.40	40.65
OpenVLA-OFT [10]	Main	48.00	44.80	75.60	23.40	47.95
π_0 [1]	Main	52.40	56.00	72.40	16.40	49.30
$\pi_{0.5}$ [21]	Main	39.00	40.20	64.00	19.00	40.55
Fast-WAM [29]	Main + wrist	74.4	73.3	67.3	69.8	71.2
GeoAlign-WAM (ours)	Main + wrist	84.3	85.2	79.3	83.7	83.1

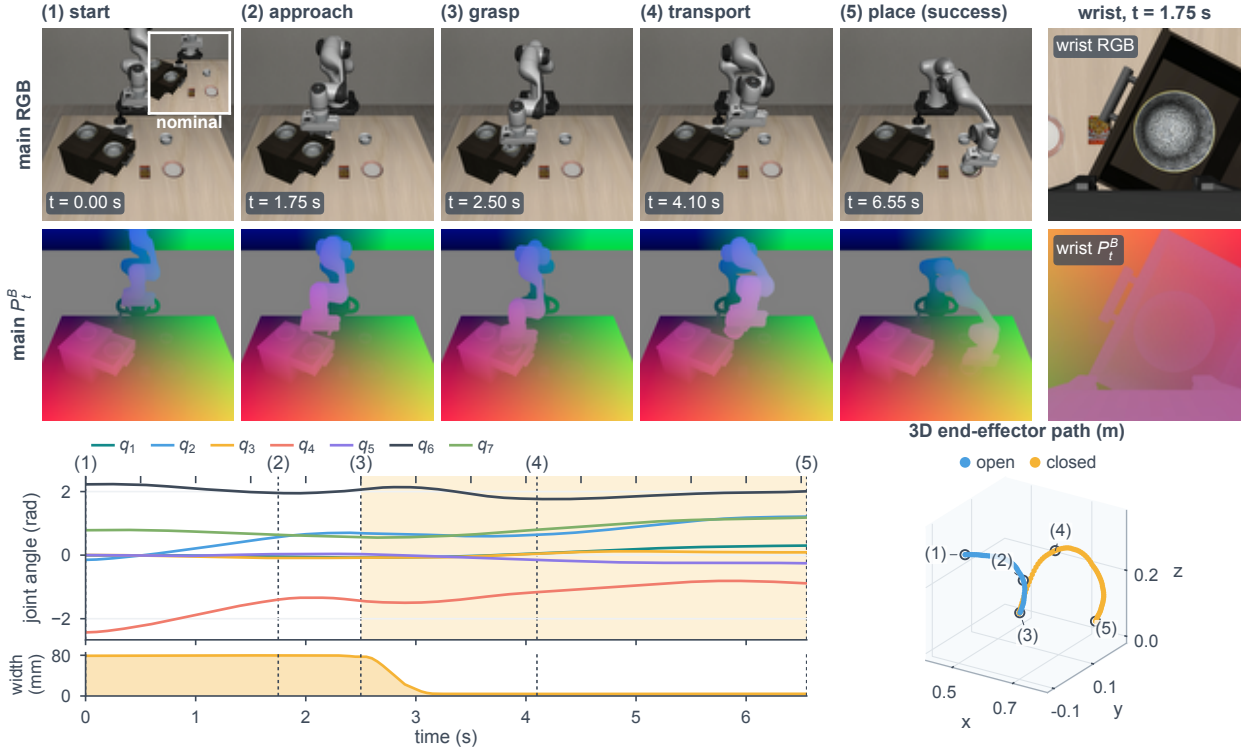


Figure 6: LIBERO-Plus viewpoint perturbation (131 control steps). The inset shows the nominal view; the remaining plots follow Fig. 5.

5.5.3. Case S3: occluded target

In the LIBERO-Occ case (Fig. 7), a wooden board lies between the main camera and the plate, leaving only part of the plate rim visible. The pointmap distinguishes the near-camera board from the plate behind it. During approach, the wrist camera observes the drawer and bowl; the board occupies only the upper-left image region and does not occlude the bowl. Grasping occurs at step 50 ($t = 2.5$ s). The bowl reaches a peak transport height of 0.35 m, approximately 2–3 cm above the other two cases, before being lowered behind the board onto the plate. Success is reached at $t = 6.5$ s with the end effector at (0.71, 0.24, 0.04) m. Despite the different images, endpoints across the three cases differ by at most 3 cm in the base frame.

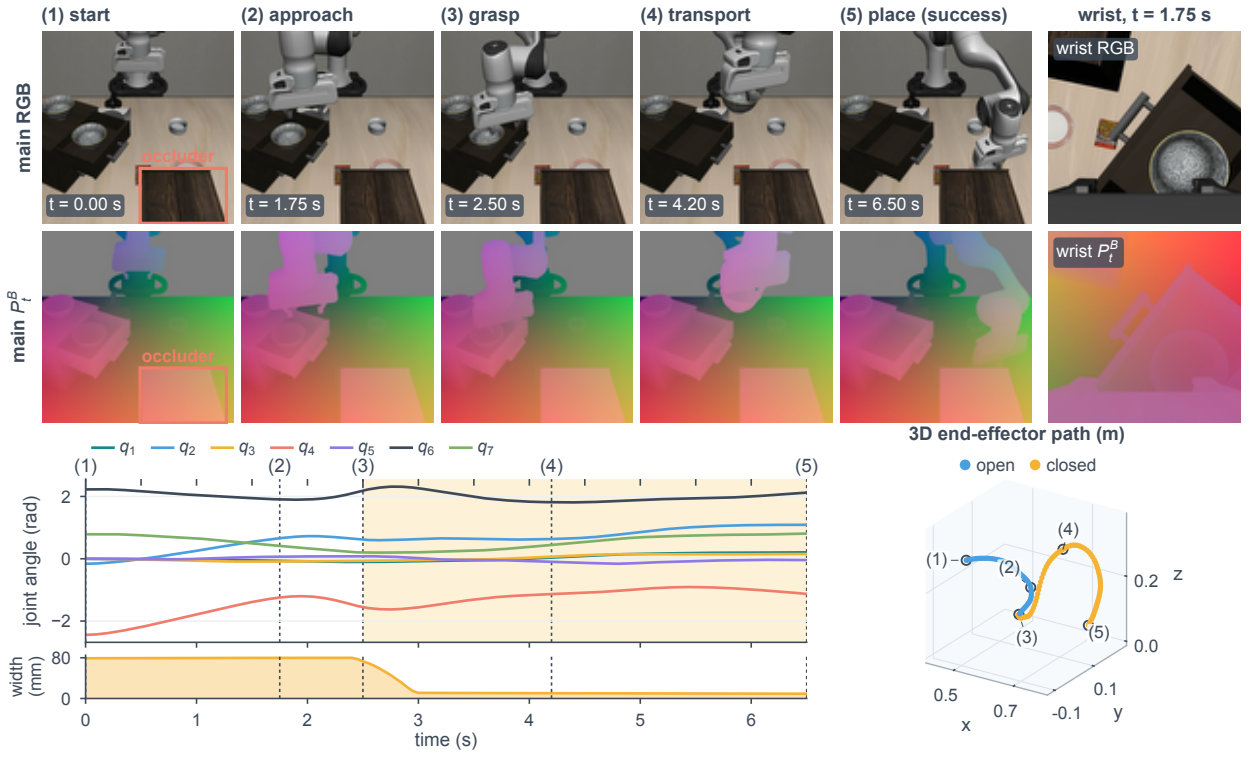


Figure 7: LIBERO-Occ case (130 control steps). The occluder is outlined in the first keyframe; the remaining plots follow Fig. 5.

6. Real-robot experiments

6.1. Setup

The real-robot platform comprises a 7-DoF Franka arm with a parallel-jaw gripper (Fig. 8) and two RGB-D cameras. The external main camera is positioned approximately 1.0 m in front of the robot base and 0.6 m above it, facing the robot; the wrist camera is mounted on the gripper. Both provide 640×480 color and depth images at 15 Hz. Intrinsics are obtained from the camera drivers. The main-camera pose is calibrated in the base frame, while the wrist-camera pose follows from hand–camera calibration and measured robot state. Depth images are converted to pointmaps using Eq. (8), with invalid-depth pixels masked. The policy receives the same 224×896 packed frame used in simulation.

A policy server returns 32-step chunks of seven-dimensional end-effector increments. The robot executes these in a 20 Hz control loop and requests a new chunk before the current one is exhausted. Each normalized action is scaled to a maximum translation of 12 mm and rotation of 0.036 rad per axis. The controller limits contact force and total episode displacement. An operator terminates each episode upon completion or failure, subject to a 120 s time limit.

6.2. Tasks, data and baselines

The three tasks assess placement precision, relative object positioning, and color-conditioned goal selection. In *Pick Cup*, the robot grasps a pink cup and places it upright on a yellow plate. In *Stack the Cup*, it grasps a yellow cup and nests it inside a green cup. In *Color Match*, it places a yellow cup on the plate of the same color while a white plate acts as a distractor. Task-specific policies are trained from 50 demonstrations per task. The baselines are:

- $\pi_{0.5}$ [21], adapted through full-parameter fine-tuning on the task demonstrations.
- Fast WAM [29], using RGB only, with the same task data, backbone, action interface, and inference settings as GeoAlign-WAM.

All real-robot policies are trained for two epochs.

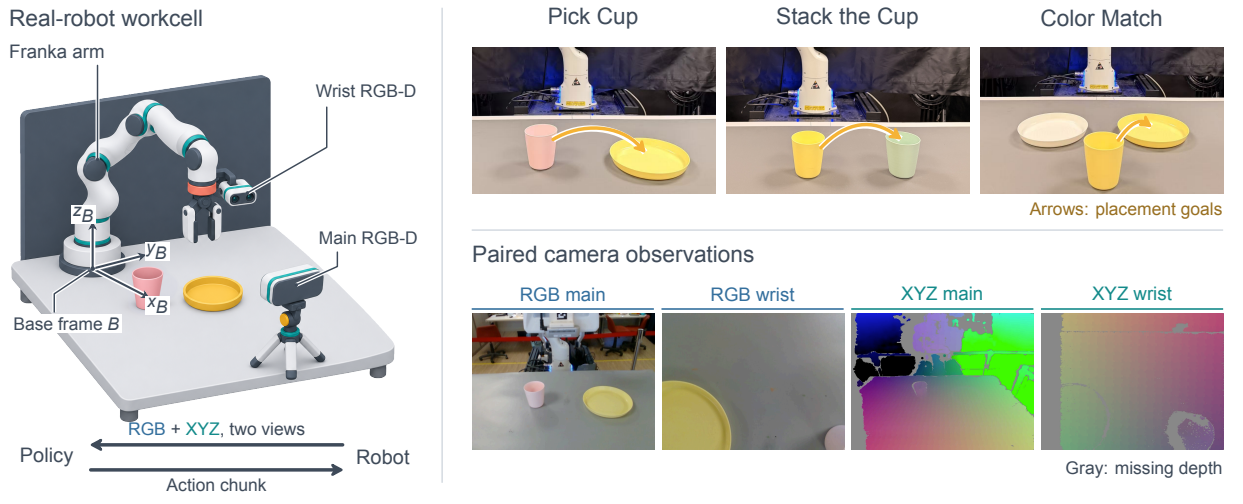


Figure 8: Real-robot workcell, the three manipulation tasks, and paired RGB/XYZ observations from the main and wrist cameras. Gray pointmap pixels indicate missing depth.

Table 5
Real-robot success rate (%), with 25 trials per reported task. Average is the mean across the three tasks.

Method	Pick Cup	Stack the Cup	Color Match	Average
$\pi_{0.5}$ [21]	64	48	76	62.7
Fast WAM, RGB only [29]	72	68	72	70.7
GeoAlign-WAM (ours)	**80**	**84**	**88**	**84.0**

6.3. Evaluation protocol

Each policy–task pair is evaluated over 25 trials. An operator labels task completion, and success rates are computed from these trial outcomes.

6.4. Results

Table 5 reports success rates of 80%, 84%, and 88% for GeoAlign-WAM on Pick Cup, Stack the Cup, and Color Match, respectively, corresponding to 20, 21, and 22 successes in 25 trials. Mean success is 84.0%, compared with 70.7% for RGB-only Fast WAM and 62.7% for $\pi_{0.5}$, giving gains of 13.3 and 21.3 percentage points. Improvements over Fast WAM are 8, 16, and 16 points across the three tasks.

These rates are descriptive estimates from 25 trials per task. The selected rollouts illustrate observed behavior but do not establish failure frequency or causes. The comparisons also do not isolate alignment from geometric input or controller settings.

6.5. Case studies

Figures 9–11 pair one successful GeoAlign-WAM trial per task with recorded failures of $\pi_{0.5}$ and RGB-only Fast WAM. Trials were selected for visibility of the relevant behavior in the main camera. Joint angles were recovered from recorded transforms between consecutive links; forward kinematics reproduces the measured end-effector position with a median error below 0.02 mm in all three trials. Times are measured from the start of each recording.

Across these trials, approach motion primarily involves elbow (q_4) and wrist (q_6) flexion, with shoulder motion (q_2) in Color Match. The arm remains stationary during gripper closure and opening, while base and upper-arm joints (q_1, q_3) contribute most of the transfer motion.

6.5.1. Case R1: Pick Cup

The policy begins acting at $t = 7.4$ s and lowers the gripper toward the pink cup, with changes of -0.39 rad in q_4 and $+0.33$ rad in q_6 during descent (Fig. 9). The fingers close on the cup wall between $t = 17.0$ and 18.9 s. Rotations of approximately 0.2–0.3 rad in each of q_1, q_3, and q_7 lift and transport the cup. It is released over the plate between

$t = 24.9$ and 26.1 s, lands upright, and remains there for the final 8 s. Before grasping, the wrist image shows the cup rim at its right edge. The pointmap, defined on the depth camera's image grid, resolves the table but contains no valid depth in the nearest region. In the $\pi_{0.5}$ failure, the cup is grasped and lifted but misses the plate and lands on the table behind it. RGB-only Fast WAM approaches the cup and then moves to the plate without it.

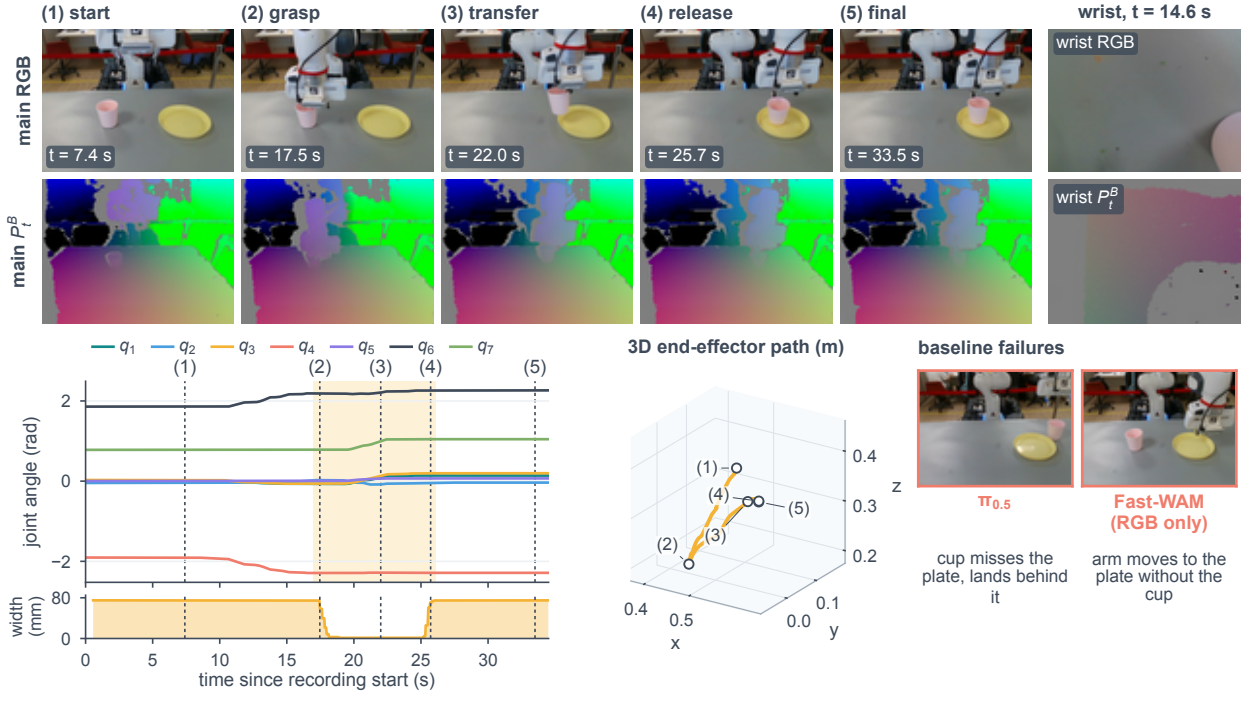


Figure 9: Pick Cup case (34.5 s recording): main and wrist RGB/XYZ views, reconstructed joint angles and measured gripper opening, the 3D end-effector trajectory, and same-task failure frames from $\pi_{0.5}$ and RGB-only Fast WAM.

6.5.2. Case R2: Stack the Cup

Stack the Cup requires lowering the yellow cup into the slightly larger opening of the green cup (Fig. 10). The gripper closes between $t = 13.6$ and 15.5 s. Transfer and descent take approximately 13 s, with net joint changes of 0.16 rad in q_1 and 0.14 rad in q_3. The cup is released inside the green cup between $t = 28.2$ and 29.4 s, and both remain upright for the final 9 s. In the $\pi_{0.5}$ failure, the yellow cup is brought over the green cup, which tilts during contact before both cups fall. RGB-only Fast WAM releases the yellow cup beside the green cup, leaving them side by side.

6.5.3. Case R3: Color Match

Color Match requires placing the yellow cup on the yellow plate while avoiding the white distractor (Fig. 11). The gripper closes between $t = 10.9$ and 13.0 s. Transfer produces the largest motion among the three tasks: q_4 and q_2 change by -0.52 and -0.48 rad. The cup is carried over the yellow plate at $t = 20.0$ s, is released at low height between $t = 20.6$ and 22.0 s, and settles upright at $t = 22.5$ s. In the $\pi_{0.5}$ failure, the cup tips after gripper contact, leaving the yellow plate empty. RGB-only Fast WAM lifts the cup but stops between the two plates, leaving it outside both.

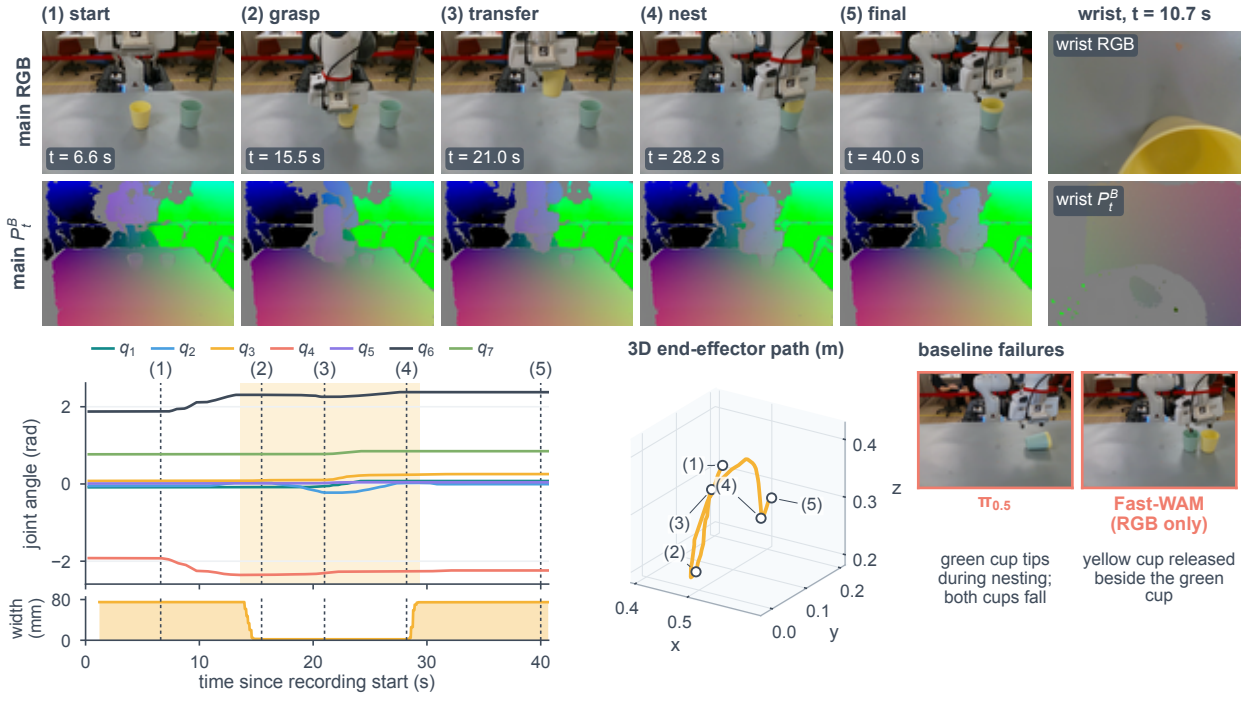


Figure 10: Stack the Cup case (40.7 s recording), with the same views and plots as Fig. 9.

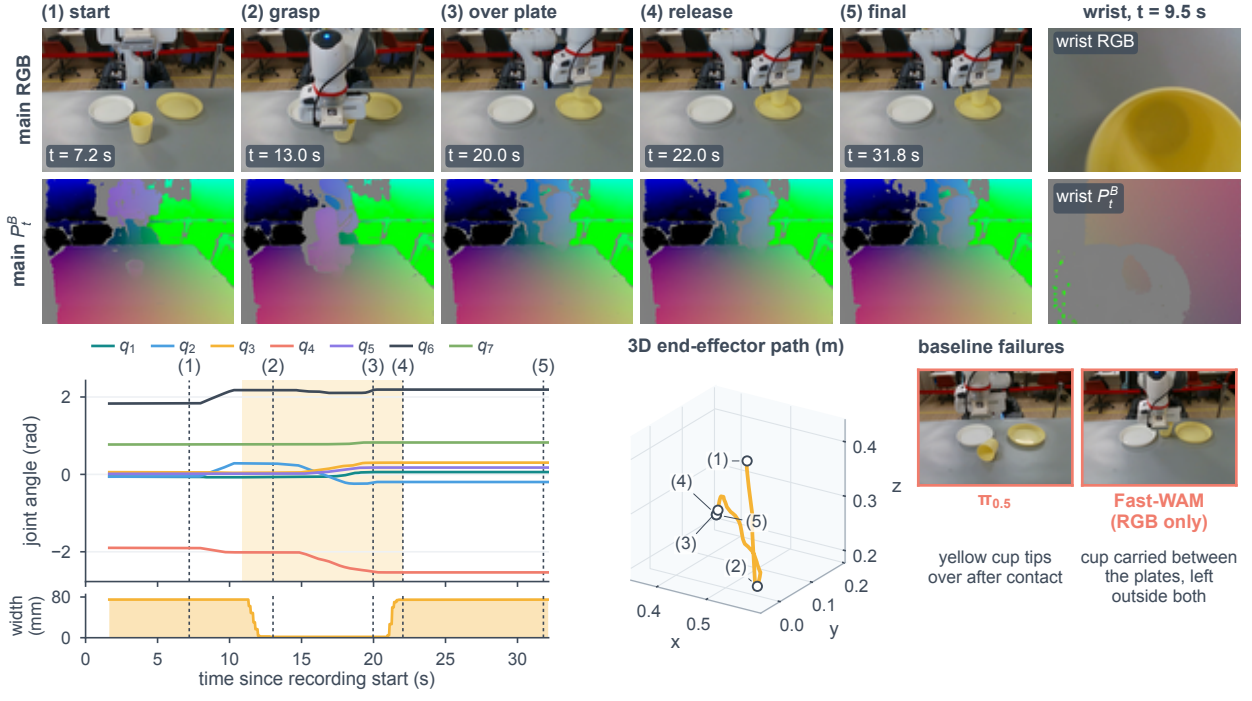


Figure 11: Color Match case (32.2 s recording), with the same views and plots as Fig. 9.

7. Conclusion

We presented GeoAlign-WAM, a world action model that combines a pretrained video prior with metric pointmaps in the robot base frame. RGB and XYZ observations are encoded into compatible latent grids and fused along their width. A world backbone predicts future appearance and geometry, while an action expert generates executable chunks from current context alone. A convolutional readout maps the predicted future latent to an auxiliary action chunk, and symmetric alignment couples it to the expert's prediction. Future geometric supervision thus contributes to learning while future synthesis is omitted at deployment. Evaluation covers nominal LIBERO, distribution shifts in LIBERO-Plus, occlusion in LIBERO-Occ, and three real-robot tasks. GeoAlign-WAM achieves 97.4% mean success on nominal LIBERO, 83.1% on LIBERO-Occ, and 84.0% across the real-robot tasks. The latter two improve over Fast WAM by 11.9 and 13.3 percentage points, respectively. LIBERO-Plus results show large gains under camera shifts, with remaining weaknesses under some language and lighting changes.

The method relies on calibrated metric depth, and action agreement alone does not guarantee geometric correctness. Future work will investigate learned metric geometry, uncertainty-weighted alignment, and transfer across tasks and embodiments. We will also explore active observation and memory grounded in viewpoint evidence before manipulation [9, 14], together with policy improvement from mixed-quality deployment experience [22].

A. RGB–pointmap fusion formulations

This appendix formalizes the five fusion layouts in Section 4.4, following 4DNeX [4]. GeoAlign-WAM uses width fusion. The other layouts and interaction operators define candidates for controlled comparisons; they are not reported as evaluated variants.

A.1. Paired latent grids and correspondence

Including the minibatch dimension, the normalized modality latents are

$$X = Z^{\mathrm{rgb}}, \quad Y = Z^{\mathrm{xyz}}, \qquad X, Y \in \mathbb{R}^{N\times T_z\times h\times w\times c}. \tag{21}$$

Here N is the number of paired samples, and w includes the fixed main/wrist camera layout. Entries $X_{n,k,i,j,:}$ and $Y_{n,k,i,j,:}$ share sample, time, and image-grid indices, but encode appearance and normalized metric coordinates, respectively. Indices are zero-based, and $[\,\cdot\,;\,\cdot\,]_a$ denotes concatenation along axis a.

A.2. Axis-wise fusion layouts

A.2.1. Channel fusion

At each location, concatenate the two feature vectors:

$$Z^{\mathrm{ch}} = [X;Y]_c \in \mathbb{R}^{N\times T_z\times h\times w\times 2c}, \qquad Z^{\mathrm{ch}}_{n,k,i,j,:} = [X_{n,k,i,j,:}; Y_{n,k,i,j,:}]. \tag{22}$$

A patch projection mixes the paired channels. For q grid locations per patch, the input dimension increases from qc to $2qc$, and $W^{\mathrm{ch}}_{\mathrm{in}} \in \mathbb{R}^{2qc\times d}$ projects it to backbone width d. The output projection predicts $2qc$ values, split into RGB and XYZ velocities. At fixed patch size, the token count is unchanged, but both projections require dimension adaptation.

A.2.2. Batch layout

Stack the modalities as separate batch entries:

$$Z^{\mathrm{ba}} = [X;Y]_N \in \mathbb{R}^{2N\times T_z\times h\times w\times c}, \qquad Z^{\mathrm{ba}}_{n,:,:,:,:} = X_n, \quad Z^{\mathrm{ba}}_{N+n,:,:,:,:} = Y_n. \tag{23}$$

Standard transformer attention operates independently on each batch entry. Thus, for a shared sample-wise network f_θ,

$$f_\theta(Z^{\mathrm{ba}}) = [f_\theta(X); f_\theta(Y)]_N. \tag{24}$$

Parameter sharing alone does not exchange information between paired RGB and XYZ inputs. A batch-layout comparison must therefore specify how paired outputs are combined and supplied to the action branch. Cross-modal attention or feature combination constitutes an additional operation.

A.2.3. Frame fusion

Pack complete modality sequences along the latent-time axis:

$$Z^{\mathrm{fr}} = [X; Y]_{T_z} \in \mathbb{R}^{N\times 2T_z\times h\times w\times c}, \qquad Z^{\mathrm{fr}}_{n,k,:,:,:} = X_{n,k}, \quad Z^{\mathrm{fr}}_{n,T_z+k,:,:,:} = Y_{n,k}. \tag{25}$$

Packed positions k and $T_z + k$ share a physical timestamp. Interleaving permutes this layout as $Z^{\mathrm{int}}_{n,2k} = X_{n,k}$ and $Z^{\mathrm{int}}_{n,2k+1} = Y_{n,k}$, with consistent modality and position labels. Temporal patches must remain within a modality unless mixed patches are explicitly intended.

A.2.4. Height fusion

Stack the two grids vertically:

$$Z^{\mathrm{ht}} = [X; Y]_h \in \mathbb{R}^{N\times T_z\times 2h\times w\times c}, \qquad Z^{\mathrm{ht}}_{n,k,i,j,:} = X_{n,k,i,j,:}, \quad Z^{\mathrm{ht}}_{n,k,h+i,j,:} = Y_{n,k,i,j,:}. \tag{26}$$

The packed row $h + i$ corresponds to the same physical image row as i. The stacking boundary is a modality boundary, not a spatial adjacency in the scene.

A.2.5. Width fusion

The adopted construction places XYZ beside RGB:

$$\begin{aligned} Z^{\mathrm{wd}} &= [X; Y]_w \in \mathbb{R}^{N\times T_z\times h\times 2w\times c}, \\ Z^{\mathrm{wd}}_{n,k,i,j,:} &= X_{n,k,i,j,:}, \qquad Z^{\mathrm{wd}}_{n,k,i,w+j,:} = Y_{n,k,i,j,:}. \end{aligned} \tag{27}$$

This is Eq. (12) with its batch axis restored. With two views per modality, the order is RGB main, RGB wrist, XYZ main, XYZ wrist. Width fusion preserves the latent channel dimension. Its inverse is a fixed split:

$$X = Z^{\mathrm{wd}}_{:,:,:,0:w,:}, \qquad Y = Z^{\mathrm{wd}}_{:,:,:,w:2w,:}. \tag{28}$$

Each original axis length is assumed divisible by its patch size, so that a patch does not cross an artificial modality boundary. The same packing and unpacking apply to clean context, noisy future latents, and predicted future velocities.

A.3. Tokenization, positions, and attention

Let a modality grid contain $L = (T_z/p_t)(h/p_h)(w/p_w)$ patches of size (p_t, p_h, p_w), with $q = p_t p_h p_w$. Denote patch extraction and flattening by $\mathcal{P}$. For layouts that retain separate modality patches, one explicit token construction is

$$H^m_r = \mathcal{P}(Z^m)_r W_{\mathrm{in}} + e_m + e_{\mathrm{pos}}(\gamma_r), \qquad m \in \{\mathrm{rgb}, \mathrm{xyz}\}, \tag{29}$$

where $W_{\mathrm{in}} \in \mathbb{R}^{qc\times d}$, e_m is a modality embedding, and γ_r records physical time, view and image-grid position. A rotary implementation applies the corresponding indices to queries and keys instead of adding e_{pos}. Equation (29) specifies a correspondence-preserving option for comparing layouts; the adopted width-fusion configuration follows 4DNeX. Packing alone neither supplies a modality embedding nor determines the rotary indices.

Channel fusion yields L joint tokens per sample. Under the divisibility assumption, frame, height, and width fusion yield $2L$ tokens, while batch layout yields two independent sets of L tokens. Unrestricted self-attention therefore forms an $L \times L$ matrix, a $2L \times 2L$ matrix, or two separate $L \times L$ blocks, respectively. These dimensions describe attention arithmetic rather than measured latency or total model cost.

For joint token processing, let $H = [H^{\mathrm{rgb}}; H^{\mathrm{xyz}}]_L$. Cross-modal exchange arises through the off-diagonal blocks of

$$\begin{aligned} Q &= HW_Q, \quad K = HW_K, \quad V = HW_V, \\ S &= \mathrm{softmax}\Big(QK^\top/\sqrt{d_h} + M\Big) = \begin{bmatrix} S_{rr} & S_{rp} \\ S_{pr} & S_{pp} \end{bmatrix}, \\ H'_{\mathrm{rgb}} &= S_{rr}V_{\mathrm{rgb}} + S_{rp}V_{\mathrm{xyz}}, \qquad H'_{\mathrm{xyz}} = S_{pr}V_{\mathrm{rgb}} + S_{pp}V_{\mathrm{xyz}}. \end{aligned} \tag{30}$$

Each output is a single-head feature matrix in $\mathbb{R}^{L\times d_h}$, where d_h is the head width; multiple heads are concatenated and projected. These modality blocks are distinct from the current/future/action groups in Eq. (13). RGB and XYZ can interact within C or F, while A cannot attend to F. Packing permutations must preserve the corresponding group mask and positional metadata.

A.4. Feature-combination alternatives

Aligned features $U = H^{\mathrm{rgb}}$ and $V = H^{\mathrm{xyz}}$ can also be combined without sequence concatenation. The following operators are alternatives to, rather than components of, the reported width-fusion model.

A.4.1. Projected concatenation and addition

For projection matrices of compatible size,

$$H_{\mathrm{cat}} = [U; V]_d W_c + b_c, \qquad H_{\mathrm{add}} = U W_r + V W_p + b, \tag{31}$$

where $W_c \in \mathbb{R}^{2d\times d}$ and $W_r, W_p \in \mathbb{R}^{d\times d}$. Both produce one token per aligned location. A single linear projection of concatenated features is algebraically equivalent to projected addition when $W_c = [W_r^\top \; W_p^\top]^\top$ and the biases match. They become different designs only when their parameter constraints or nonlinear processing differ.

A.4.2. Gated fusion

A data-dependent gate can interpolate the projected features:

$$G = \sigma([U; V]_d W_g + b_g), \qquad H_{\mathrm{gate}} = G \odot (U W_r) + (1 - G) \odot (V W_p). \tag{32}$$

Here $W_g \in \mathbb{R}^{2d\times d}$, $b_g \in \mathbb{R}^d$, and σ is the sigmoid. The gate supplies a learned weight for each token and output channel. Interpreting it as depth confidence would require calibration; invalid-depth masks and confidence inputs must be specified separately if used.

A.4.3. Bidirectional cross-attention

Separate modality streams can exchange information through

$$\begin{aligned} \overline{U} &= U + \mathrm{Attn}(U W_Q^r, V W_K^p, V W_V^p) W_O^r, \\ \overline{V} &= V + \mathrm{Attn}(V W_Q^p, U W_K^r, U W_V^r) W_O^p. \end{aligned} \tag{33}$$

Both updates use the incoming U, V. A subsequent operator such as Eq. (31) supplies the combined context to the policy. Such a design introduces additional projections and attention operations, which must be included in any parameter, memory and runtime comparison.

A.5. Future readout and comparison protocol

The CNN in Eq. (15) directly reads the predicted clean width-fused latent $\widehat{Z}^{\mathrm{wd}}$. The modality split in Eq. (28) is used to recover separate outputs, not as a preprocessing step for the CNN. Alternative transformer layouts can retain the same normalized action space and alignment target.

Controlled fusion comparisons must hold paired demonstrations, normalization, current/future visibility, action horizon, and evaluation protocol fixed. Changes to projections, interaction layers, positional encoding, trainable parameters, and computational cost must be reported. Modality losses are evaluated after reversing the corresponding layout. No empirical ranking is assigned to these alternatives; their evaluation remains pending in Section 5.4.

B. Offline reconstruction of predicted future pointmaps

B.1. Scope and coordinate convention

We adopt 4DNeX post-optimization [4] to reconstruct predicted future pointmaps offline after world generation and decoding. This procedure is separate from latent flow matching and geometry–action alignment and is excluded from the deployed action policy.

For a camera view, let $\widehat{\widehat{P}}_k$ be the decoded normalized XYZ map at future time $t + \delta_k$. Inverting latent normalization before decoding and then restoring the workspace transform gives

$$\widehat{P}_k^B = \widehat{\widehat{P}}_k \odot \boldsymbol{s} + \boldsymbol{b}. \tag{34}$$

Fitting retains the robot base frame and metric scale. Independent centering of predicted frames would discard the offsets needed to recover their positions.

B.2. Camera and depth fitting

Let $\mathbf{K}$ denote intrinsics for one camera stream, $\mathbf{R}^B_{C,k} \in \mathrm{SO}(3)$ its camera-to-base rotation, $\boldsymbol{c}^B_k \in \mathbb{R}^3$ its camera origin in the base frame, and $d_k(u) > 0$ a depth variable at pixel u. The point represented by these variables is

$$\widetilde{P}^B_k(u) = \mathbf{R}^B_{C,k}\left[d_k(u)\mathbf{K}^{-1}\widetilde{u}\right] + \boldsymbol{c}^B_k, \qquad \widetilde{u} = (u_x, u_y, 1)^\top. \tag{35}$$

Using camera-to-base transforms makes this expression consistent with Eq. (8); it is equivalent to the inverse-extrinsic convention in 4DNeX. Its camera/depth fitting objective is

$$\underset{\mathbf{K},\{\mathbf{R}^B_{C,k},\boldsymbol{c}^B_k,d_k\}_{k=1}^T}{\text{minimize}} \quad \frac{1}{\sum_k |\Omega_k|}\sum_{k=1}^{T}\sum_{u\in\Omega_k}\left\|\widetilde{P}^B_k(u) - \widehat{P}^B_k(u)\right\|_2^2, \tag{36}$$

where Ω_k is the set of retained pixels. Including all pixels gives the unweighted point-fitting objective of the released 4DNeX implementation, up to constant normalization. The residual measures three-dimensional distance. Predicted pointmaps remain fixed while camera and depth variables are optimized; RGB provides appearance for the reconstructed points but does not enter the objective.

We use the released 4DNeX fitting configuration: intrinsics are shared within each camera stream, while poses and depths vary by frame. Intrinsics, six-dimensional rotation parameters, translations, and pixel depths are optimized with Adam for 1,500 iterations at a learning rate of 5×10^{-3}. The objective includes all predicted pixels, without an additional validity mask, regularizer, or early-stopping rule. Initialization follows the released solver, and depths are clamped to a minimum of 10^{-4} during fitting. Different cameras have separate intrinsic matrices. The fitted variables reconstruct each sequence through Eq. (35). Camera-consistent fitting does not enforce temporal point identity or guarantee physically correct interactions.

References

[1] Black, K., Brown, N., Driess, D., Esmail, A., Equi, M.R., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Shi, L.X., Smith, L., Tanner, J., Vuong, Q., Walling, A., Wang, H., Zhilinsky, U., 2025. π_0: A Vision-Language-Action Flow Model for General Robot Control, in: Proceedings of Robotics: Science and Systems, LosAngeles, CA, USA.

[2] Bu, Q., Yang, Y., Cai, J., Gao, S., Ren, G., Yao, M., Luo, P., Li, H., 2025. UniVLA: Learning to act anywhere with task-centric latent actions. arXiv preprint arXiv:2505.06111 Accepted to Robotics: Science and Systems (RSS) 2025.

[3] Cen, J., Yu, C., Yuan, H., Jiang, Y., Huang, S., Guo, J., Li, X., Song, Y., Luo, H., Wang, F., Zhao, D., Chen, H., 2025. WorldVLA: Towards autoregressive action world model. arXiv preprint arXiv:2506.21539 .

[4] Chen, Z., Liu, T., Zhuo, L., Ren, J., Tao, Z., Zhu, H., Hong, F., Pan, L., Liu, Z., 2025. 4DNeX: Feed-forward 4D generative modeling made easy. arXiv preprint arXiv:2508.13154 .

[5] Chi, C., Feng, S., Du, Y., Xu, Z., Cousineau, E., Burchfiel, B.C., Song, S., 2023. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, in: Proceedings of Robotics: Science and Systems, Daegu, Republic of Korea.

[6] Fei, S., Wang, S., Shi, J., Dai, Z., Cai, J., Qian, P., Ji, L., He, X., Zhang, S., Fei, Z., Fu, J., Gong, J., Qiu, X., 2025. LIBERO-Plus: In-depth Robustness Analysis of Vision-Language-Action Models. arXiv preprint arXiv:2510.13626 .

[7] Guo, J., Li, Q., Li, P., Chen, Z., Sun, N., Su, Y., Wang, H., Zhang, Y., Li, X., Liu, H., 2026. Unified 4D World Action Modeling from Video Priors with Asynchronous Denoising. arXiv preprint arXiv:2604.26694 .

[8] Hung, C.Y., Sun, Q., Hong, P., Zadeh, A., Li, C., Tan, U.X., Majumder, N., Poria, S., 2025. NORA: A small open-sourced generalist vision language action model for embodied tasks. arXiv preprint arXiv:2504.19854 .

[9] Ji, Y., Li, Z., Cai, Y., Wu, S., Han, B.S., Lv, C., 2026. SOR-Nav: Search or Relocate? Context-Gated Exploration and Cross-Region Relocation for Object Navigation. arXiv preprint arXiv:2609.34707 .

[10] Kim, M.J., Finn, C., Liang, P., 2025a. Fine-tuning vision-language-action models: Optimizing speed and success. arXiv preprint arXiv:2502.19645 Accepted to Robotics: Science and Systems (RSS) 2025.

[11] Kim, M.J., Gao, Y., Lin, T.Y., Lin, Y.C., Ge, Y., Lam, G., Liang, P., Song, S., Liu, M.Y., Finn, C., Gu, J., 2026. Cosmos Policy: Fine-tuning video models for visuomotor control and planning. arXiv preprint arXiv:2601.16163 Accepted to the International Conference on Learning Representations (ICLR) 2026.

[12] Kim, M.J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E.P., Sanketi, P.R., Vuong, Q., Kollar, T., Burchfiel, B., Tedrake, R., Sadigh, D., Levine, S., Liang, P., Finn, C., 2025b. Openvla: An open-source vision-language-action model, in: Agrawal, P., Kroemer, O., Burgard, W. (Eds.), Proceedings of The 8th Conference on Robot Learning, PMLR. pp. 2679–2713.

[13] Levine, S., Finn, C., Darrell, T., Abbeel, P., 2016. End-to-end training of deep visuomotor policies. Journal of Machine Learning Research 17, 1–40.

[14] Lewis, S.H., Guo, Z., Chen, B., Li, Z., Lin, H., Huang, H., 2026. SafeVantage: Vantage-Aware Memory for Reliable Embodied Decisions. arXiv preprint arXiv:2609.36906 .

[15] Li, F., Song, W., Zhao, H., Wang, J., Ding, P., Wang, D., Zeng, L., Li, H., 2025. Spatial Forcing: Implicit Spatial Representation Alignment for Vision-language-action Model. arXiv preprint arXiv:2510.12276 .

[16] Li, T., Zhang, J., Wang, S., Huang, X., Wei, Z., 2026a. LIBERO-Occ: Evaluating and Improving Vision-Language-Action Models under Scene-Induced Occlusion via Viewpoint Imagination. arXiv preprint arXiv:2606.10862 .

[17] Li, Y., Wei, X., Cao, J., Wang, H., Chi, X., Bai, C., Sun, Q., Li, J., Zhang, X., Jia, P., Tang, J., Han, S., Zhang, S., 2026b. WAM4D: Fast 4D World Action Model via Spatial Register Tokens. arXiv preprint arXiv:2606.14048 .

[18] Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M., 2022. Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 .

[19] Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., Stone, P., 2023. Libero: Benchmarking knowledge transfer for lifelong robot learning, in: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 44776–44791.

[20] Pertsch, K., Stachowicz, K., Ichter, B., Driess, D., Nair, S., Vuong, Q., Mees, O., Finn, C., Levine, S., 2025. FAST: Efficient action tokenization for vision-language-action models. arXiv preprint arXiv:2501.09747 Accepted to Robotics: Science and Systems (RSS) 2025.

[21] Physical Intelligence, Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al., 2025. $\pi_{0.5}$: a vision-language-action model with open-world generalization. arXiv preprint arXiv:2504.16054 Accepted to the Conference on Robot Learning (CoRL) 2025.

[22] Ren, Y., Yan, Y., Li, Z., Guo, J., Zeng, D., Tao, J., Yu, L., Tian, X., Lv, C., 2026. Learning from Mixed-Quality Deployment Experience for Robot Manipulation. arXiv preprint arXiv:2609.29000 .

[23] Shridhar, M., Manuelli, L., Fox, D., 2023. Perceiver-actor: A multi-task transformer for robotic manipulation, in: Liu, K., Kulic, D., Ichnowski, J. (Eds.), Proceedings of The 6th Conference on Robot Learning, PMLR. pp. 785–799.

[24] Sun, G., Du, T., Feng, K., Luo, C., Ding, X., Shen, Z., Wang, Z., He, Y., Li, A., 2026. ROCKET: Residual-oriented multi-layer alignment for spatially-aware vision-language-action models. arXiv preprint arXiv:2602.17951 Accepted to Advances in Neural Information Processing Systems (NeurIPS) 2026.

[25] Tan, S., Dou, K., Zhao, Y., Krähenbühl, P., 2025. Interactive post-training for vision-language-action models. arXiv preprint arXiv:2505.17016 .

[26] Team Wan, Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., et al., 2025. Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 .

[27] Wang, Y., Li, X., Wang, W., Zhang, J., Li, Y., Chen, Y., Wang, X., Zhang, Z., 2025. Unified vision-language-action model. arXiv preprint arXiv:2506.19850 .

[28] Ye, S., Ge, Y., Zheng, K., Gao, S., Yu, S., Kurian, G., Indupuru, S., Tan, Y.L., Zhu, C., Xiang, J., Malik, A., Lee, K., Liang, W., Ranawaka, N., Gu, J., Xu, Y., Wang, G., Hu, F., Narayan, A., Bjorck, J., Wang, J., Kim, G., Niu, D., Zheng, R., Xie, Y., Wu, J., Wang, Q., Julian, R., Xu, D., Du, Y., Chebotar, Y., Reed, S., Kautz, J., Zhu, Y., Fan, L.J., Jang, J., 2026. World Action Models are Zero-shot Policies. arXiv preprint arXiv:2602.15922 .

[29] Yuan, T., Dong, Z., Liu, Y., Zhao, H., 2026. Fast-WAM: Do world action models need test-time future imagination? arXiv preprint arXiv:2603.16666 .

[30] Ze, Y., Zhang, G., Zhang, K., Hu, C., Wang, M., Xu, H., 2024. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations, in: Proceedings of Robotics: Science and Systems, Delft, Netherlands.

[31] Zhao, T.Z., Kumar, V., Levine, S., Finn, C., 2023. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware, in: Proceedings of Robotics: Science and Systems, Daegu, Republic of Korea.

[32] Zhu, C., Yu, R., Feng, S., Burchfiel, B., Shah, P., Gupta, A., 2025. Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets, in: Proceedings of Robotics: Science and Systems, LosAngeles, CA, USA.

[33] Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., Vuong, Q., Vanhoucke, V., Tran, H., Soricut, R., Singh, A., Singh, J., Sermanet, P., Sanketi, P.R., Salazar, G., Ryoo, M.S., Reymann, K., Rao, K., Pertsch, K., Mordatch, I., Michalewski, H., Lu, Y., Levine, S., Lee, L., Lee, T.W.E., Leal, I., Kuang, Y., Kalashnikov, D., Julian, R., Joshi, N.J., Irpan, A., Ichter, B., Hsu, J., Herzog, A., Hausman, K., Gopalakrishnan, K., Fu, C., Florence, P., Finn, C., Dubey, K.A., Driess, D., Ding, T., Choromanski, K.M., Chen, X., Chebotar, Y., Carbajal, J., Brown, N., Brohan, A., Arenas, M.G., Han, K., 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control, in: Tan, J., Toussaint, M., Darvish, K. (Eds.), Proceedings of The 7th Conference on Robot Learning, PMLR. pp. 2165–2183.