SCULPT-VLA: LEARNING STRUCTURED CONTROL THROUGH STAGED ACTION GROUNDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) policies increasingly incorporate structured intermediate supervision beyond action labels. Yet specifying what an intermediate representation should encode leaves open how action prediction learns to depend on it. We introduce **SCULPT-VLA**, a policy that learns structured control through staged action grounding. Its action-conditioning state comprises complementary factors for task progression, scene dynamics, and spatial grounding. Training first forms these factors with teacher scaffolds, then grounds coarse action prediction through their composition as scaffold inputs are withdrawn. Direct perceptual access is subsequently restored for continuous refinement, combining the learned state with perceptual detail. The curriculum separates learning to condition actions on structure from refining continuous control. Deployment requires neither teachers nor discrete-action autoregression. SCULPT-VLA achieves higher average success than shared-backbone baselines on LIBERO, SimplerEnv-WidowX, and RoboTwin 2.0 Full. On SimplerEnv-WidowX, final success is 83.5%, versus 71.3% when Stage-II action learning directly accesses vision and language. Across four physical robot tasks, average success under the tested distribution shifts reaches 58.1%, compared with 45.6% for $\pi_{0.5}$. Training ablations and factor-wise interventions support the staged design and show that the learned state continues to contribute to control after direct perceptual access is restored.

1 INTRODUCTION

As a robot stacks a cup, the relevant control problem changes from transporting the object to aligning its rim and releasing the gripper. The objects and instruction may stay the same, yet the next action depends on a changing combination of task progress, relative geometry, and interaction-induced motion. Vision-language-action (VLA) models bring broad semantic knowledge to these decisions (Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024). The opportunity is to turn this knowledge into a control state that tracks the changing demands of manipulation.

Recent policies enrich action learning with intermediate supervision beyond demonstrated actions. Task decompositions and affordance plans organize execution and interaction (Physical Intelligence et al., 2025; Zawalski et al., 2024; Nasiriany et al., 2024), while spatial and predictive representations supply geometric and transition information (Qu et al., 2025; Bu et al., 2025; Qiu et al., 2026). World-action models and structured feature objectives further connect these signals to action generation (Bai et al., 2026b; Zheng et al., 2026). These advances raise a complementary question about policy learning (Fig. 1): *how should action learning establish a functional role for this structure?*

The conditioning pathways that support policy learning may differ from those needed at deployment. When the action decoder has unrestricted access to multimodal context from the outset, joint optimization must learn both the representations and how to condition actions on them. Conditioning actions exclusively on an intermediate state instead requires that state to preserve every detail needed for continuous control. This motivates a staged approach: first ground action learning through a structured state, then restore direct perceptual access for refinement. The central question is whether the resulting state continues to contribute when the final policy can also use perceptual detail.

SCULPT-VLA realizes this principle through *staged action grounding*, which shapes the dependence of action prediction on the structured control state during training. A shared backbone organizes this state into task-progression, scene-dynamics, and spatial-grounding factors, distinct from

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the action readout. Teacher scaffolds first establish factor roles. Coarse action learning then draws its non-action context from their composition as scaffold inputs are withdrawn. Continuous refinement then restores direct perceptual access, conditioning on factor states, action readout, and visual detail. The deployed policy forms its control state from observations and instructions, without teacher encoders or discrete-action autoregression.

Evaluations on LIBERO, SimplerEnv-WidowX, and RoboTwin 2.0 Full show higher average success than shared-backbone baselines. Four physical robot tasks further demonstrate gains under the tested distribution shifts. Training controls examine how this capability is acquired, and final-state interventions test whether the factors still contribute after direct access is restored. We contribute (1) a **structured control state** that organizes task progression, scene dynamics, and spatial grounding separately from action readout; (2) **staged action grounding**, a training method linking factor formation to continuous control through staged changes in information access; and (3) **simulation, physical-robot, and functional-intervention evidence** supporting this design and the learned state's contribution to deployed control.

2 Related Work

2.1 Vision-Language-Action Models

VLA models transfer pretrained vision-language representations to robot control through discrete action prediction (Zitkovich et al., 2023; Kim et al., 2024; Pertsch et al., 2025) or continuous action generation (Black et al., 2024). Their action interfaces increasingly structure multimodal representations before decoding: bridge representations and lightweight adapters connect pretrained backbones to action heads (Li et al., 2025; Yang et al., 2025; Wang et al., 2026c), while Action QFormer constructs an instruction-conditioned, query-based representation for action decoding (Ji et al., 2026).

2.2 World Models and Predictive Robot Learning

Predictive policies learn from anticipated observations and scene changes (Zhang et al., 2025a; Zhu et al., 2026), while latent-action models extract compact transition representations (Bu et al., 2025; Cai et al., 2026). Recent world-action models explicitly address how predictive features support action generation. AGRA aligns video-model features with spatially coherent semantic representations to improve the world-action interface (Qiu et al., 2026). Bridge-WA distills future, change, and motion-flow priors into a policy, conditions action generation on these priors, and removes the teacher at inference (Bai et al., 2026b). These methods establish predictive representations as a source of control-relevant information beyond current observations.

2.3 Structured Intermediate Representations for Robot Control

Affordance plans, geometric representations, and structured feature supervision organize control-relevant information (Nasiriany et al., 2024; Qu et al., 2025; Zheng et al., 2026). Reasoning policies connect task semantics to actions through explicit subtasks or reasoning traces (Zawalski et al., 2024; Physical Intelligence et al., 2025; Zhao et al., 2025; Wang et al., 2026a) and latent reasoning states (Bai et al., 2026a; Liu et al., 2026; Huang et al., 2026). Latent Semantic Scaffolding aligns action-token representations with phase-specific reasoning targets during training and discards the alignment head at inference (Li et al., 2026). Counterfactual VLA analyses also examine whether encoded instruction information affects behavior (Grant et al., 2026; Fang et al., 2026). SCULPT-VLA examines how structured representations acquire a functional role during action learning. Factor formation is followed by action grounding through the composed state, then refinement with direct perceptual access restored.

3 Method

SCULPT-VLA learns structured control through staged action grounding (Fig. 1). Structured control denotes action prediction conditioned on complementary control factors explicitly trained for use in

action learning. The method separates factor formation from action readout and schedules their interaction during training.

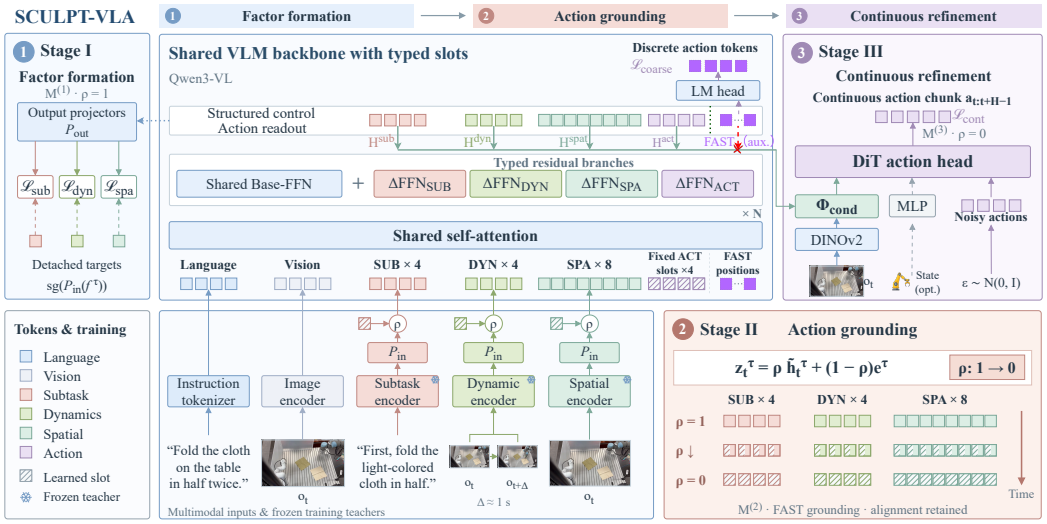


Figure 1: SCULPT-VLA overview. Stage I forms teacher-aligned factors; Stage II withdraws scaffold inputs while grounding FAST prediction through interacting factors; Stage III restores direct perceptual access for continuous refinement. The continuous head jointly uses factor states, fixed action readout, and DINOv2 features.

3.1 Structured Control State

Inputs and control factors. At time t, the policy receives an observation o_t and instruction x, denoted by $\mathbf{q}_t = (o_t, x)$, and predicts an action chunk $\mathbf{a}_{t:t+H-1} \in \mathbb{R}^{H \times d_a}$, where H is the action horizon and d_a the action dimension. We structure the action-conditioning state around three control-oriented factors: **task progression** (sub), **scene dynamics** (dyn), and **spatial grounding** (spa). They address recurring requirements of manipulation: tracking the active subtask, accounting for interaction-induced changes, and localizing relevant geometry.

Structured factors and action readout. We implement the factors as typed latent slots, fixed token positions whose contextualized states form the structured control state. We denote these states by $\mathbf{h}_t^{\text{sub}}, \mathbf{h}_t^{\text{dyn}}, \mathbf{h}_t^{\text{spa}}$. Additional fixed action-side slots $\mathbf{h}_t^{\text{act}}$ serve as *action readout tokens*, aggregating context for action decoding. These differ from the FAST positions $\mathbf{Z}_t^{\text{fast}}$ used to predict a discrete action sequence (Pertsch et al., 2025). For discrete prediction, the action group uses causal attention at every layer, with fixed readout preceding FAST positions and unable to access the teacher-forced FAST suffix. We denote the complete input sequence by $\mathbf{Z}_t^{(0)}$. Group definitions, slot counts, and token layouts are given in Appendix A.

Teacher-guided factor formation. Complementary frozen teachers provide training-time scaffolds: Qwen3-Embedding supplies instruction-aware task semantics, UniVLA latent actions provide inverse-dynamics cues, and Depth Anything V2 supplies current-scene geometry (Zhang et al., 2025b; Bu et al., 2025; Yang et al., 2024). For $\tau \in \{\text{sub}, \text{dyn}, \text{spa}\}$, the feature $\mathbf{f}_t^\tau$ is mapped to a scaffold input $\tilde{\mathbf{h}}_t^\tau = P_{\text{in}}^\tau(\text{sg}(\mathbf{f}_t^\tau))$. The student representation is aligned to the detached reference $\bar{\mathbf{h}}_t^\tau = \text{sg}(\tilde{\mathbf{h}}_t^\tau)$:

$$\mathcal{L}_\tau = \frac{1}{|\mathbf{h}_t^\tau|} \sum_k \left\| P_{\text{out}}^\tau(\mathbf{h}_{t,k}^\tau) - \bar{\mathbf{h}}_{t,k}^\tau \right\|_2^2 \tag{1}$$

Here sg stops gradients, and both projectors are trainable while teacher encoders remain frozen. The scaffolds establish factor-specific targets; the following curriculum trains those factors to support action without teacher inputs.

Factor-specific computation. Self-attention and the pretrained FFN remain shared. Small type-specific residual FFNs preserve a dedicated computational path for each factor and the action readout. For token i at layer l,

$$\widetilde{\mathbf{z}}_l^{(i)} = \mathbf{z}_l^{(i)} + \mathrm{FFN}_{\mathrm{base}}^{(l)}(\mathbf{z}_l^{(i)}) + \mathbb{I}[\tau(i) \neq \emptyset]\, \Delta\mathrm{FFN}_{\tau(i)}^{(l)}(\mathbf{z}_l^{(i)}) \tag{2}$$

Vision and language tokens have $\tau(i) = \emptyset$ and retain the base path. Token types select their residual FFNs deterministically.

3.2 Staged Action Grounding

Action grounding trains the structured state to support action prediction. The three stages regulate factor interaction and action conditioning through the attention patterns in Fig. 2.

Stage I: Forming structured control states. Under $\mathbf{M}^{(1)}$, task and dynamics factors attend to vision, language, and their own group; the spatial factor attends to vision and its own group, without language access. Cross-factor attention and the action group are inactive, allowing factor-specific alignment to teacher targets.

Stage II: Grounding actions through structured states. Under $\mathbf{M}^{(2)}$, the factors interact bidirectionally, while action readout and FAST positions can read these factors but cannot directly read vision or language. The composed factor state therefore provides the non-action context for FAST prediction.

During Stage II, teacher-derived inputs are replaced gradually by learned query embeddings $\mathbf{e}_k^\tau$:

$$\mathbf{z}_{t,k}^{\tau,(0)} = \rho(r)\tilde{\mathbf{h}}_{t,k}^{\tau} + (1 - \rho(r))\mathbf{e}_k^{\tau}, \tag{3}$$

$$\rho(r) = \mathrm{clip}\left(1 - \frac{r - r_{\mathrm{anneal}}^{\mathrm{start}}}{r_{\mathrm{anneal}}^{\mathrm{end}} - r_{\mathrm{anneal}}^{\mathrm{start}}}, 0, 1\right) \tag{4}$$

Here r is the training step. Alignment remains active as ρ decreases to zero. By the end of annealing, the backbone infers factor states from observations and instructions.

Stage III: Refining continuous control with restored perception. Under $\mathbf{M}^{(3)}$, action-side positions regain direct perceptual and language access for continuous refinement; factor-group visibility remains unchanged. The continuous-control pass contains factor and fixed readout positions, with no teacher-forced FAST suffix. Weak FAST supervision uses a separate autoregressive auxiliary path.

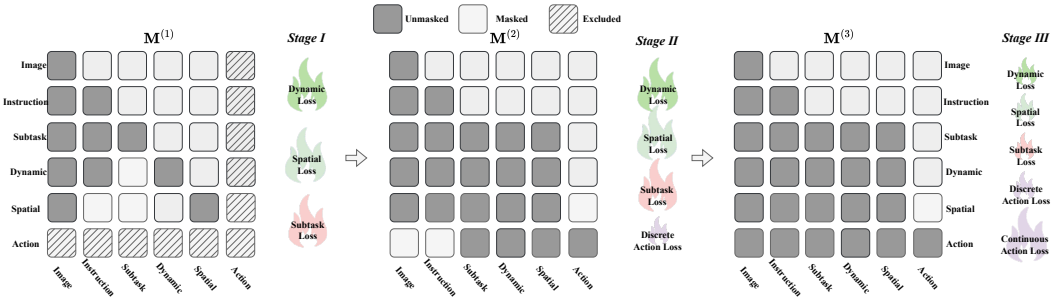


Figure 2: Stage-specific attention masks and objectives. Rows are query groups; columns are key groups. Dark gray permits attention, light gray blocks it, and hatching marks excluded groups. The discrete Action group uses causal attention over fixed readout followed by FAST positions. The continuous Action group contains only fixed readout.

3.3 Training Objectives and Deployment

Structured state formation. The Stage I objective combines the factor-alignment losses:

$$\mathcal{L}^{(1)} = \sum_{\tau \in \{\mathrm{sub,dyn,spa}\}} \lambda_\tau^{(1)} \mathcal{L}_\tau \tag{5}$$

Coarse action grounding. FAST provides coarse action supervision in Stage II and a weak auxiliary objective through a separate discrete pass in Stage III. We tokenize the demonstrated action chunk into a sequence $\mathbf{c}_t$ and train its autoregressive prediction:

$$\mathcal{L}^{(s)}_{\text{coarse}} = \mathcal{L}_{\text{FAST}}(\mathbf{c}_t \mid \mathbf{Z}^{(0)}_t, \mathbf{M}^{(s)}), \qquad s \in \{2, 3\} \tag{6}$$

Stage II combines this loss with continued factor alignment:

$$\mathcal{L}^{(2)} = \sum_{\tau} \lambda^{(2)}_{\tau}(r)\mathcal{L}_{\tau} + \lambda^{(2)}_{\text{coarse}}(r)\mathcal{L}^{(2)}_{\text{coarse}} \tag{7}$$

Continuous refinement. A flow-matching head (Black et al., 2024) conditions on the final factor representations, action readout states, and a DINOv2 visual-detail branch (Oquab et al., 2023):

$$\mathbf{u}_t = \Phi_{\text{cond}}(\mathbf{H}^{\text{slot},(L)}_t, \mathbf{H}^{\text{act},(L)}_t, \mathbf{E}^{\text{DINOv2}}_t) \tag{8}$$

Here $\mathbf{H}^{\text{slot},(L)}_t$ denotes final-layer structured factor states, $\mathbf{H}^{\text{act},(L)}_t$ the fixed action-readout states, and $\mathbf{E}^{\text{DINOv2}}_t$ the visual-detail features. For Gaussian noise $\boldsymbol{\epsilon}$ and $\alpha \sim \mathcal{U}(0, 1)$, let $\mathbf{a}_\alpha = (1-\alpha)\boldsymbol{\epsilon} + \alpha\mathbf{a}_{t:t+H-1}$. The action expert v_θ learns the corresponding velocity field:

$$\mathcal{L}_{\text{cont}} = \mathbb{E}_{\mathbf{a},\boldsymbol{\epsilon},\alpha}\left[\left\|v_\theta(\mathbf{a}_\alpha, \alpha \mid \mathbf{u}_t) - (\mathbf{a}_{t:t+H-1} - \boldsymbol{\epsilon})\right\|^2_2\right] \tag{9}$$

Stage III combines flow matching with weak alignment and the separate FAST auxiliary loss:

$$\mathcal{L}^{(3)} = \lambda^{(3)}_{\text{cont}}\mathcal{L}_{\text{cont}} + \sum_{\tau} \eta_\tau \mathcal{L}_\tau + \eta_{\text{coarse}}\mathcal{L}^{(3)}_{\text{coarse}}, \qquad \eta_\cdot \ll 1 \tag{10}$$

Deployment. At inference, learned query embeddings initialize the factor and action readout positions, and the backbone forms the control state without teacher encoders or alignment projectors. The continuous head decodes action chunks from these states and DINOv2 visual-detail features, without FAST autoregression. Implementation and training settings are in Appendix A.

4 Experiments

We evaluated whether SCULPT-VLA converts structured supervision into closed-loop visuomotor control across simulation and physical manipulation. Training controls tested the learning pathway, and factor-wise interventions examined the structured state's role after direct perceptual access was restored.

4.1 Evaluation Setup

We evaluated on the four LIBERO suites (Liu et al., 2023), SimplerEnv-WidowX (Li et al., 2024), RoboTwin 2.0 Full (Chen et al., 2025b), and four real-world tabletop tasks. For LIBERO and SimplerEnv, training uses LIBERO-SUBTASK and BRIDGE-SUBTASK (Appendix A.3). Our main results on these benchmarks use three seeds and 50 simulation trials per task and seed. Four Qwen3-VL baselines share the backbone, action data, action horizon, and StarVLA evaluation pipeline (Community, 2026), with FAST, OFT, π_0-style flow, or GR00T-style action interfaces. Published results from other training stacks provide additional context and are listed separately. We report closed-loop success and mean open-loop score (mOLS), the average of four thresholded action-error scores, with higher values indicating better agreement with demonstration actions (Appendix B.2). Table 2 reports RoboTwin 2.0 Full results: joint training over 50 tasks with 50 clean and 500 randomized demonstrations per task, followed by Clean and Random evaluation. Evaluation protocols appear in Appendices B.1 and B.

4.2 Main Results

Simulation. SCULPT-VLA achieves 98.5% average LIBERO success and 96.9% on the Long suite, compared with 96.6% and 93.8% for Qwen3-VL-OFT, the strongest shared-backbone baseline on LIBERO (Table 1). On SimplerEnv-WidowX, it reaches 83.5%, exceeding Qwen3-VL-GR00T at

65.3% and the direct continuous Qwen3-VL-PI policy at 60.9%. The gain over Qwen3-VL-GR00T is largest on Stack, where success rises from 18.8% to 75.3%. On RoboTwin 2.0 Full, SCULPT-VLA reaches 92.2% average success, exceeding Qwen3-VL-PI, the strongest shared-backbone StarVLA action-head variant in Table 2. Since LIBERO scores are close to saturation, the following ablations focus on SimplerEnv.

Table 1: Main results on LIBERO and SimplerEnv-WidowX (success, %). Published methods retain their reported protocols. Shared-backbone variants use Qwen3-VL in StarVLA (Community, 2026). SCULPT-VLA entries are three-seed means (Appendix B.3). Dashes indicate no reported result for the listed setting. Shades rank the five highest distinct scores per column, from green to yellow. For $\pi_{0.5}$, LIBERO results follow the official openpi release; SimplerEnv results follow Lian et al. (2026).

Method	**LIBERO**					**SimplerEnv-WidowX**				
	Spatial	Object	Goal	Long	Avg.	Spoon	Carrot	Stack	Eggplant	Avg.
Published results										
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5	0.0	0.0	0.0	4.1	1.0
OpenVLA-OFT (Kim et al., 2024; 2025)	97.6	98.4	97.9	94.5	97.1	12.5	4.2	8.3	37.5	15.6
Octo-Small (Team et al., 2024)	78.9	85.7	84.6	51.1	75.1	47.2	9.7	4.2	56.9	29.5
π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2	29.1	0.0	16.7	62.5	27.1
LoLA (Wang et al., 2025a)	**99.6**	99.6	97.2	88.2	96.2	95.8	58.3	54.2	79.2	71.9
LaMP (Wang et al., 2026b)	99.4	**99.8**	97.4	96.7	98.3	79.1	66.7	75.0	95.8	79.2
UniVLA (Wang et al., 2025b)	95.4	98.8	93.6	94.0	95.5	83.3	66.7	33.3	95.8	69.8
FlowVLA (Zhong et al., 2025)	93.2	95.0	91.6	72.6	88.1	70.8	62.5	62.5	**100.0**	74.0
VLA-JEPA (Sun et al., 2026b)	96.2	99.6	97.2	95.8	97.2	75.0	70.8	12.5	70.8	57.3
DreamVLA (Zhang et al., 2025a)	97.6	98.8	97.2	95.0	97.2	79.2	41.7	20.8	**100.0**	60.4
CoWVLA (Yang et al., 2026)	97.2	97.8	94.6	92.8	95.6	79.2	66.7	62.5	95.8	76.0
ThinkAct (Huang et al., 2025)	88.3	91.4	87.1	70.9	84.4	58.3	37.5	8.7	70.8	43.8
F1 (Lv et al., 2025)	98.2	97.8	95.4	91.3	95.7	50.0	70.8	50.0	66.7	59.4
UD-VLA (Chen et al., 2025a)	94.1	95.7	91.2	89.6	92.7	58.3	62.5	54.1	75.0	62.5
Fast-ThinkAct (Huang et al., 2026)	92.0	97.2	90.2	79.4	89.7	—	—	—	—	—
LaRA-VLA (Bai et al., 2026a)	96.4	**99.8**	**98.6**	96.6	97.9	95.8	62.5	25.0	91.7	68.8
X-VLA (Zheng et al., 2025)	98.2	98.6	97.8	**97.6**	98.1	**100.0**	**91.7**	**95.8**	95.8	**95.8**
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	98.8	98.2	98.0	92.4	96.9	49.3	64.7	44.7	69.7	57.1
Shared-backbone comparisons										
Qwen3-VL-FAST	97.3	97.4	96.3	90.6	95.4	18.8	31.3	4.2	71.9	31.6
Qwen3-VL-OFT	97.8	98.6	96.2	93.8	96.6	90.3	38.5	29.7	**100.0**	64.6
Qwen3-VL-PI	98.8	99.6	95.8	88.4	95.7	78.1	46.9	30.2	88.5	60.9
Qwen3-VL-GR00T	97.8	98.8	97.4	92.0	96.5	83.0	59.4	18.8	**100.0**	65.3
SCULPT-VLA	99.1	99.5	98.3	96.9	**98.5**	90.7	70.7	75.3	97.3	83.5

Physical evaluation under distribution shift. The four physical tasks require object collection, stacking, insertion, and deformable-object manipulation (Figure 3). Each method is trained on 100 ID demonstrations per task. We evaluate one trained checkpoint per method and task using three independent evaluation batches of 30 physical rollouts per ID/OOD condition, for 90 rollouts per condition. Mean and sample standard deviation are computed across these three batches. All compared methods follow the same task-specific ID/OOD evaluation protocol. OOD settings change appearance, object properties, or placement relative to these demonstrations (Appendix B.5). SCULPT-VLA achieves higher OOD success on all four tasks, averaging 58.1% versus 45.6% for $\pi_{0.5}$ (Table 3). Cloth Folding remains the hardest setting at 37.8% OOD success. Here, SCULPT-VLA's mOLS is 0.227 versus 0.229 for $\pi_{0.5}$. Closed-loop success and open-loop action agreement thus rank the two policies differently on this task.

Table 2: Comparison on LIBERO and RoboTwin 2.0 Full (success, %). Published results retain their source protocols (Appendix B.1). Shared-backbone variants use Qwen3-VL in StarVLA (Community, 2026). RoboTwin Avg. averages Clean and Random. †: LIBERO reproduction by Sun et al. (2026a); ‡: LIBERO episode limits of 600 steps (Spatial/Object/Goal) and 700 (Long). Motus's LIBERO and π_0's RoboTwin results follow Yuan et al. (2026). Darker and lighter green mark the best and second-best distinct scores per column, respectively.

Method	**LIBERO**					**RoboTwin 2.0 Full**		
	Spatial	Object	Goal	Long	Avg.	Clean	Random	Avg.
Published results								
π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2	65.9	58.4	62.2
LingBot-VLA (Wu et al., 2026)	81.8†	95.0†	86.6†	67.8†	82.8†	88.6	86.7	87.7
Motus (Bi et al., 2025)	96.8	99.8	96.6	97.6	97.7	88.7	87.0	87.9
Fast-WAM (Yuan et al., 2026)	98.2	**100.0**	97.0	95.2	97.6	91.9	91.8	91.9
OpenWAM-α (Wang et al., 2026d)	**99.6**	99.6	**99.8**	**98.2**	**99.3**‡	**93.7**	**93.5**	**93.6**
Shared-backbone comparisons								
Qwen3-VL-FAST	97.3	97.4	96.3	90.6	95.4	72.5	83.2	77.9
Qwen3-VL-OFT	97.8	98.6	96.2	93.8	96.6	88.2	88.3	88.3
Qwen3-VL-PI	98.8	99.6	95.8	88.4	95.7	88.1	88.8	88.5
Qwen3-VL-GR00T	97.8	98.8	97.4	92.0	96.5	88.0	88.5	88.3
SCULPT-VLA	99.1	99.5	98.3	96.9	98.5	92.3	92.1	92.2

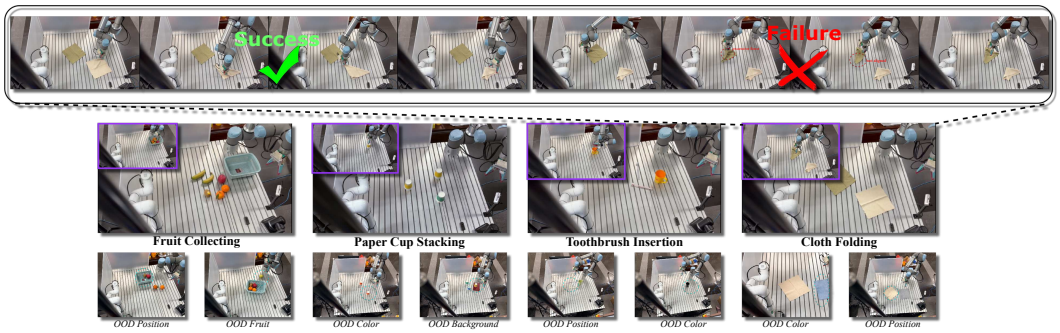


Figure 3: Real-world evaluation on Fruit Collecting, Paper Cup Stacking, Toothbrush Insertion, and Cloth Folding. Top: representative Cloth Folding success and failure. Middle: four task setups. Bottom: task-specific OOD configurations. Task definitions and perturbations are detailed in Table 12.

Table 3: Real-world task success (%). Each method uses 100 ID demonstrations and one trained checkpoint per task. Entries are mean ± sample SD across three independent evaluation batches of 30 physical rollouts per condition. OOD settings are defined in Table 12; all open-loop scores are in Table 13.

	Fruit Collecting		Cup Stacking		Toothbrush Insertion		Cloth Folding	
Method	ID	OOD	ID	OOD	ID	OOD	ID	OOD
OpenVLA-OFT	50.0±3.3	24.4±5.1	61.1±1.9	36.7±6.7	44.4±1.9	17.8±5.1	25.6±1.9	2.2±1.9
π_0	65.6±1.9	45.6±6.9	75.6±1.9	56.7±3.3	67.8±3.8	44.4±5.1	35.6±1.9	14.4±3.8
$\pi_{0.5}$	74.4±1.9	52.2±5.1	81.1±1.9	60.0±6.7	73.3±3.3	47.8±3.8	41.1±1.9	22.2±5.1
SCULPT-VLA	80.0±3.3	60.0±3.3	85.6±1.9	70.0±6.7	84.4±5.1	64.4±5.1	56.7±3.3	37.8±5.1

4.3 Structured Organization and Action Grounding

Table 4 tests the representation and curriculum on SimplerEnv-WidowX, including individual factor removals. Variant definitions and per-threshold open-loop scores are in Appendix B.4.

Table 4: Complete ablations on SimplerEnv-WidowX. FAST and Continuous denote the Stage-II discrete and Stage-III continuous policies. Success (%) is reported as mean ± sample SD over evaluation runs; variants are trained independently. mOLS is defined in Appendix B.2; per-threshold scores and variant definitions are in Appendix B.4.

Variant	FAST	Continuous	mOLS ↑
Full SCULPT-VLA	74.7±1.0	83.5±0.0	0.369
shared factor FFN	65.7±1.0	76.8±1.0	0.329
generic latent pool	52.8±1.2	63.9±1.1	0.271
w/o subtask factor	67.3±1.3	78.3±1.3	0.342
w/o spatial factor	64.8±0.8	72.2±0.8	0.312
w/o dynamics factor	61.6±1.0	73.3±0.8	0.322
w/o action-side latent slots	69.2±1.3	77.0±1.0	0.335
w/o Stage-II visibility restriction	55.7±0.8	71.3±1.0	0.302
w/o Stage I	60.0±1.0	70.2±1.0	0.301
Stage III only	—	61.0±1.0	0.262
w/o FAST supervision	—	68.2±1.3	0.290
w/o DINOv2 detail branch	73.7±1.0	81.0±1.0	0.354

Structured organization. A generic latent pool achieves 63.9% success, compared with 83.5% for the full model. This control preserves the teacher targets, token budget, action objectives, and three-stage schedule, while jointly removing explicit type identities, factor-specific FFNs, and type-dependent visibility constraints. It tests their combined contribution. Sharing the factor FFN gives 76.8%, testing factor-specific computation. Removing subtask, spatial, or dynamics factors tests the respective factor families; each removal reduces both FAST and continuous success. The largest factor-removal drop shifts from dynamics under FAST decoding to spatial grounding under continuous control. Removing action-side latent slots leaves factor states available for decoding, but degrades both policies. This supports using a dedicated readout to combine factor information for action prediction.

Action grounding. Allowing direct vision–language access to action positions in Stage II reduces success to 71.3%. The deficit is already present under FAST decoding and persists in Stage III, when both variants have the same direct perceptual access. This supports establishing factor-conditioned action prediction before continuous refinement. The benefit is task-dependent: unrestricted Stage-II access gives 98.7% on LIBERO-Spatial, close to the standard 99.1% (Appendix C.3.1). Removing FAST supervision tests the coarse action objective and yields 68.2% success.

Training stages and refinement. Skipping Stage I yields 70.2%, and the Stage III only variant gives 61.0%. The latter retains the SCULPT-VLA architecture and auxiliary losses, distinguishing it from the direct Qwen3-VL-PI baseline. Omitting training stages also changes the total optimization budget under these schedules. Within Stage III, removing the DINOv2 detail branch gives 81.0%, a smaller reduction than removing structured organization.

Across the reported variants, mOLS and mean continuous success rank the models in the same order. The gains therefore appear in both action prediction on demonstration observations and closed-loop task execution. The following analysis complements these training-time ablations with interventions on a fixed trained policy, testing whether the resulting factors remain functionally involved during continuous control.

4.4 Functional Analysis of Learned Structured Control States

Factor-wise functional intervention. We replaced selected final factor states with their training-set means before continuous decoding, retaining the backbone, action readout states, and DINOv2 features. On Fruit Collecting, replacing all three factors reduced success from 80.0% to 15.6% ID and from 60.0% to 5.6% OOD (Table 5). Subtask replacement had the largest effect among individual groups. These behavioral changes show that the learned structured state contributes to deployed control after direct perceptual access is restored. The intervention protocol is described in Appendix C.4.4.

Table 5: Factor-wise intervention on Fruit Collecting. Selected final factor states are replaced with their training-set means before continuous decoding. Success (%): mean ± sample SD across three evaluation batches of 30 rollouts per condition. Raw counts: Table 16.

Replaced states	ID	OOD
Full SCULPT-VLA	80.0±3.3	60.0±3.3
Replace subtask factor	54.4±5.1	28.9±10.2
Replace spatial factor	73.3±3.3	50.0±6.7
Replace dynamics factor	65.6±5.1	42.2±6.9
Replace all factors	15.6±5.1	5.6±5.1

Decodable factor content. Figure 4 visualizes depth reconstructed from spatial factors and depth change predicted from dynamics factors. The separately trained probes reveal scene geometry and state-transition cues in the learned state, complementing the behavioral interventions.

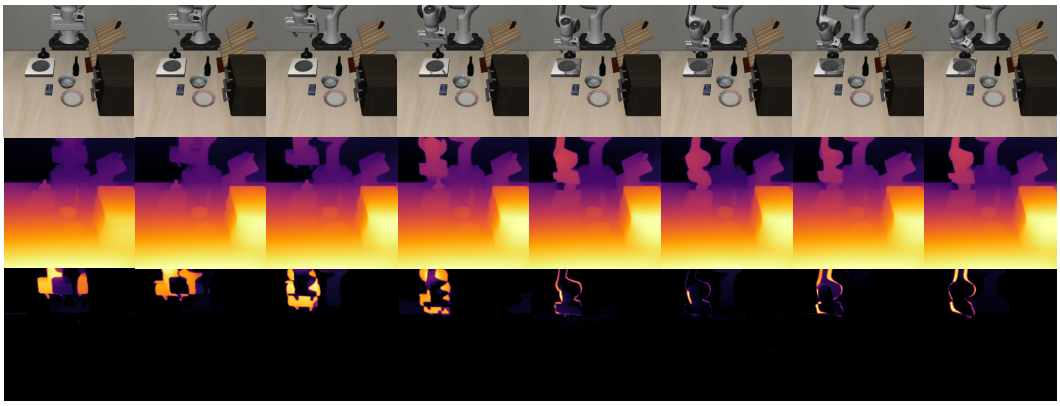


Figure 4: Factor probes on *turn_on_the_stove*. Rows show observations, depth reconstructed from spatial slots, and depth-change predictions from dynamics slots. These probes visualize decodable information; Appendix C.4.1 describes the diagnostic setup.

Further representation, attention, and trajectory analyses are reported in Appendix C.

5 Conclusion and Limitations

SCULPT-VLA connects structured supervision to control through staged action grounding. Closed-loop evaluations on LIBERO, SimplerEnv-WidowX, RoboTwin 2.0 Full, and four physical robot tasks support this design. Training controls and factor-wise functional interventions show the learned state's contribution after direct perceptual access is restored. The results suggest that learning how to use structure matters alongside learning what that structure encodes.

The current design uses fixed control factors and training-time teachers. Physical evaluation covers four tabletop tasks on one UR5e platform (Appendix E).

AI DISCLOSURE

Generative AI tools were used for language polishing of selected sentences. All scientific content, analyses, and reported results were reviewed and verified by the authors. Qwen3-VL-32B-Instruct was used in the automated subtask-annotation pipeline described in Appendix A.3.

REPRODUCIBILITY STATEMENT

Section 3.2 describes staged action grounding. Appendix A provides training and subtask-annotation details. Appendix B gives evaluation protocols, ablation definitions, and complete results. Appendix C describes the probes and factor-wise intervention procedure.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Shuanghao Bai, Jing Lyu, Wanqi Zhou, Zhe Li, Dakai Wang, Lei Xing, Xiaoguang Zhao, Pengwei Wang, Zhongyuan Wang, Cheng Chi, et al. Latent reasoning vla: Latent thinking and prediction for vision-language-action models. *arXiv preprint arXiv:2602.01166*, 2026a.

Yongjie Bai, Hanting Wang, Mingtong Dai, Qijun Zhong, Yang Liu, and Liang Lin. Bridge-WA: Predicting where and how the world changes for robotic action. *arXiv preprint arXiv:2607.02195*, 2026b. URL https://arxiv.org/abs/2607.02195.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025. URL https://arxiv.org/abs/2512.13030.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Zhejia Cai, Yandan Yang, Xinyuan Chang, Shiyi Liang, Ronghan Chen, Feng Xiong, Mu Xu, and Ruqi Huang. Seeing space and motion: Enhancing latent actions with geometric and dynamic awareness for vision-language-action models. *arXiv preprint arXiv:2509.26251*, 2026. URL https://arxiv.org/abs/2509.26251v2. Version 2, revised March 2026.

Jiayi Chen, Wenxuan Song, Pengxiang Ding, Ziyang Zhou, Han Zhao, Feilong Tang, Donglin Wang, and Haoang Li. Unified diffusion vla: Vision-language-action model via joint discrete denoising diffusion process. *arXiv preprint arXiv:2511.01718*, 2025a.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b. URL https://arxiv.org/abs/2506.18088.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

Yu Fang, Yuchun Feng, Dong Jing, Jiaqi Liu, Yue Yang, Zhenyu Wei, Daniel Szafir, and Mingyu Ding. When vision overrides language: Evaluating and mitigating counterfactual failures in vlas. *arXiv preprint arXiv:2602.17659*, 2026.

Bryce Grant, Xijia Zhao, and Peng Wang. Not all features are created equal: A mechanistic study of vision-language-action models. *arXiv preprint arXiv:2603.19233*, 2026.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025.

Chi-Pin Huang, Yunze Man, Zhiding Yu, Min-Hung Chen, Jan Kautz, Yu-Chiang Frank Wang, and Fu-En Yang. Fast-thinkact: Efficient vision-language-action reasoning via verbalizable latent planning. *arXiv preprint arXiv:2601.09708*, 2026.

Yufeng Ji, Wenhao Tang, Haoyi Niu, Koushil Sreenath, Yi Wu, and Zhongyu Li. Action QFormer: Structured representation shaping under action supervision in vision-language-action models. *arXiv preprint arXiv:2607.14635*, 2026. URL https://arxiv.org/abs/2607.14635.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Andrew Ting Yan Li, Zhuo Li, Zhelin Yang, Zhipeng Dong, Quentin Rouxel, and Fei Chen. Reasoning without inference cost: Latent semantic scaffolding for robot VLA policies. *arXiv preprint arXiv:2609.04893*, 2026. URL https://arxiv.org/abs/2609.04893.

Peiyan Li, Yixiang Chen, Hongtao Wu, Xiao Ma, Xiangnan Wu, Yan Huang, Liang Wang, Tao Kong, and Tieniu Tan. Bridgevla: Input-output alignment for efficient 3d manipulation learning with vision-language models. *arXiv preprint arXiv:2506.07961*, 2025.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

Shijie Lian, Bin Yu, Xiaopeng Lin, Laurence T. Yang, Zhaolong Shen, Changti Wu, Yuzhuo Miao, Cong Huang, and Kai Chen. LangForce: Bayesian decomposition of vision language action models via latent action queries. *arXiv preprint arXiv:2601.15197*, 2026. URL https://arxiv.org/abs/2601.15197v7. Version 7.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, et al. LaST_0: Latent spatio-temporal chain-of-thought for robotic vision-language-action model. *arXiv preprint arXiv:2601.05248*, 2026.

Qi Lv, Weijie Kong, Hao Li, Jia Zeng, Zherui Qiu, Delin Qu, Haoming Song, Qizhi Chen, Xiang Deng, and Jiangmiao Pang. F1: A vision-language-action model bridging understanding and generation to actions. *arXiv preprint arXiv:2509.06951*, 2025.

Soroush Nasiriany, Sean Kirmani, Tianli Ding, Laura Smith, Yuke Zhu, Danny Driess, Dorsa Sadigh, and Ted Xiao. RT-Affordance: Affordances are versatile intermediate representations for robot manipulation. *arXiv preprint arXiv:2411.02704*, 2024. URL https://arxiv.org/abs/2411.02704.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Lu Qiu, Yizhuo Li, Yi Chen, Yuying Ge, Yixiao Ge, and Xihui Liu. Making foresight actionable: Repurposing representation alignment in world action models. *arXiv preprint arXiv:2606.12217*, 2026. URL https://arxiv.org/abs/2606.12217.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Guoheng Sun, Kaixi Feng, Shwai He, Xiaochuan Gong, Yexiao He, Ziyao Wang, Zheyu Shen, Wanghao Ye, Ramana Rao Kompella, Gaowen Liu, and Ang Li. Drop-then-recovery: How redundant are vision-language-action models? *arXiv preprint arXiv:2606.27755*, 2026a. URL https://arxiv.org/abs/2606.27755.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. Vla-jepa: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026b.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023.

Chaoyang Wang, Wenrui Bao, Sicheng Gao, Bingxin Xu, Yu Tian, Yogesh S Rawat, Yunhao Ge, and Yuzhang Shang. Vla-thinker: Boosting vision-language-action models through thinking-with-image reasoning. *arXiv preprint arXiv:2603.14523*, 2026a.

Xiaofan Wang, Xingyu Gao, Jianlong Fu, Zuolei Li, Dean Fortier, Galen Mullins, Andrey Kolobov, and Baining Guo. Lola: Long horizon latent action learning for general robot manipulation. *arXiv preprint arXiv:2512.20166*, 2025a.

Xinkai Wang, Chenyi Wang, Yifu Xu, Mingzhe Ye, Fucheng Zhang, Jialin Tian, Xinyu Zhan, Lifeng Zhu, Cewu Lu, and Lixin Yang. Lamp: Learning vision-language-action policy with 3d scene flow as latent motion prior. *arXiv preprint arXiv:2603.25399*, 2026b.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pp. 18638–18646, 2026c.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025b. URL https://arxiv.org/abs/2506.19850.

Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, et al. OpenWAM: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026d. URL https://arxiv.org/abs/2609.07398.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic VLA foundation model. *arXiv preprint arXiv:2601.18692*, 2026. URL https://arxiv.org/abs/2601.18692.

Fuxiang Yang, Donglin Di, Lulu Tang, Xuancheng Zhang, Lei Fan, Hao Li, Chen Wei, Tonghua Su, and Baorui Ma. Chain of world: World model thinking in latent motion. *arXiv preprint arXiv:2603.03195*, 2026.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024.

Shuai Yang, Hao Li, Bin Wang, Yilun Chen, Yang Tian, Tai Wang, Hanqing Wang, Feng Zhao, Yiyi Liao, and Jiangmiao Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation. *arXiv preprint arXiv:2507.17520*, 2025.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026. URL https://arxiv.org/abs/2603.16666.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025a.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1702–1713, 2025.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025. URL https://arxiv.org/abs/2510.10274.

Yupeng Zheng, Xiang Li, Songen Gu, Yuhang Zheng, Shuai Tian, Weize Li, Linbo Wang, Chaoyue Li, Qichao Zhang, Haoran Li, Zhongpu Xia, Ya-Qin Zhang, Shuicheng Yan, and Dongbin Zhao. GIFT: Guided intermediate feature training via action-oriented structural supervision for robotic manipulation. *arXiv preprint arXiv:2609.04193*, 2026. URL https://arxiv.org/abs/2609.04193.

Zhide Zhong, Haodong Yan, Junfeng Li, Xiangchen Liu, Xin Gong, Tianran Zhang, Wenxuan Song, Jiayi Chen, Xinhu Zheng, Hesheng Wang, et al. Flowvla: Visual chain of thought-based motion reasoning for vision-language-action models. *arXiv preprint arXiv:2508.18269*, 2025.

Yijie Zhu, Jie He, Rui Shao, Kaishen Yuan, Tao Tan, Xiaochen Yuan, and Zitong Yu. DeltaVLA: Prior-guided vision-language-action models via world knowledge variation. *arXiv preprint arXiv:2603.08361*, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

APPENDIX

This appendix provides the training configuration, evaluation protocols and complete results, functional and representation analyses, additional rollouts, and the scope of the method.

	Contents	Page
A	Training and data construction	14
B	Evaluation protocols and complete results	18
C	Functional and representation analysis	22
D	Additional rollouts	28
E	Design scope and limitations	31

A TRAINING AND DATA CONSTRUCTION

A.1 TEACHER SCAFFOLDS AND TOKEN LAYOUT

Teacher inputs and alignment targets. The frozen teachers are Qwen3-Embedding for task semantics (Zhang et al., 2025b), UniVLA for task-relevant inverse-dynamics cues (Bu et al., 2025), and Depth Anything V2 for current-scene geometry (Yang et al., 2024). The corresponding slot counts are 4 for subtask, 4 for dynamics, and 8 for spatial representations. All benchmarks and real-robot tasks use these teacher sources and the same three-stage training design.

For each factor τ, the trainable input projector produces $\tilde{\mathbf{h}}_t^\tau = P_{\text{in}}^\tau(\mathbf{f}_t^\tau)$ from a detached teacher feature. This scaffold supplies the slot input before annealing. The alignment reference is separately detached, $\bar{\mathbf{h}}_t^\tau = \text{sg}(\tilde{\mathbf{h}}_t^\tau)$, so the target branch updates neither the teacher nor the input projector. Input projectors are trained through scaffold inputs and output projectors through alignment; the teacher encoders remain frozen. The projectors/adapters in Table 6 comprise teacher-to-slot input projections, student-side alignment projections, and slot-adaptation modules.

Training and deployment sequences. The formats below instantiate the three objectives in the main Method. All stages include visual and instruction tokens. Stage I adds the three factor groups; Stages II and III also use four fixed action readout slots. Stage II appends teacher-forced FAST positions for discrete supervision and anneals scaffold inputs into learned queries. Stage III continuous-control inputs contain factor and fixed readout slots, without teacher-forced FAST tokens or autoregressive generation. The continuous head conditions on these states and DINOv2 detail features. Weak FAST supervision uses a separate discrete auxiliary path; alignment targets remain active. Deployment omits teacher encoders and the auxiliary path.

A.2 OPTIMIZATION AND TRAINABILITY

Training configuration. Within each benchmark, the action horizon is fixed across the three stages. Unless otherwise specified, all stages use AdamW with a cosine learning-rate scheduler and warm-up ratio 0.1. Detailed settings are reported in Table 7.

Parameter adaptation. Table 6 specifies module trainability. Stage I updates slot-specific modules. Stage II adds LoRA updates to selected VLM attention projections and trains the action-related modules. Stage III trains the continuous head while co-training the previously learned pathway at the listed reduced learning rates. The base FFN and teacher encoders stay frozen. LoRA applies only to selected VLM linear projections; slot projectors, typed FFNs, condition fusion, and the continuous DiT head use full-rank optimization when active.

A.3 SUBTASK ANNOTATION PIPELINE

LIBERO-SUBTASK and **BRIDGE-SUBTASK** are constructed through an automated pipeline that decomposes an untrimmed manipulation video into a sequence of temporally grounded atomic task segments with aligned language instructions (Fig. 6). The pipeline comprises three stages:

Stage I: Latent Alignment
Task: "pick up the potato and place it into the bowl"
Input:
Your task is to pick up the potato and place it into the bowl.
<SOST> <slot_subtask>×4 <EOST>
<SOSD> <slot_dynamic>×4 <EOSD>
<SOSS> <slot_spatial>×8 <EOSS>
Main supervision:
Align <slot_subtask>, <slot_dynamic>,
and <slot_spatial> to frozen teacher targets under $\mathbf{M}^{(1)}$.
Note: scaffold-derived slot inputs are fully used;
coarse-action supervision and the act-typed group are not active.

Stage II: Scaffold-Input Annealing with Coarse Action Grounding
Task: "open the drawer and place the sponge inside"
Input:
Your task is to open the drawer and place the sponge inside.
<SOST> <slot_subtask>×4 <EOST>
<SOSD> <slot_dynamic>×4 <EOSD>
<SOSS> <slot_spatial>×8 <EOSS>
<SOSA> <slot_action>×4 <EOSA>
<SOFA> <fast_action>×M <EOFA>
Main supervision:
FAST next-token loss at act-typed FAST positions,
with continuing teacher alignment for <slot_subtask>,
<slot_dynamic>, and <slot_spatial> under $\mathbf{M}^{(2)}$.
Note: scaffold inputs to structured slots are annealed
into learned query embeddings; output-side
teacher alignment remains active throughout.

Stage III: Continuous Action Refinement
Task: "push the cup to the target mark"
Continuous-policy input:
Your task is to push the cup to the target mark.
<SOST> <slot_subtask>×4 <EOST>
<SOSD> <slot_dynamic>×4 <EOSD>
<SOSS> <slot_spatial>×8 <EOSS>
<SOSA> <slot_action>×4 <EOSA>
Main supervision:
Continuous action target via flow matching under $\mathbf{M}^{(3)}$.
Note: weak FAST supervision uses a separate
autoregressive path, outside this continuous-policy sequence.
The continuous head uses factor states, fixed action readout,
and DINOv2 detail features.

Figure 5: **Example training formats across the three stages of SCULPT-VLA.** <slot_subtask>, <slot_dynamic>, <slot_spatial>, and <slot_action> denote placeholder tokens reserved in the VLM sequence for subtask, dynamics, spatial, and action-side latent slots, respectively. Stage II additionally appends teacher-forced FAST action-token positions, assigned type act, for coarse next-token supervision. The Stage III continuous sequence excludes FAST tokens; weak FAST supervision uses a separate autoregressive path. The continuous head reads final factor and action-readout states, together with DINOv2 detail features. Boundary tokens such as <SOST> / <EOST> mark the start and end of each slot block for type assignment, masking, and stage-specific loss computation.

window-wise transition prediction, global boundary consolidation, and segment-level instruction assignment. The output provides the subtask-stage signal used by Qwen3-Embedding when generating instruction-aware features (Zhang et al., 2025b) for the subtask slots. The same annotation procedure is applied to RoboTwin and real-robot demonstrations to construct their task-progression scaffolds. These training annotations are separate from the analysis-only phase labels used in Appendix C.4.2.

Table 6: Stage-wise module trainability in SCULPT-VLA. ✓ denotes trainable modules, × denotes frozen modules, and △ denotes modules co-trained with a reduced learning rate. For the VLM backbone, "trainable" denotes LoRA-updated linear projections rather than full-rank backbone training.

Stage	Teacher Encoders	VLM Attn LoRA	Base-FFN	Sub./Dyn./Spa. FFNs	Action FFN	Slot Proj./ Adapters	FAST Action LM Head	DiT Action Head
Stage I	×	×	×	✓	×	✓	×	×
Stage II	×	✓	×	✓	✓	✓	✓	×
Stage III	×	△	×	△	△	✓	△	✓

Table 7: Hyperparameter settings of SCULPT-VLA across LIBERO, RoboTwin 2.0, SimplerEnv, and real-robot experiments. *Scheduled* indicates scaffold-input annealing or coarse-loss weighting following the schedule in Sec. 3.2; *Weak* denotes retained at a small weight as auxiliary regularization.

Hyperparameters	LIBERO / RoboTwin 2.0			SimplerEnv			Real Robot		
	Stage I	Stage II	Stage III	Stage I	Stage II	Stage III	Stage I	Stage II	Stage III
Learning Rates									
VLM LR	—	1×10^{-5}	5×10^{-6}	—	1×10^{-5}	5×10^{-6}	—	1×10^{-5}	5×10^{-6}
Teacher Encoder LR	—	—	—	—	—	—	—	—	—
Slot Projector LR	1×10^{-4}	1×10^{-4}	5×10^{-5}	1×10^{-4}	1×10^{-4}	5×10^{-5}	1×10^{-4}	1×10^{-4}	5×10^{-5}
FAST Action Head LR	—	1×10^{-4}	5×10^{-5}	—	1×10^{-4}	5×10^{-5}	—	1×10^{-4}	5×10^{-5}
Continuous Action Head LR	—	—	1×10^{-4}	—	—	1×10^{-4}	—	—	1×10^{-4}
Optimization Config									
Action Horizon	8	8	8	8	8	8	16	16	16
Training Steps	5k	20k	20k	5k	20k	20k	3k	15k	15k
Batch Size	8	4	8	8	4	8	8	4	8
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
LR Scheduler	Cosine	Cosine	Cosine	Cosine	Cosine	Cosine	Cosine	Cosine	Cosine
Warm-up Ratio	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Loss Weights									
Slot Alignment Loss (Sub./Dyn./Spa.)	1.0/1.0/1.0	Scheduled	Weak	1.0/1.0/1.0	Scheduled	Weak	1.0/1.0/1.0	Scheduled	Weak
Coarse Action Loss	—	Scheduled	Weak	—	Scheduled	Weak	—	Scheduled	Weak
Continuous FM Loss	—	—	Main	—	—	Main	—	—	Main
Continuous Action Head	Off	Off	On	Off	Off	On	Off	Off	On

Implementation parameters. Unless otherwise specified, videos are processed at their native frame rate with temporal windows of 64 frames and stride 32. For each window, we uniformly sample 8 frames and provide them to the locally hosted Qwen3-VL-32B-Instruct model together with a compact object-inventory prompt. The annotation model is constrained to return a JSON record with fields `thought`, `transitions`, `instructions`, and `boundary_confidence`; these are mapped to the internal metadata fields `rationale`, `transition_frames`, and `segment_instructions` for downstream use. Outputs that fail JSON parsing or contain non-monotone transition indices are regenerated once and otherwise discarded. Cross-window transition proposals within 12 frames are merged by taking the median proposal time, and segments shorter than 8 frames after consolidation are removed. Boundary confidence is the VLM-reported confidence for each proposed transition after cross-window consolidation. It serves as a filtering and quality-control score, not a training target. We manually inspect all parse-failure cases and a random 10% subset of valid annotations.

Source scale. LIBERO-SUBTASK is derived from the four standard LIBERO suites used in our evaluation, comprising 2,000 source demonstrations in total (4 suites × 10 tasks × 50 demonstrations) (Liu et al., 2023). BRIDGE-SUBTASK is derived from the WidowX/BridgeData source pool used for SimplerEnv-style training; public BridgeData V2 releases are on the order of 5×10^4–6×10^4 trajectories depending on filtering and release version (Walke et al., 2023). For RoboTwin 2.0, we use the Full training set comprising 50 tasks, with 50 clean and 500 randomized demonstrations per task, for 27,500 source demonstrations in total. The same annotation pipeline is applied to these demonstrations to construct task-progression scaffolds. Since the pipeline may split one source trajectory into multiple atomic segments and discard invalid or very short proposals, we report source-trajectory scale here and store the retained segment count and mean segment length in the generated annotation metadata.

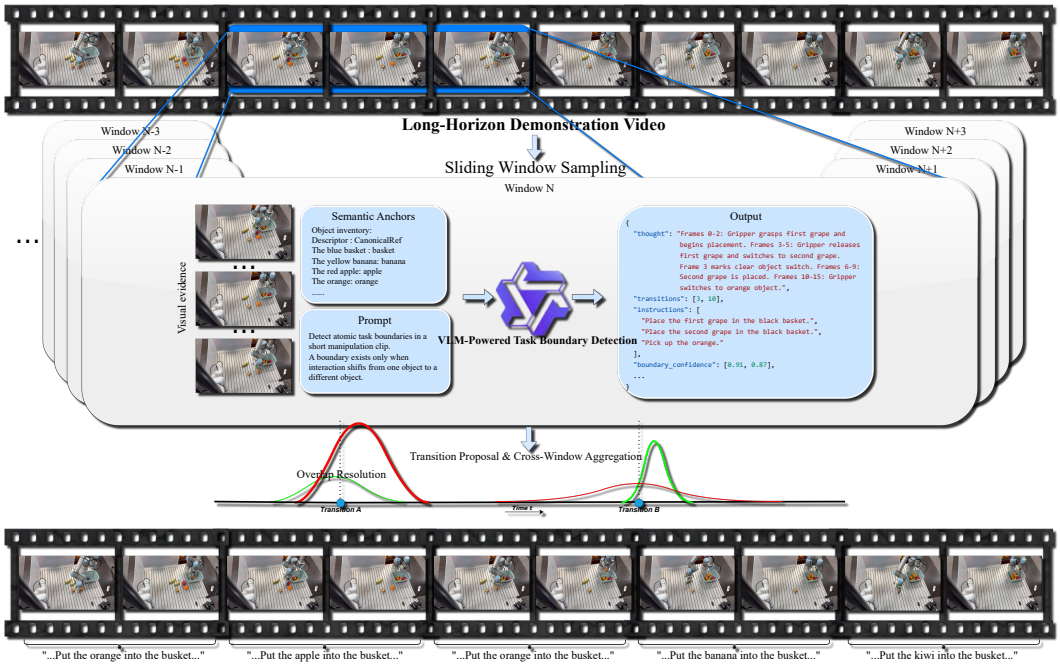


Figure 6: Automated long-video segmentation pipeline. An untrimmed demonstration video is decomposed into overlapping temporal windows. Within each window, sampled visual evidence and semantic anchors are provided to Qwen3-VL-32B-Instruct for object-centric transition prediction. The resulting transition proposals are mapped to the global timeline and merged through cross-window aggregation to remove redundant detections from overlapping windows. The final consolidated boundaries partition the original demonstration into atomic task segments with aligned language instructions.

Window-wise transition prediction. Within each temporal window, the sampled frames are paired with semantic anchors specifying the local object inventory and canonical object references. These inputs are provided to Qwen3-VL-32B-Instruct prompted to detect atomic task transitions under the fixed JSON schema. We define a transition as a switch in interaction from one manipulated object to another. This criterion suppresses spurious boundaries from motion within a single object-centric manipulation phase.

Global boundary consolidation. Local transition indices are mapped onto the global video timeline. Overlapping windows often predict the same boundary with small temporal offsets. We consolidate these proposals using the temporal merge tolerance above, yielding a consistent segmentation of the demonstration.

Instruction assignment. The instruction candidates generated in local windows are aligned with the consolidated temporal segments to assign one language instruction to each atomic task interval. Candidate instructions that disagree with the consolidated start/end interval are filtered during the same metadata validation pass.

A.4 Annotation Pipeline Statistics

Table 8 summarizes parsing validity, discarded segments, and VLM-reported boundary confidence after consolidation.

Table 8: Processing statistics of the derived subtask annotations. Validity and discard rates are computed after one regeneration attempt; confidence denotes the percentage of valid annotations whose VLM-reported boundary confidence exceeds 80%.

Dataset	**JSON valid**	**Discarded**	**Conf.** $> 80\%$
LIBERO-SUBTASK	98.1%	3.0%	94.8%
BRIDGE-SUBTASK	96.2%	5.8%	91.5%

B Evaluation Protocols and Complete Results

This section details the protocols for the main comparisons in Tables 1 and 2, followed by simulation seed counts, ablation definitions, and real-world results.

B.1 Comparison Scope and Evaluation Protocols

Published methods retain their source-reported protocols. Shared-backbone controls use Qwen3-VL under the StarVLA evaluation stack, with shared robot data, action horizon, and evaluation interface (Community, 2026; Bai et al., 2025). They use FAST, OFT, π_0-style flow, or GR00T-style action heads (Pertsch et al., 2025; Kim et al., 2025; Black et al., 2024; Bjorck et al., 2025). Protocol differences affecting direct comparability are noted in the corresponding table captions, including the † and ‡ entries in Table 2.

SCULPT-VLA follows the RoboTwin 2.0 Full protocol (Chen et al., 2025b; Community, 2026), with joint training over 50 tasks using 50 clean and 500 randomized demonstrations per task. A single final checkpoint is evaluated on 100 Clean and 100 Random episodes for each task. Clean and Random scores are arithmetic means of the 50 task-level success rates; Avg. averages these two scores. Both conditions are represented during training. Table 2 reports the main comparison, and Table 9 gives additional published results.

Table 9: Additional published RoboTwin 2.0 Full results (success, %). X-VLA results follow Bi et al. (2025), and $\pi_{0.5}$ results follow Yuan et al. (2026). Avg. is the arithmetic mean of Clean and Random.

Method	Clean	Random	Avg.
X-VLA (Zheng et al., 2025)	72.8	72.8	72.8
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	82.7	76.8	79.8

B.2 Open-Loop Evaluation Details

We additionally measure agreement with held-out demonstration action chunks on SimplerEnv-WidowX (Li et al., 2024) and the real-world ID/OOD validation splits. The latter are described in Appendix B.6. Each prediction is compared with the corresponding expert action chunk.

Formally, let N denote the number of evaluation transitions and let H denote the action horizon. For transition i, the model predicts an action chunk $\hat{\mathbf{a}}_{i,0:H-1}$, which is compared with the corresponding expert chunk $\mathbf{a}^*_{i,0:H-1}$. OLS is computed on the normalized 6-DoF continuous action component only, excluding the gripper dimension. For a raw 6-DoF action vector $\mathbf{a}$, we define the normalization map

$$\phi(\mathbf{a})_d = \frac{\mathbf{a}_d - \mu_d}{s_d + \epsilon}, \qquad s_d = \mathrm{P}_{95}(|\mathbf{a}_d - \mu_d|)\,,$$

where μ_d and s_d are computed from the training split for each translation or rotation dimension, P_{95} denotes the 95th percentile, and $\epsilon = 10^{-6}$. Translation and rotation errors are therefore measured in

the same normalized action space. For threshold δ,

$$\mathrm{OLS@}\delta = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\left(\frac{1}{H}\sum_{k=0}^{H-1}\left\|\phi(\hat{\mathbf{a}}_{i,k}) - \phi(\mathbf{a}^{*}_{i,k})\right\|_2 < \delta\right) \tag{11}$$

A predicted chunk counts as correct when its average per-step Euclidean error is below δ in normalized 6-DoF action space. The thresholds in Δ are dimensionless.

We further report the mean open-loop score

$$\mathrm{mOLS} = \frac{1}{|\Delta|}\sum_{\delta\in\Delta}\mathrm{OLS@}\delta, \qquad \Delta = \{0.1,\, 0.05,\, 0.03,\, 0.01\} \tag{12}$$

Higher OLS and mOLS indicate closer agreement with expert action chunks. Both metrics are averaged over all sampled transitions in the held-out split.

B.3 Simulation Results and Seed Counts

Table 10 reports the seed-level closed-loop success counts of SCULPT-VLA for the main simulation results on LIBERO (Liu et al., 2023) and SimplerEnv-WidowX (Li et al., 2024). For LIBERO, each suite-level entry aggregates 10 tasks with 50 trials per task, yielding 500 trials per seed. For SimplerEnv-WidowX, each displayed task is evaluated with 50 trials per seed. The averages reported in Table 1 are computed from the corresponding seed-level success rates; suite-level and task-level averages are reported as arithmetic means of the displayed scores.

Table 10: Seed-level closed-loop success counts of SCULPT-VLA on LIBERO and SimplerEnv-WidowX. LIBERO rows report successful trials out of 500 per seed, corresponding to 10 tasks with 50 trials each. SimplerEnv-WidowX rows report successful trials out of 50 per seed for each displayed task. Success Rate is mean $\pm$ sample SD over three seeds.

Benchmark / Task	**Seed 1**	**Seed 2**	**Seed 3**	**Success Rate (%)**
LIBERO-Spatial	496/500	494/500	497/500	99.1 ± 0.3
LIBERO-Object	498/500	496/500	499/500	99.5 ± 0.3
LIBERO-Goal	492/500	493/500	489/500	98.3 ± 0.4
LIBERO-Long	485/500	482/500	487/500	96.9 ± 0.5
SimplerEnv-Spoon	45/50	45/50	46/50	90.7 ± 1.2
SimplerEnv-Carrot	35/50	36/50	35/50	70.7 ± 1.2
SimplerEnv-Stack	38/50	37/50	38/50	75.3 ± 1.2
SimplerEnv-Eggplant	49/50	49/50	48/50	97.3 ± 1.2

For SimplerEnv-WidowX, each seed totals 167 successes in 200 trials. The four-task aggregate is therefore $83.5 \pm 0.0\%$, although individual task scores vary across seeds.

B.4 Ablation Protocol and Variant Definitions

Protocol. SimplerEnv-WidowX ablations report success as mean $\pm$ sample SD over evaluation runs (Li et al., 2024). Unless changed by the intervention, variants share the Qwen3-VL backbone, action data, horizon, optimizer, teacher targets, FAST tokenizer (Pertsch et al., 2025), flow head, and evaluation. Each variant is trained in independent runs. FAST denotes the Stage-II policy and continuous success the Stage-III policy. We report FAST success for variants trained with the Stage-II FAST objective; a dash denotes variants without that supervised Stage-II policy. OLS/mOLS follows Appendix B.2. Table 11 retains all variants and thresholds.

Table 11: Ablation results on SimplerEnv-WidowX Avg. We report closed-loop success under Stage-II FAST decoding and Stage-III continuous control, together with open-loop OLS and mOLS. Success rates are mean ± sample SD over evaluation runs; variants are trained independently. OLS and mOLS are defined in Appendix B.2. Dashes indicate variants without a supervised Stage-II FAST policy.

Variant	Success Rate (%)		OLS ↑				mOLS ↑
	FAST	Cont.	@0.1	@0.05	@0.03	@0.01	
Full SCULPT-VLA	74.7±1.0	83.5±0.0	0.738	0.431	0.248	0.057	0.369
shared factor FFN	65.7±1.0	76.8±1.0	0.697	0.379	0.201	0.039	0.329
generic latent pool	52.8±1.2	63.9±1.1	0.632	0.301	0.132	0.020	0.271
w/o subtask factor	67.3±1.3	78.3±1.3	0.713	0.397	0.216	0.044	0.342
w/o spatial factor	64.8±0.8	72.2±0.8	0.680	0.356	0.183	0.031	0.312
w/o dynamics factor	61.6±1.0	73.3±0.8	0.687	0.372	0.193	0.036	0.322
w/o action-side latent slots $\mathbf{h}^{\mathrm{act}}$	69.2±1.3	77.0±1.0	0.706	0.386	0.207	0.040	0.335
w/o Stage-II visibility restriction	55.7±0.8	71.3±1.0	0.671	0.342	0.168	0.028	0.302
w/o Stage I	60.0±1.0	70.2±1.0	0.667	0.343	0.166	0.027	0.301
Stage III only	—	61.0±1.0	0.621	0.287	0.121	0.018	0.262
w/o FAST supervision	—	68.2±1.3	0.651	0.329	0.154	0.025	0.290
w/o DINOv2 detail branch	73.7±1.0	81.0±1.0	0.724	0.414	0.229	0.049	0.354

Representation controls. The *generic latent pool* removes explicit type identities, factor-specific residual FFNs, and type-dependent visibility constraints. Teacher targets remain assigned to the original 4 subtask, 4 dynamics, and 8 spatial positions. Token count, DINOv2, both action objectives, and the three-stage schedule are unchanged. This control tests explicit factor organization under the same supervision.

The *shared factor FFN* replaces $\Delta\mathrm{FFN}_{\mathrm{sub}}$, $\Delta\mathrm{FFN}_{\mathrm{dyn}}$, and $\Delta\mathrm{FFN}_{\mathrm{spa}}$ with one $\Delta\mathrm{FFN}_{\mathrm{factor}}$. Type labels, visibility, and the action branch $\Delta\mathrm{FFN}_{\mathrm{act}}$ remain unchanged.

Each *factor-removal* variant removes one factor's slots and scaffold loss. The *w/o action-side latent slots* variant removes $\mathbf{h}^{\mathrm{act}}$ and decodes from the factor states. The *w/o Stage-II visibility restriction* variant permits act-typed positions to read vision and language directly.

Training controls. *w/o Stage I* skips alignment warm-up. *Stage III only* omits the first two stages while retaining the SCULPT-VLA architecture and auxiliary losses, distinguishing it from the direct Qwen3-VL-PI baseline. *w/o FAST supervision* removes the FAST objective while preserving staged training.

The *w/o DINOv2 detail branch* variant removes the additional visual features from the Stage-III head. Its Stage-II checkpoint comes from an independent training run. Since the branch is introduced in Stage III, the FAST-score difference reflects variation between training runs.

Interpretation. The shared-FFN control isolates residual sharing under typed labels and visibility, whereas the generic pool changes all three together. These interventions test complementary design choices rather than additive contributions. Stage-removal comparisons differ in total optimization budget under the reported schedules.

B.5 Real-World Experimental Setup

Hardware and control. We use a Universal Robots UR5e arm with a three-finger flexible gripper on a $1.0 \times 0.5 \times 0.70$ m tabletop workspace. At each policy query, the model outputs an action chunk of length $H = 16$, where each action contains a 6-DoF end-effector pose target and a gripper command. The policy is queried at approximately 7.8 Hz, and the resulting targets are executed by an RTDE position controller running at 125 Hz. Visual observations are provided by synchronized Intel RealSense D435i cameras, including a wrist-mounted view for local interaction and a third-person view for global scene context. Qualitative figures in this appendix are rendered from the third-person view.

Task definitions and OOD settings. Table 12 summarizes the real-world tasks, success criteria, and OOD perturbations. ID settings reuse the demonstration setup, while OOD settings perturb object, layout, or visual factors not seen in the corresponding demonstrations.

Table 12: Real-world task definitions and OOD perturbations.

Task	Success criterion	ID setting	OOD perturbations
Fruit Collecting	All target fruits are picked and placed into the basket without dropping, collision-induced failure, or premature termination.	Training fruit instances and canonical basket layout.	Unseen fruit instances, changed fruit order, and altered initial object positions.
Paper Cup Stacking	The target cup is lifted, aligned, and stably stacked on the base cup.	Training cup colors/textures and nominal relative placement.	Novel cup colors/textures, shifted cup positions, and altered approach directions.
Toothbrush Insertion	The toothbrush is grasped, transported, and inserted into the target container with stable final placement.	Training toothbrush/container pair and nominal insertion pose.	Unseen toothbrush or container appearance, shifted container position, and altered insertion orientation.
Cloth Folding	The cloth is folded along the intended direction and remains approximately aligned after release.	Training cloth appearance and canonical tabletop placement.	Changed cloth position/orientation, visual texture variation, and perturbed initial spread.

Complete real-world results. Table 13 supplements the main success table with open-loop scores for every task and condition.

Table 13: Complete real-world open-loop and closed-loop results. Each task has ID and OOD columns for OLS@0.1, mOLS, and success rate (SC, %). Open-loop scores use held-out expert trajectories. SC is mean ± sample SD across three independent evaluation batches of 30 physical rollouts per condition, using one checkpoint per method/task.

		ID			OOD		
Method	Task	OLS@0.1	mOLS	SC	OLS@0.1	mOLS	SC
OpenVLA-OFT	Fruit Collecting	0.648	0.271	50.0±3.3	0.529	0.203	24.4±5.1
	Paper Cup Stacking	0.689	0.298	61.1±1.9	0.596	0.239	36.7±6.7
	Toothbrush Insertion	0.629	0.263	44.4±1.9	0.485	0.184	17.8±5.1
	Cloth Folding	0.556	0.224	25.6±1.9	0.427	0.129	2.2±1.9
π_0	Fruit Collecting	0.704	0.319	65.6±1.9	0.637	0.276	45.6±6.9
	Paper Cup Stacking	0.771	0.353	75.6±1.9	0.668	0.302	56.7±3.3
	Toothbrush Insertion	0.724	0.337	67.8±3.8	0.619	0.282	44.4±5.1
	Cloth Folding	0.641	0.273	35.6±1.9	0.536	0.185	14.4±3.8
$\pi_{0.5}$	Fruit Collecting	0.741	0.345	74.4±1.9	0.653	0.292	52.2±5.1
	Paper Cup Stacking	0.796	0.362	81.1±1.9	0.705	0.323	60.0±6.7
	Toothbrush Insertion	0.748	0.351	73.3±3.3	0.646	0.294	47.8±3.8
	Cloth Folding	0.665	0.288	41.1±1.9	0.574	0.229	22.2±5.1
SCULPT-VLA	Fruit Collecting	0.758	0.353	80.0±3.3	0.681	0.304	60.0±3.3
	Paper Cup Stacking	0.791	0.366	85.6±1.9	0.732	0.336	70.0±6.7
	Toothbrush Insertion	0.762	0.356	84.4±5.1	0.688	0.308	64.4±5.1
	Cloth Folding	0.691	0.299	56.7±3.3	0.603	0.227	37.8±5.1

B.6 Statistical Reporting for Real-World Evaluation

Open-loop validation data. For real-world open-loop evaluation, we use held-out expert trajectories collected separately for ID and OOD settings. Training uses only the ID demonstrations reported in the main text. The ID validation trajectories follow the training distribution, while the OOD validation trajectories follow the same perturbation families used for the closed-loop OOD evaluation in Table 12. The validation trajectories are reserved for OLS/mOLS computation and excluded from training, scaffold construction, and model selection. During open-loop evaluation, the policy receives only the observation and instruction at each timestep; expert action chunks are used solely as references for computing OLS/mOLS following Appendix B.2.

Closed-loop statistical reporting. Each method uses one trained checkpoint per task. For each ID/OOD condition, success (SC) is measured in three independent evaluation batches of 30 physical

rollouts, totaling 90 rollouts per condition. Mean ± sample SD reports variation across these three evaluation batches. Table 14 gives the counts.

Table 14: Real-world SC counts across evaluation batches. Each entry reports successful rollouts out of 30 for Batch 1/Batch 2/Batch 3, using one trained checkpoint per method and task.

Method	Fruit ID	Fruit OOD	Cup ID	Cup OOD	Toothbrush ID	Toothbrush OOD	Cloth ID	Cloth OOD
OpenVLA-OFT	15/14/16	7/9/6	19/18/18	11/9/13	13/14/13	5/7/4	8/8/7	1/1/0
π_0	20/19/20	13/16/12	23/22/23	17/16/18	21/19/21	13/15/12	11/11/10	5/5/3
$\pi_{0.5}$	22/22/23	16/14/17	24/25/24	18/20/16	22/23/21	15/15/13	13/12/12	8/7/5
SCULPT-VLA	24/25/23	18/17/19	25/26/26	21/23/19	25/24/27	19/21/18	18/17/16	13/11/10

Variation across OOD tasks. The OOD gains over $\pi_{0.5}$ are largest on Toothbrush Insertion ($+16.7$ pp) and Cloth Folding ($+15.6$ pp), followed by Paper Cup Stacking ($+10.0$ pp) and Fruit Collecting ($+7.8$ pp). These tasks differ in both interaction demands and perturbation composition (Table 12). The task-level results characterize performance under these combined perturbations.

C Functional Analysis of Learned Structured Control States

The following analyses distinguish group geometry, decodable content, attention allocation, and functional policy dependence. They supplement Section 4.4 of the main paper.

C.1 Analysis Setting

The stage-wise geometry and group-level attention analyses use the LIBERO 40-task video collection (Liu et al., 2023). We decode each video frame by frame and extract final-layer hidden states and attention. Task-specific probes, trajectory analysis, and the real-world intervention use the settings described below. Tokens are partitioned into six groups: visual (`vis`), instruction (`lang`), subtask (`sub`), dynamics (`dyn`), spatial (`spa`), and action-side (`act`). The action group includes fixed readout slots $\mathbf{h}^{\text{act}}$ and, in discrete-policy forwards, FAST positions. Stage-III continuous forwards contain no FAST positions.

For geometry visualization, we average the final hidden states within each group to obtain a frame-level representation:

$$\mathbf{h}_g^{(t)} = \frac{1}{|S_g^{(t)}|} \sum_{i \in S_g^{(t)}} \mathbf{h}_i^{(t)}, \tag{13}$$

where $S_g^{(t)}$ is the set of tokens belonging to group g in frame t. Stage I, Stage II, and Stage III are embedded into a shared PCA+UMAP space so that their low-dimensional geometries are directly comparable. Silhouette scores are computed in the original hidden space, avoiding distortion from the nonlinear 2D UMAP projection.

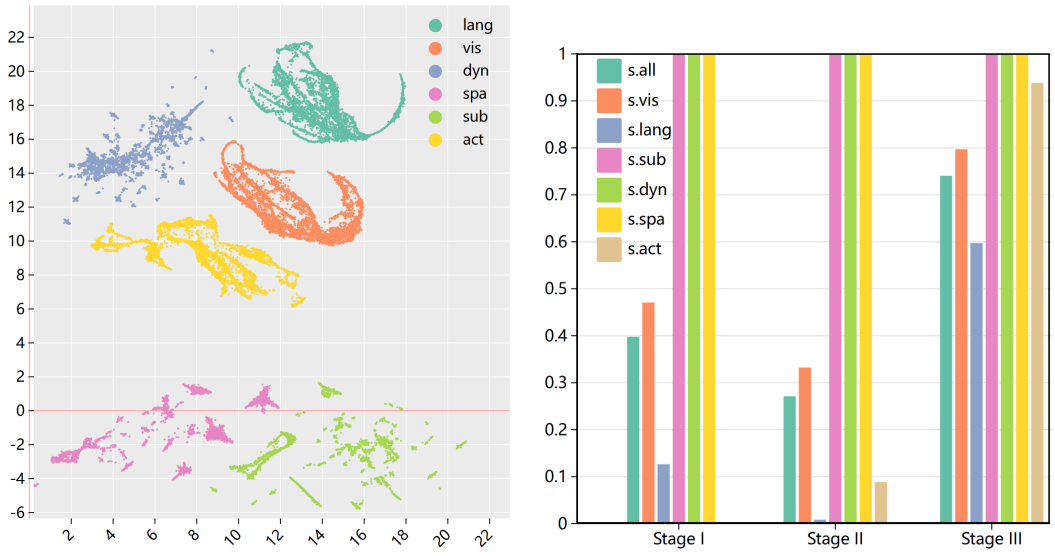
For routing analysis, we use the final-layer attention matrix after head averaging. Given a query group q and a key group k, we compute the query-normalized attention mass routed from q to k:

$$A(q,k) = \frac{1}{|Q_q|} \sum_{i \in Q_q} \sum_{j \in K_k} \text{Attn}_{ij}, \tag{14}$$

where Q_q and K_k denote the sets of query and key positions belonging to groups q and k, respectively. Since attention weights are normalized over key positions for each query token, $A(q,k)$ measures the fraction of attention mass from an average query in group q assigned to key group k. These values are averaged over all frames to form the stage-level Sankey diagrams.

C.2 Stage-Wise Latent Geometry

Figs. 8–9 extend the Stage III overview into a complete three-stage view, separating an all-group UMAP from a factor/action-group UMAP.



(a) Stage-III group representations.

(b) Per-group silhouette scores.

Figure 7: Group-level representation diagnostics. UMAP uses frame-level group means. Silhouette scores are calculated in the original hidden space.

The factor/action-group projection separates the three factor groups at Stage I. The all-group projection changes most at Stage II and shows more distinct groups at Stage III. These patterns describe the projected geometry of the frame-level group representations, whose separation is encouraged by factor-specific embeddings and teacher supervision.

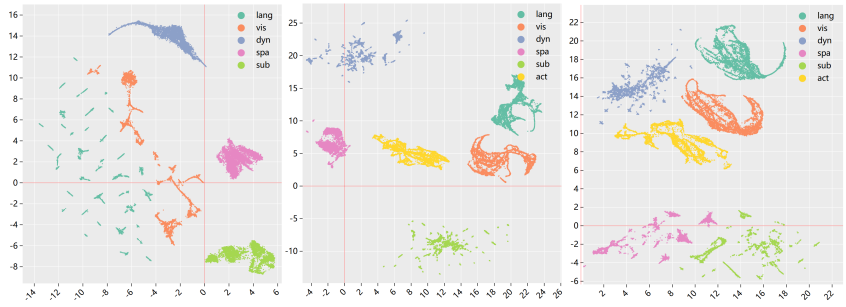


Figure 8: All-stage group-level UMAP in a shared PCA+UMAP space. Points are frame-level means for the active token groups; the action group appears from Stage II onward. All stages share the same fitted projection.

Silhouette scores in the original hidden space complement the projected geometry (Fig. 7b). The factor groups remain distinguishable across stages. Visual and language groups show lower separation in Stage II and recover in Stage III, while action-group separation increases from Stage II to Stage III.

C.3 Routing Dynamics Across Stages

The Sankey diagrams summarize final-layer attention mass between token groups. Stage I blocks cross-factor attention, so its within-group pattern is partly determined by the mask. Stage II permits cross-factor attention but blocks direct vision/language access by action tokens. Only Stage III permits both routes. Attention describes information allocation under each mask; factor-wise intervention separately tests the policy's functional use of the resulting states.



Figure 9: Factor- and action-group UMAP across training stages. The three factor groups occupy distinct regions from Stage I onward; the action group is included in Stages II and III.

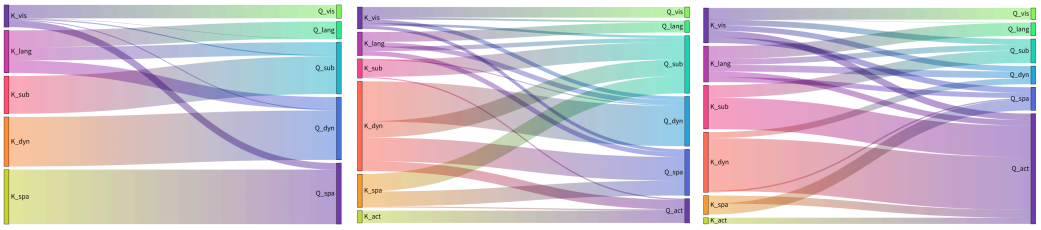


Figure 10: Stage-wise final-layer attention mass. Panels show Stages I, II, and III. Left nodes are key groups and right nodes are query groups. Band widths represent attention mass between groups.

C.3.1 Effect of Stage-II Perceptual Access

Stage III restores direct vision/language access for action tokens. To examine the effect of introducing this access earlier, we compare the standard schedule with a variant that uses $\mathbf{M}^{(3)}$ in both Stages II and III. On LIBERO-Spatial, success is 98.7% for this variant and 99.1% for the standard schedule (Table 15).

Table 15: Stage-II perceptual-access diagnostic on LIBERO-Spatial. Success is the three-seed mean under the main evaluation protocol. The w/o Stage-II visibility restriction variant uses $\mathbf{M}^{(3)}$ in both Stages II and III, allowing act-typed positions to read vision and language throughout action grounding.

Variant	Stage-II Mask	Stage-III Mask	LIBERO-Spatial Success (%)
Standard SCULPT-VLA	$\mathbf{M}^{(2)}$	$\mathbf{M}^{(3)}$	99.1
w/o Stage-II visibility restriction	$\mathbf{M}^{(3)}$	$\mathbf{M}^{(3)}$	98.7

The effect is smaller than on SimplerEnv-WidowX, indicating that the benefit of restricted Stage-II access depends on the task.

C.4 Factor Specialization and Functional Use

Spatial and dynamics probes characterize decodable content, while trajectory-level subtask geometry and slot-to-visual attention describe factor specialization. Inference-time replacement tests how final factor states contribute to the Stage-III continuous policy.

C.4.1 Spatial and Dynamics Probes

Probe design. The spatial probe reconstructs a dense current-scene depth map. The dynamics probe predicts a depth-residual map to characterize task-relevant state transitions. Both probes are lightweight readouts trained separately with the policy frozen, and provide qualitative visualizations of the learned representations.

Spatial-slot probe. A lightweight probe maps spatial-slot states to a depth prediction $\hat{D}_t^{\text{spa}}$. Figure 11 compares it with the current-depth target; the reconstruction preserves the tabletop, manipulator silhouette, and coarse object layout.

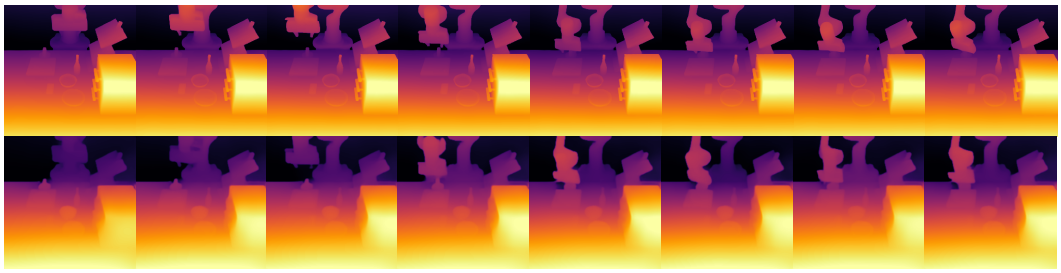


Figure 11: Spatial-slot probe against the depth target. Top: target depth map of the current observation; bottom: depth map reconstructed from spatial slots. The qualitative agreement in overall geometry is consistent with spatial slots preserving dense scene structure and spatial priors.

Dynamics-slot probe. To probe transition information, we construct a target from local depth changes between frames. Given two neighboring frames with monocular depth estimates D_t and $D_{t+\Delta}$, we first perform affine alignment to mitigate global scale drift. We use Δ corresponding to approximately one second in the processed videos.

$$D'_{t+\Delta} = aD_{t+\Delta} + b, \tag{15}$$

where (a, b) are obtained by least-squares fitting so that $D'_{t+\Delta}$ best matches D_t. We then compute the absolute residual

$$R_t = \left|D_t - D'_{t+\Delta}\right| \tag{16}$$

To suppress high-frequency noise, we apply Gaussian smoothing to R_t with $\sigma = 1.0$, threshold the smoothed residual at its 85th percentile, apply binary opening with a 3×3 structuring element, and remove connected components smaller than 0.5% of the image area. The processed residual provides a sparse target for the dynamics probe, emphasizing salient local depth changes. Its predicted visualization is $\widehat{\Delta D}_t^{\text{dyn}}$.

Probe observations. The spatial reconstruction shows overall scene geometry, whereas the dynamics probe responds around changing interaction regions (Fig. 4). These separately trained probes characterize decodable geometry and transition information under their respective targets.

C.4.2 Subtask-Slot PCA Analysis

Analysis method. We use trajectory-level PCA to examine whether subtask-slot representations track execution phases in latent space.

Data construction. We perform this analysis on the *turn_on_the_stove* task using 100 trajectories. Each trajectory is divided post hoc into five execution phases: *reach*, *align*, *grasp*, *turn*, and *finish*. These post-hoc labels are used only to color the extracted representations and are independent of the object-switch annotations used for training. For each frame, we extract the hidden states of the subtask-slot tokens and average them over the slot dimension to obtain a frame-level subtask representation:

$$\bar{\mathbf{h}}_t^{\text{sub}} = \frac{1}{N_{\text{sub}}} \sum_{i=1}^{N_{\text{sub}}} \mathbf{h}_{t,i}^{\text{sub}}, \tag{17}$$

where N_{sub} is the number of subtask-slot tokens (N_{sub}=4). Frame-level representations from all trajectories are aggregated and projected into PCA space for visualization.

Phase-associated geometry. The projected representations vary with the annotated execution phases. Adjacent phases occupy connected regions, while *finish* is comparatively compact. The 3D view separates some regions that overlap in 2D (Fig. 12). These patterns associate subtask-slot geometry with execution progress.

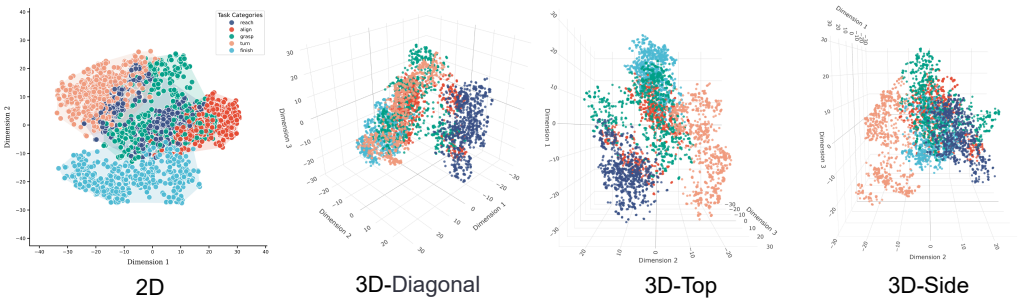


Figure 12: PCA of frame-level subtask-slot means from 100 trajectories of *turn_on_the_stove*. Colors indicate five post-hoc, analysis-only execution phases. The 2D and 3D views show phase-associated geometry, including connected neighboring regions and a compact terminal region.

C.4.3 Attention-Based Slot Specialization

We visualize final-layer slot-to-visual attention (Figs. 13–15) to examine factor specialization. For each branch $b \in \{\mathrm{sub}, \mathrm{dyn}, \mathrm{spa}\}$, the branch-to-visual attention response is

$$\bar{\mathbf{a}}^{(b)} = \frac{1}{|\mathcal{H}|\,|\mathcal{S}_b|} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}_b} \mathbf{A}^{(L,h)}_{s,\mathcal{V}},$$

where $\mathcal{S}_b$ and $\mathcal{V}$ are the branch slot and visual token sets, and $\mathbf{A}^{(L,h)}$ is the final-layer attention matrix. The 7×7 visual token response map is min-max normalized, threshold-masked, and bilinearly upsampled to 14×14 for display. The overlays retain the coarse spatial resolution of the visual token grid.

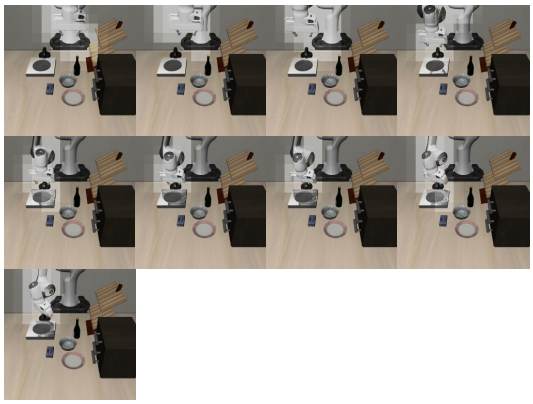


Figure 13: Dynamics-branch slot-to-visual attention on *turn_on_the_stove*. The high-response region shifts with the evolving end-effector–knob interaction, consistent with sensitivity to motion-dependent state change.

C.4.4 Factor-Wise Functional Intervention

We test the functional role of structured control factors in the Stage-III continuous policy on the real-world Fruit Collecting task. After the backbone forward pass, we replace the final hidden states of a selected factor group with its training-set mean before continuous action prediction. The mean is computed separately for each query position and factor type.

Fruit Collecting requires repeated pick-and-place cycles and provides a setting for comparing group-specific interventions. We replace each factor separately and all factors jointly, retaining action readout states and DINOv2 features. Table 16 reports successful trials and variation across evaluation batches using the same trained checkpoint.



Figure 14: Spatial-branch slot-to-visual attention on *turn_on_the_stove*. Attention remains concentrated around the target knob and its geometric neighborhood, consistent with a role in target localization and local geometric grounding.

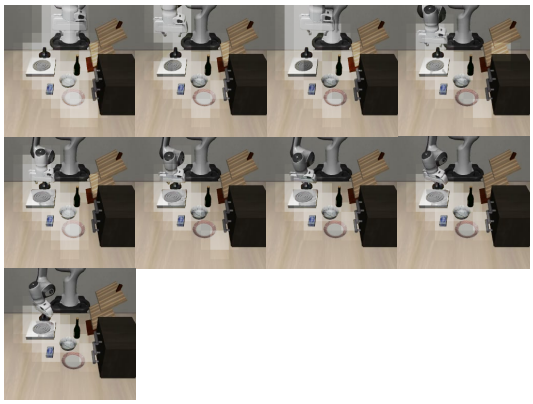


Figure 15: Subtask-branch slot-to-visual attention on *turn_on_the_stove*. Attention is broader and temporally more stable, covering the stove and surrounding semantic context, consistent with stage-level task progression modeling.

Table 16: Factor-wise functional intervention on the real-world Fruit Collecting task. Each evaluation batch reports successful rollouts out of 30 under the same Stage-III checkpoint. SC is mean ± sample SD over three independent evaluation batches per condition.

Variant	Fruit Collecting ID				Fruit Collecting OOD			
	Batch 1	Batch 2	Batch 3	SC (%)	Batch 1	Batch 2	Batch 3	SC (%)
Full SCULPT-VLA	24/30	25/30	23/30	80.0±3.3	18/30	17/30	19/30	60.0±3.3
Replace subtask factor	16/30	15/30	18/30	54.4±5.1	8/30	12/30	6/30	28.9±10.2
Replace spatial factor	22/30	23/30	21/30	73.3±3.3	15/30	13/30	17/30	50.0±6.7
Replace dynamics factor	20/30	21/30	18/30	65.6±5.1	12/30	15/30	11/30	42.2±6.9
Replace all factors	5/30	3/30	6/30	15.6±5.1	2/30	0/30	3/30	5.6±5.1

Task-progression replacement causes the largest drop among single-group interventions, followed by dynamics and spatial replacement. Replacing all three groups leaves low success despite retaining action readout states and DINOv2 features, providing behavioral evidence that the learned structured control state remains functionally involved after direct perceptual access is restored.

Mean replacement alters both factor content and the conditioning distribution, so the measured changes reflect the full intervention.

D Additional Rollouts

D.1 Additional Simulation Visualizations

Figs. 16–21 present rollout visualizations on LIBERO (Liu et al., 2023) and SimplerEnv-WidowX (Li et al., 2024) covering articulated interaction, multi-stage compositional tasks, and fine-grained placement.

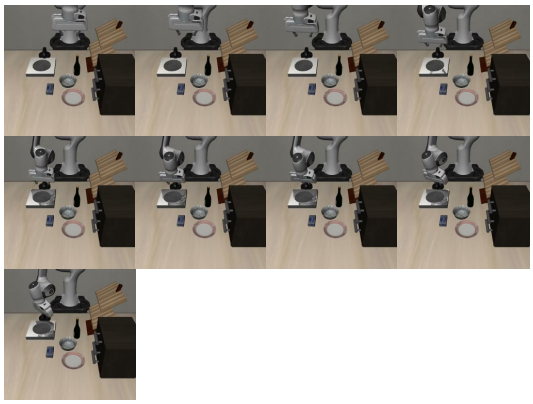


Figure 16: LIBERO *turn_on_the_stove*: approach, end-effector alignment with the knob, and contact-sensitive knob turning.

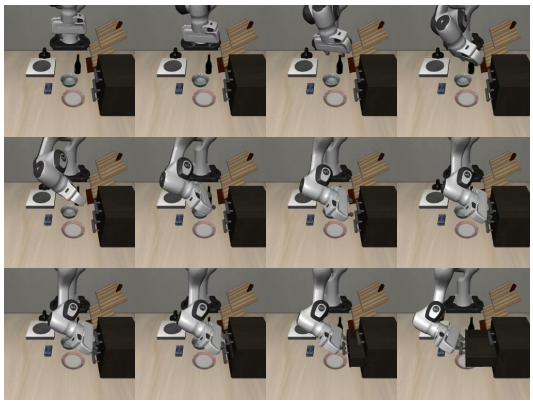


Figure 17: LIBERO *open_the_middle_drawer_of_the_cabinet*: handle approach and sustained drawer pulling while maintaining contact with the handle.

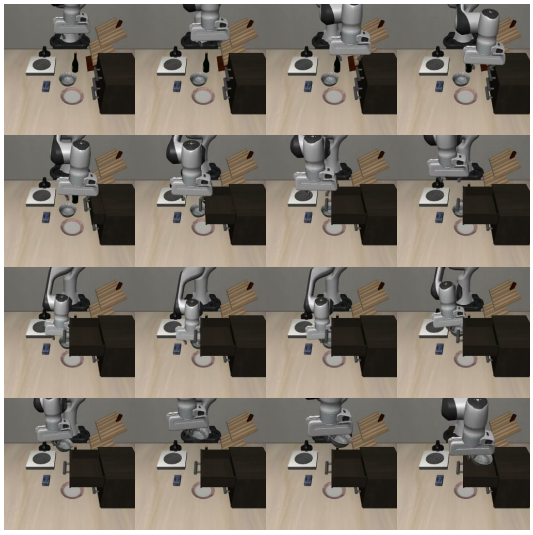


Figure 18: LIBERO *open_the_top_drawer_and_put_the_bowl_inside*: a two-phase task (drawer opening, bowl transport and placement) under changing scene geometry.

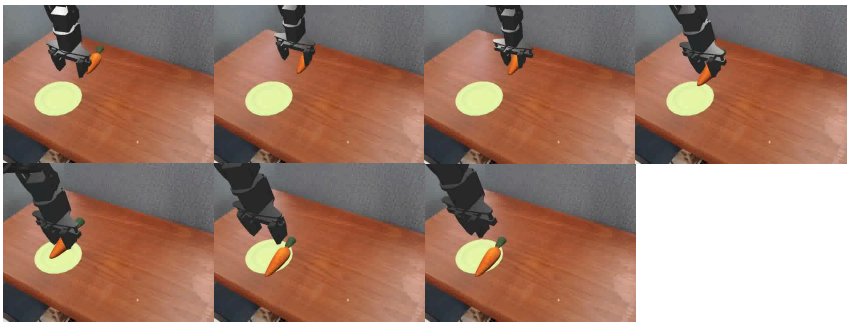


Figure 19: SimplerEnv-WidowX *carrot*: stable grasp with consistent target alignment during transport.

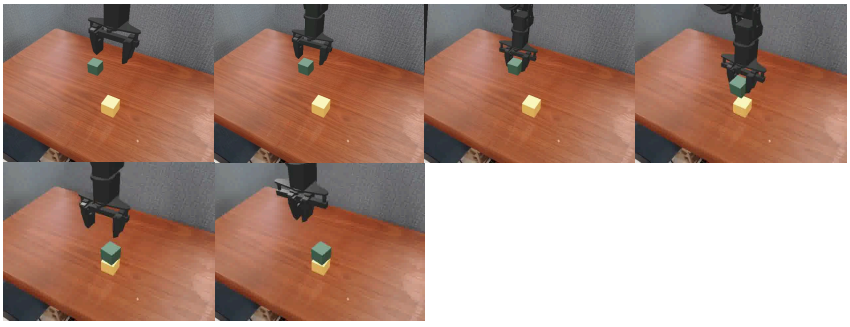


Figure 20: SimplerEnv-WidowX *stack*: relative-pose alignment prior to final placement, illustrating spatial grounding in contact-sensitive stacking.

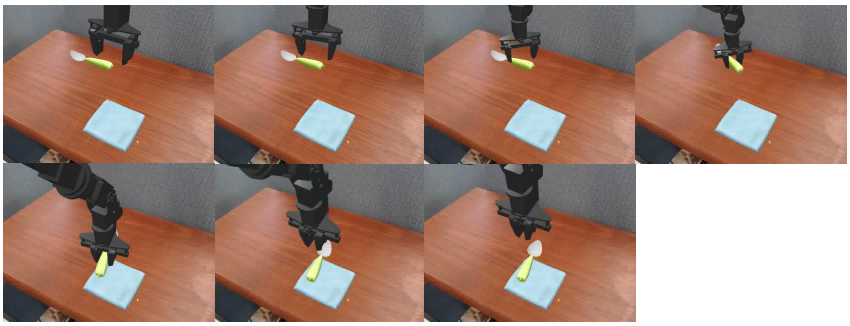


Figure 21: SimplerEnv-WidowX *spoon*: stable grasp and placement of an elongated tool-shaped object under shape-dependent pose constraints.

D.2 Additional Real-World Qualitative Results

Figs. 22–25 show third-person views of representative executions of the four tasks. Quantitative success and open-loop results are in Appendix B.6.

Failure examples. Figure 3 includes Cloth Folding failures involving misalignment and excessive force. Additional failure categories described in the evaluation include premature release and incomplete folding.



Figure 22: Representative **Fruit Collecting** rollout showing repeated grasp–transport–release cycles as the basket fills.

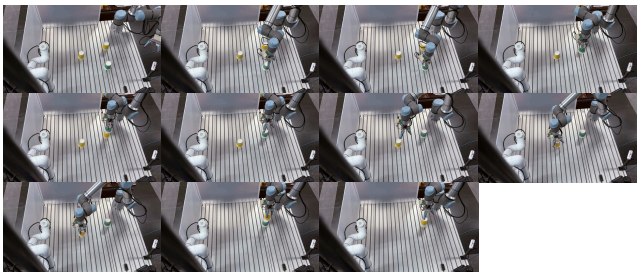


Figure 23: Representative **Paper Cup Stacking** rollout showing cup lifting, alignment, and placement.

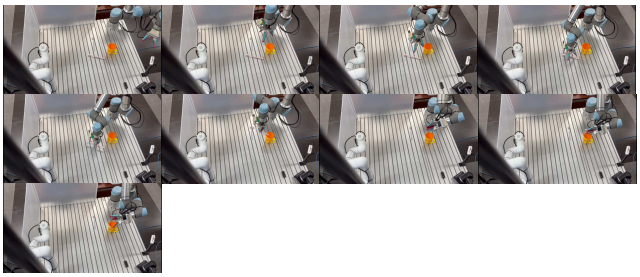


Figure 24: Representative **Toothbrush Insertion** rollout showing grasp, transport, pose adjustment, and insertion.

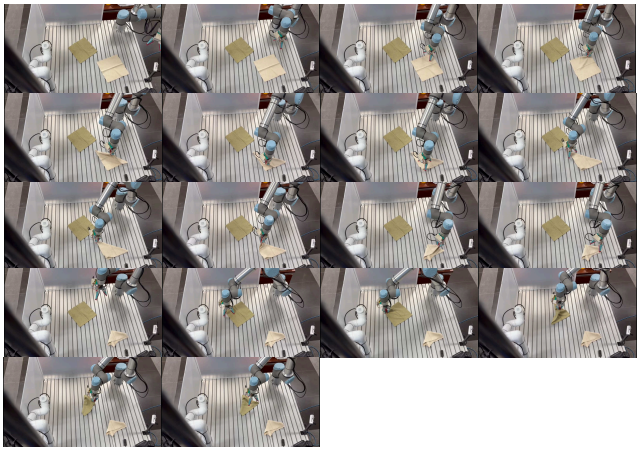


Figure 25: Representative **Cloth Folding** rollout showing pulling and folding as the cloth configuration changes.

E Design Scope and Limitations

SCULPT-VLA uses a fixed decomposition into task-progression, scene-dynamics, and spatial-grounding factors, whose roles are established by training-time teachers.

Physical evaluation covers four tabletop tasks under task-specific ID and OOD conditions on one UR5e platform.