LUTE: FOLDING VISUAL HIERARCHIES AROUND ACTION FOR WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World–action models generate actions and predict the resulting future world states. Existing methods reduce future generation or reuse predictive features to make control more efficient. These strategies can still leave dense visual processing on the action path, while downsampling prediction targets removes spatial detail needed for world estimation. We introduce Lute (distinct strings, one shared resonant body), where folding visual hierarchies around action formation keeps action readout compact while preserving spatial detail for future prediction. Computation proceeds from visual analysis to action generation and then future prediction. Hierarchical processing progressively compresses visual features, with action tokens introduced only at the most compact stage. A paired analysis and synthesis transform retains complementary spatial information that bypasses action formation and supports subsequent prediction conditioned on action representations. All stages use the same block design, with roles defined by spatial scale and information access. Joint flow learning lets both objectives shape the shared hierarchy. When only actions are required, inference omits the subsequent prediction stages. Experiments show competitive control and improved tradeoffs between scene robustness and memory cost over efficient WAM baselines, while retaining structured future predictions.

1 INTRODUCTION

Robot manipulation requires both action generation and an understanding of how actions change the environment. Visuomotor policies map current observations to robot actions (Chi et al., 2023; Kim et al., 2025b; Black et al., 2025), while predictive world models estimate how visual scenes evolve over time (Zhou et al., 2025; Assran et al., 2025). World–action models unify these two capabilities by jointly generating actions and estimating the future world states that follow from them. Their central challenge is therefore to model decision-making and action-conditioned world dynamics within a single system (Zhao et al., 2026; Li et al., 2026b).

Existing world–action models mainly organize this combination in two ways. Joint-generation methods place action and visual futures in a shared generative process, either learning them together or directly using prediction during control (Cen et al., 2025; Kim et al., 2026; Ye et al., 2026; Li et al., 2026a). Representation-access methods instead reduce explicit future generation and expose predictive information to the action process through learned representations, distillation, reuse, or sparse interaction (Yuan et al., 2026b; Fang et al., 2026; Zeng et al., 2026; Zhao et al., 2026; Zhang et al., 2026; Yuan et al., 2026a). Figure 1 illustrates the taxonomy, with cached features representing one form of representation access. One keeps prediction in the execution loop; the other compresses, removes, or reuses it. Although different, the two lines optimize the same interface: how prediction is delivered to action. However, neither line directly addresses how the visual computation should be organized when action formation and future prediction share a single backbone.

This unresolved division affects both inference cost and predictive learning. Removing future generation or caching predictive features reduces generation or interaction costs but leaves dense visual encoding on the action path (Zeng et al., 2026; Zhao et al., 2026). Downsampling future supervision lowers training cost but removes spatial detail from the targets, while action inference still processes original-resolution observations (Li et al., 2026b). *These limitations motivate a common question. How should shared visual computation be organized to improve and balance control performance, prediction quality, and inference efficiency?*

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.

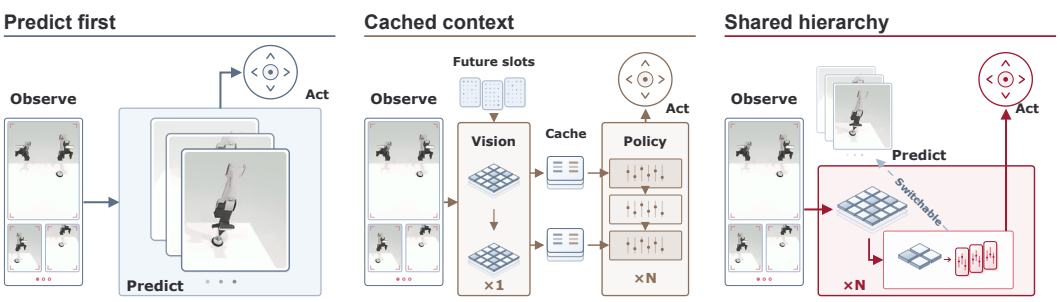

Figure 1: Contrast predict-first, cached-context, and shared-hierarchy organizations. Lute adopts a shared hierarchy with differentiated action and prediction roles.

In response to this question, we introduce Lute, which folds a shared visual hierarchy around action formation. Brain-inspired functional specialization and U-Net-style multiscale processing guide this organization (Felleman & Van Essen, 1991; Ronneberger et al., 2015). All stages use the same DiT block design with different spatial scales and information access. Analysis transforms progressively form compact visual summaries while retaining complementary spatial detail. Action tokens enter only at the most compact scale; subsequent synthesis stages combine action representations with retained details for future prediction. Joint flow matching trains both tasks within this shared backbone, without a separate policy network. When only actions are required, inference omits the subsequent prediction stages. Our key contributions are as follow:

- Organize action formation within a Slow–Mid–Fast hierarchy, differentiating stages through spatial resolution and information access and confining visual–action interaction to the Fast stage.
- Introduce paired analysis and synthesis operators R/W that separate compact summaries from spatial details and reuse the retained details for action-conditioned visual prediction.
- Establish a triangular dependency between the jointly learned action and visual flow fields, enabling predictive supervision of action representations and separate execution of the action flow.

Experiments on LIBERO, LIBERO-Plus, and RoboTwin 2.0 evaluate standard manipulation, visual robustness, and bimanual manipulation. Specifically, Lute matches speed-focused baselines' action latency while using 40% less peak GPU memory and achieving $3\% \sim 6\%$ higher success. Against lightweight baselines, Lute improves efficiency, achieves over 10% higher success, and produces substantially better future visual predictions. Extensive controlled comparisons further validate the roles of predictive supervision, the shared backbone, and spatial differentiation.

2 Related Work

From visual representations to robot actions. Vision–language–action policies connect pretrained visual representations to robot control through action tokenization, diffusion, or continuous flow matching (Kim et al., 2025b; Team et al., 2024; Black et al., 2025). Spatial grounding, latent actions, and efficient fine-tuning further improve the use of visual features for control (Qu et al., 2025; Bu et al., 2025; Kim et al., 2025a). These methods mainly focus on how visual representations are converted into actions. In contrast, world–action models additionally require the shared representation to support prediction of action consequences.

Visual prediction during learning and execution. World–action models couple action generation with predicted visual futures through autoregressive or joint generative objectives (Cen et al., 2025; Kim et al., 2026; Ye et al., 2026). Related approaches study causal world modeling for robot control (Li et al., 2026a), while feature-space world models predict semantic representations for planning (Zhou et al., 2025; Assran et al., 2025). Fast-WAM removes explicit test-time imagination while retaining video-based training (Yuan et al., 2026b). Faster-WAM uses sparse interaction and feature reuse to retain future-conditioned information (Zhao et al., 2026). Light-WAM instead pools visual features and downsamples future supervision to reduce computational cost (Li et al., 2026b). Together, these methods advance joint action–world modeling through different choices about visual prediction during training and inference.

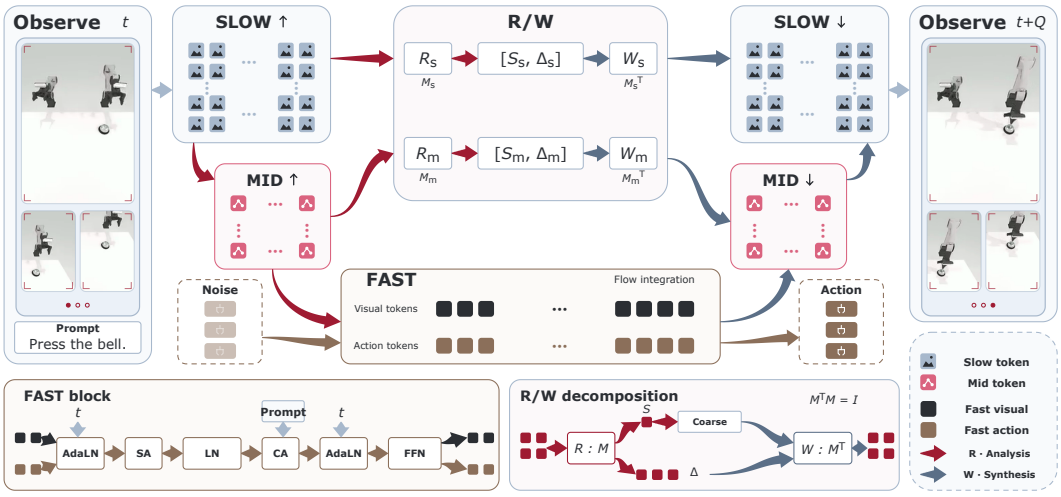

Figure 2: Lute uses one hierarchical backbone for action and world prediction. Spatial analysis R passes compact summaries toward action formation and retains detail coefficients. After action formation, the return path combines its representation with retained detail through transpose-tied synthesis W to predict future visual features. The action path ends before the predictive return path. Here, Q is the number of actions executed before replanning at $t + Q$.

3 Method

Lute folds a shared visual hierarchy around action formation (Fig. 2). Homogeneous multiscale DiT blocks support compact action readout followed by detail-preserving visual prediction. Sec. 3.1 defines the Slow, Mid, and Fast stages and their information flow. Sec. 3.2 introduces orthogonal R/W analysis and synthesis, compressing visual summaries while retaining complementary spatial detail. Sec. 3.3 describes joint flow matching, with actions read out at Fast before their representations and retained detail enter the predictive return. Sec. 3.4 derives action-only inference from this ordering, stopping before visual synthesis while preserving the jointly trained action computation.

3.1 Formulation and Information Flow

At physical time t, the agent receives observations $I_t = \{I_t^{(v)}\}_{v=1}^{V}$ from V synchronized cameras, robot state s_t, and instruction ℓ. It generates an action chunk A_t of horizon H_a and replans after executing its first Q actions. A frozen framewise encoder E produces visual grids $Z_t = E(I_t)$, and $C_t = \{Z_r : r \in \mathcal{H}_t,\ r \le t\}$ is the causal history. The index set $\mathcal{H}_t$ selects at most L observed frames. Camera identity uses a learned embedding, while physical time enters the positional encoding. For each source frame r used in training, the clean visual target is $Z^{+*,(r)} = E(I_{r+Q})$, paired with an action chunk beginning at r. We denote flow time by $u \in [0, 1]$.

Slow, Mid, and Fast (subscripts s, m, f) denote the dense, intermediate, and compact spatial stages. Slow uses within-frame visual attention; Mid analysis alternates spatial and causal temporal attention. Fast introduces action tokens, which read their source-frame visual representation and their own action chunk. Visual self-attention stays within the visual tokens. The return blocks process future visual queries using retained detail and the Fast action representation.

All stages use the same DiT block design, with their roles determined by spatial resolution and information access. For input tokens X_b at block b, an attention mask M_b controls token access and $\chi_b \in \{0, 1\}$ selects language cross-attention:

$$\begin{aligned} H_b &= X_b + \mathrm{SA}_b(X_b; M_b, u), \\ \widetilde{H}_b &= H_b + \chi_b\, \mathrm{CA}_b(\mathcal{N}_b(H_b), \ell), \qquad X_{b+1} = \widetilde{H}_b + \mathrm{FFN}_b(\widetilde{H}_b; u). \end{aligned} \tag{1}$$

Here, SA_b and FFN_b are flow-conditioned self-attention and feed-forward modules, CA_b attends to instruction tokens ℓ, and $\mathcal{N}_b$ normalizes its queries.

3.2 DETAIL-PRESERVING SPATIAL HIERARCHY

We adopt a multiscale analysis–synthesis layout (Ronneberger et al., 2015; Tian et al., 2024), passing compact summaries toward action formation and retaining complementary detail for prediction. At reduction $j \in \{s, m\}$, analysis R_j applies a learned orthogonal transform to each channel's non-overlapping 2×2 spatial group $x \in \mathbb{R}^4$:

$$\begin{aligned} [S_j, \Delta_j]^\top = M_j x, \qquad M_j = G(\theta_j) H_{\mathrm{Haar}}, \\ W_j(S_j, \Delta_j) = M_j^\top [S_j, \Delta_j]^\top, \qquad M_j^\top M_j = I. \end{aligned} \tag{2}$$

Here, H_{Haar} is the Haar basis. The rotation $G(\theta_j)$ has learned angles θ_j initialized to zero, giving $G(\theta_j) = I$. One coefficient forms the summary S_j; the other three form the retained detail Δ_j. The transform operates independently within each channel, camera view, and physical frame. Synthesis W_j reconstructs the local input from its summary and detail using $M_j^\top$ (App. C.2).

Only the summary continues to the next analysis stage. With P_Z projecting visual features to the backbone width and F_s, F_m denoting the Slow and Mid analysis blocks, the two reductions are

$$(X_m, \Delta_s) = R_s(F_s(P_Z C_t)), \qquad (X_f, \Delta_m) = R_m(F_m(X_m)). \tag{3}$$

Each reduction quarters the number of spatial tokens, leaving Fast with one sixteenth of the original tokens per frame and camera. The detail coefficients Δ_s and Δ_m bypass Fast. The predictive return pathway combines them with updated summaries to predict future visual features.

3.3 JOINT ACTION AND PREDICTION LEARNING

We jointly train action formation and visual prediction with conditional flow matching (Lipman et al., 2023). For source frame r, the targets share a sampled flow time u and use independent Gaussian noises $\epsilon_A^{(r)}$ and $\epsilon_Z^{(r)}$. The clean targets $A^{*,(r)}$ and $Z^{+*,(r)}$ define

$$A_u^{(r)} = (1-u)\epsilon_A^{(r)} + u A^{*,(r)}, \qquad Z_u^{+,(r)} = (1-u)\epsilon_Z^{(r)} + u Z^{+*,(r)}, \tag{4}$$

with target velocities $A^{*,(r)} - \epsilon_A^{(r)}$ and $Z^{+*,(r)} - \epsilon_Z^{(r)}$. Projections P_A, P_s map actions and robot state to the backbone width. Given source-frame features $X_f^{(r)}$ and state s_r, the Fast blocks F_f produce hidden states $H_r^{\mathrm{vis}}, H_r^{\mathrm{act}}$; head h_A reads out the action velocity:

$$(H_r^{\mathrm{vis}}, H_r^{\mathrm{act}}) = F_f(X_f^{(r)}, P_A A_u^{(r)} + P_s s_r; \ell, u), \qquad v_A^{(r)} = h_A(H_r^{\mathrm{act}}). \tag{5}$$

At each u, the predictive return pathway uses the current action representation. For source frame r, the Qth action-token hidden state $H_{r,Q}^{\mathrm{act}}$ conditions prediction at $r+Q$. Only after action readout, projection and summary reduction map $Z_u^{+,(r)}$ to inputs $(N_s^{(r)}, N_m^{(r)}, N_f^{(r)})$ at the three spatial scales (App. C.3). The return representations Y_f, Y_m, Y_s and visual velocity v_Z are

$$\begin{aligned} Y_f^{(r)} &= H_r^{\mathrm{vis}} + P_{aw} H_{r,Q}^{\mathrm{act}} + N_f^{(r)}, \qquad Y_m^{(r)} = G_m\Big(W_m(Y_f^{(r)}, \Delta_m^{(r)}) + N_m^{(r)}; \ell, u\Big), \\ Y_s^{(r)} &= G_s\Big(W_s(Y_m^{(r)}, \Delta_s^{(r)}) + N_s^{(r)}; u\Big), \qquad v_Z^{(r)} = h_Z(Y_s^{(r)}). \end{aligned} \tag{6}$$

The projection P_{aw} maps action features to visual channels and broadcasts them over spatial positions. Transpose-tied synthesis W_m, W_s combines updated summaries with source-observation detail; G_m, G_s are the Mid and Slow return blocks, and h_Z is the visual-velocity head.

The two flow fields are optimized with equally weighted, separately normalized objectives:

$$\mathcal{L} = \mathbb{E}\left[\left\langle \left(v_A^{(r)} - (A^{*,(r)} - \epsilon_A^{(r)})\right)^2 \right\rangle_{M_A} + \left\langle \left(v_Z^{(r)} - (Z^{+*,(r)} - \epsilon_Z^{(r)})\right)^2 \right\rangle_{M_Z}\right]. \tag{7}$$

Here, M_A, M_Z select valid targets; $\langle e^2 \rangle_M$ averages elementwise squared errors over valid targets and feature dimensions. Both losses train the shared analysis pathway. The visual loss also propagates through H^{act} in Eq. 6, so prediction directly supervises the representation used to form actions.

3.4 ACTION-ONLY INFERENCE

Action formation depends on observed context and the current action state A_u. Prediction reads the action representation. Writing the action and visual fields as f_θ and g_θ, with model parameters θ, gives the triangular dependency:

$$\frac{dA_u}{du} = f_\theta(A_u, u; C_t, s_t, \ell), \qquad \frac{dZ_u^+}{du} = g_\theta(Z_u^+, A_u, u; C_t, s_t, \ell), \quad \frac{\partial v_A}{\partial Z_u^+} = 0. \tag{8}$$

Omitting the predictive return pathway leaves the action field unchanged. For K integration steps with nodes $0 = u_0 < \cdots < u_K = 1$, action inference evaluates only

$$A_{u_{i+1}} = A_{u_i} + (u_{i+1} - u_i) f_\theta(A_{u_i}, u_i; C_t, s_t, \ell), \qquad A_{u_0} = \epsilon_A. \tag{9}$$

Each step runs the analysis pathway and Fast action blocks, ending at the action readout. Future-latent projections, synthesis operators, and return blocks are skipped. After executing Q actions, the agent updates its history with Z_{t+Q}, keeps at most L observations, and replans.

4 EXPERIMENTS

We evaluate Lute as a scene-robust WAM with practical deployment costs. Sec. 4.1 defines how these properties are evaluated. Sec. 4.2 tests whether robustness and world estimation improve over existing methods. Sec. 4.3 examines which design choices support control and efficient execution. Sec. 4.4 examines the deployment trade-off and how the hierarchy responds during action generation.

4.1 EXPERIMENTAL SETUP

Benchmarks and evaluation. LIBERO covers spatial, object, goal, and long-horizon manipulation (Liu et al., 2023); we report the equally weighted average across its four suites. LIBERO-Plus tests visual and environmental perturbations using the official aggregate (Fei et al., 2025), with category results in App. A. RoboTwin evaluates bimanual manipulation under separate Easy and Hard protocols (Chen et al., 2025). We report first-attempt success for two model scales and an action-only training variant with matched action capacity. Action latency and peak memory quantify deployment cost. The trade-off in Fig. 5 measures action-only inference; predictive capability is evaluated separately against observed futures.

Implementation. We initialize from Wan and train with AdamW for 10 epochs on LIBERO and 5 on RoboTwin, using eight NVIDIA RTX PRO 6000D GPUs, half precision, and ZeRO Stage 2. Each GPU processes 16 samples, giving a global batch of 128. The visual encoder is frozen, and action and visual flow losses are separately normalized with equal weights. App. C provides architecture and optimization details for both model scales and the benchmark-specific interfaces.

Table 1: Canonical LIBERO success (%). Params (w/o text encoder and autoencoders) reports total / active parameters when they differ; otherwise, a single value is shown. $\sim$ denotes an architecture-derived estimate. Pretrain denotes the initialization checkpoint. Lute averages the four suites equally.

Method	Params	Pretrain	Spatial	Object	Goal	Long	Average
Octo (Team et al., 2024)	93M	Scratch (OXE)	78.9	85.7	84.6	51.1	75.1
OpenVLA (Kim et al., 2025b)	7B	Prismatic-7B	84.7	88.4	79.2	53.7	76.5
π_0 (Black et al., 2025)	3.3B	PaliGemma-3B	96.8	98.8	95.8	85.2	94.2
OpenVLA-OFT (Kim et al., 2025a)	7B	OpenVLA-7B	97.6	98.4	97.9	94.5	97.1
UniVLA (Bu et al., 2025)	7B	Prismatic-7B	96.5	96.8	95.6	92.0	95.2
OpenVLA-OFT + RIPT (Tan et al., 2025)	7B	OpenVLA-OFT	98.6	98.6	99.0	93.8	97.5
Fast-WAM (Yuan et al., 2026b)	6B	Wan2.2-TI2V-5B	98.2	100.0	97.0	95.2	97.6
PFD (Fang et al., 2026)	$\sim$6B	Fast-WAM	98.6	99.2	99.2	95.4	98.1
Light-WAM (Li et al., 2026b)	1.99B	Wan2.1-T2V-1.3B	98.2	99.6	97.8	93.0	97.2
Enfold (Zeng et al., 2026)	3B	Cosmos-Predict2.5-2B	97.0	100.0	96.8	97.4	97.8
DreamWAM (Yuan et al., 2026a)	$\sim$6B	Wan2.2-TI2V-5B	99.6	99.8	98.6	97.6	98.9
Faster-WAM (Zhao et al., 2026)	$\sim$6B	Wan2.2-TI2V-5B	99.6	99.8	98.2	98.2	99.0
Lute 1.4B(A0.87B)	1.4B / 0.87B	Wan2.1-T2V-1.3B	97.6	99.0	97.5	93.5	96.9
Lute 5B(A3.05B)	5B / 3.05B	Wan2.2-TI2V-5B	99.1	99.5	98.3	97.5	98.6

Table 2: **Shifted-benchmark results. (a)** Official LIBERO-Plus Total; full dimensions appear in Tab. 4 (Fei et al., 2025). **(b)** Official RoboTwin 2.0 fifty-task Easy/Hard protocol (Chen et al., 2025; RoboTwin Team, 2026). Daggers mark co-trained policies; published values retain their source protocols. Lute reports first-attempt results. VLA indicates only action training.

(a) LIBERO-Plus

Method family	Total
OpenVLA (Kim et al., 2025b)	15.6
OpenVLA-OFT (Kim et al., 2025a)	69.6
WorldVLA (Cen et al., 2025)	25.0
UniVLA (Bu et al., 2025)	43.9
π_0 (Black et al., 2025)	53.6
RIPT-VLA (Tan et al., 2025)	68.4
LightWAM (Li et al., 2026b)	54.4
Faster-WAM (Zhao et al., 2026)	73.6
Lute VLA(0.87B)	62.1
Lute 1.4B(A0.87B)	65.5
Lute 5.0B(A3.05B)	76.8

(b) RoboTwin 2.0

Method	Easy	Hard
ACT (Zhao et al., 2023)	29.74	1.74
Diffusion Policy (Chi et al., 2023)	28.04	0.64
DP3 (Ze et al., 2024)	55.24	4.96
RDT-1B (Liu et al., 2024)	34.50	13.72
π_0 (Black et al., 2025)	46.42	16.34
X-VLA† (Zheng et al., 2025)	68.00	20.90
$\pi^\dagger_{0.5}$ (Physical Intelligence et al., 2025)	70.70	46.00
Fast-WAM† (Yuan et al., 2026b)	77.80	1.90
Lute VLA(0.87B)	41.10	30.20
Lute 1.4B(A0.87B)	43.50	38.60
Lute 5.0B(A3.05B)	79.20	52.50

4.2 Main comparisons

Standard manipulation. Tab. 1 compares canonical LIBERO success across four suites. Strong VLAs already approach saturation on Spatial, Object, and Goal; leading WAMs show clearer improvements on Long-Horizon. The smaller Lute performs close to the lightweight WAM overall and slightly better on Long-Horizon, with fewer active parameters. Scaling brings it close to the leading larger WAMs, with the clearest gain on extended manipulation. Compact action formation thus preserves competitive control while separating active action capacity from total predictive capacity. The smaller scale offers a compact policy with predictive training, while the larger scale closes much of the remaining long-horizon gap to the strongest WAMs. This establishes the control foundation for evaluating robustness and deployment cost together.

Scene robustness. Reading Tab. 1 and Tab. 2(a) together reveals substantial ranking changes under perturbations. Similar canonical scores conceal different responses to changed scenes. The lightweight WAM falls behind adapted VLA baselines and the speed-focused larger WAM on LIBERO-Plus. Robustness therefore provides a clearer distinction among methods than success on nearly saturated tasks. The smaller model gains more under perturbations despite similar familiar-scene accuracy, showing greater reliability under changed observations.

Both Lute scales improve their relative position. The smaller model moves from comparable canonical performance to about a 20% relative advantage over the lightweight baseline. The larger model leads the reported LIBERO-Plus comparison, exceeding the speed-focused larger baseline by about 3%. Together with Fig. 5, these results combine stronger scene robustness than lightweight alternatives with competitive latency and lower memory than larger WAMs. The smaller model offers robustness under tighter memory budgets, while scaling strengthens both perturbed-scene reliability and demanding bimanual control within the same architecture.

Challenging bimanual control. Tab. 2(b) shows that strong RoboTwin Easy performance can accompany a sharp decline on Hard. Baseline rankings change as coordinated manipulation becomes more demanding. The smaller Lute is more competitive on Hard than its Easy ranking suggests; scaling improves both settings and yields the highest reported success in each column. The gains across both difficulty levels show that the larger model improves average success while also strengthening performance on demanding interactions. Across the two tables, additional capacity brings clearer benefits under scene changes and difficult interactions than on familiar, nearly saturated tasks.

Contribution of predictive learning. At matched action capacity, joint prediction improves LIBERO-Plus and both RoboTwin settings, with the largest relative gain on Hard. These comparisons separate the benefit of predictive supervision from the gains obtained through scaling. The benefit spans changed observations and difficult bimanual interaction, while action inference still stops before visual synthesis. Predictive learning therefore strengthens control without requiring future-image generation during execution, supporting robust control at low inference cost.

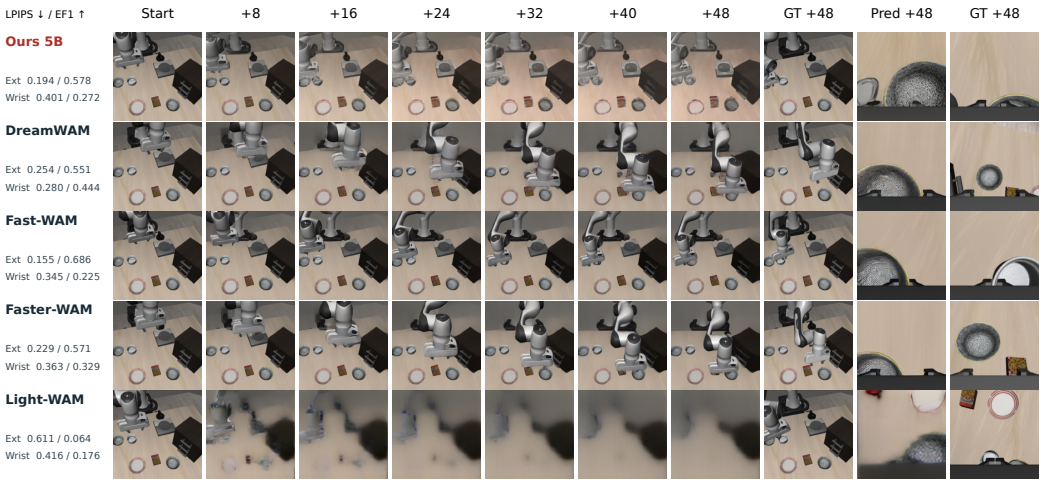

Figure 3: Recursive prediction under camera perturbation. Ground truth follows each method's own actions. Ext/Wrist report view-specific LPIPS and edge F1. DreamWAM uses its joint configuration.

Acting as a world model. Fig. 3 shows a clear separation in the selected camera-perturbation episode. Lute retains the tabletop arrangement, while the lightweight baseline progressively loses object contours. After two eight-step warm-up rounds, each method runs six eight-action rounds using predicted camera views as subsequent inputs, with measured robot states available. Evaluation uses observations following each method's actions. Objects remain recognizable as predictions replace real camera observations.

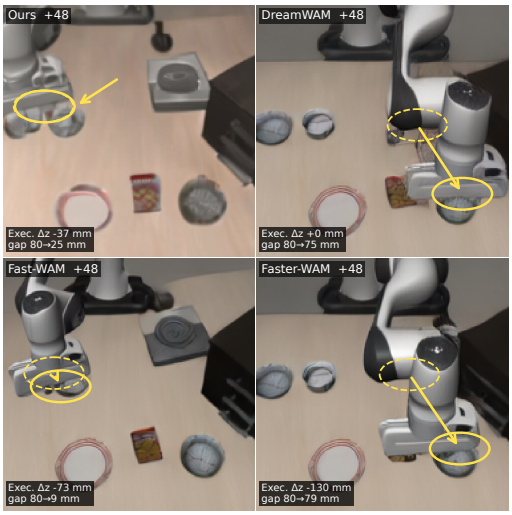

Figure 4: Gripper regions at $+24$ (dashed) and $+48$ (solid), overlaid on final predictions. Arrows indicate displacement or overlap; insets show measured motion and finger separation.

Perceptual and structural agreement. Learned perceptual image patch similarity (LPIPS, lower is better) measures visual-feature discrepancy; edge F1 (EF1, higher is better) measures boundary agreement. External (Ext) and wrist views are scored separately. The external-view advantage over the lightweight baseline appears in both measures, matching the visible contrast between recognizable objects and diffuse regions. The smaller-model examples also report mean absolute pixel error (MAE, lower is better). The baseline retains modest pixel error despite near-zero external-view edge agreement. Broad appearance survives while object boundaries disappear.

World estimation at smaller scale. App. Fig. 7 shows the smaller Lute preserving plates, containers, and scene boundaries across Spatial, Goal, and Object examples. The lightweight baseline blurs these structures as prediction proceeds. Stronger external-view edge agreement and lower pixel error in both views accompany the smaller model's visible advantage. Together with its perturbed-scene success and action-only memory results, these examples show that a compact policy retains spatially informative world estimates. The hierarchy supports robust control at low execution cost while preserving predictive capability in the same backbone.

Prediction and executed motion. Fig. 4 exposes distinct prediction–execution relationships. DreamWAM and Faster-WAM show large predicted gripper shifts, yet their measured descents differ sharply and both retain open fingers. Fast-WAM and Lute execute descent and closure with smaller shifts between the marked predicted regions. Visible gripper displacement therefore tracks a different aspect of interaction from vertical motion and finger closure. These examples make action-conditioned consistency a separate evaluation target alongside scene preservation, linking predicted visual changes to the interaction actually executed.

Table 3: **Controlled ablations.** Colored subscripts show changes from the omitted reference using pretrained S_F, 20/60/20 Slow/Mid/Fast, and R/W with residual, scoring 65.1 on LIBERO-Plus and 93.5 on LIBERO Long. S_F denotes the three Slow blocks nearest Fast; Δ denotes the retained residual. In (c), Action/Joint distinguish action-only inference from joint action and visual prediction.

(a) Language Injection

CA placement	Plus
All pretrained	$65.8_{+0.7}$
Fast only	$64.3_{-0.8}$
Mid only	$64.8_{-0.3}$
S_F, scratch	$63.5_{-1.6}$

(b) Downsample.

Op.	Δ	Plus
Pool	–	$62.9_{-2.2}$
Pool	+	$63.2_{-1.9}$
Conv	+	$65.4_{+0.3}$
R/W	–	$64.8_{-0.3}$

(c) Inference mode

Model	Mode	Peak GiB	p50 ms
1.4B(A0.87B)	Action	1.66	124.6
	Joint	3.89	192.6
5.0B(A3.05B)	Action	5.75	250.6
	Joint	10.55	361.0

(d) Axis allocation

$S/M/F$	Plus	Long
40/40/20	$62.7_{-2.4}$	$91.0_{-2.5}$
20/40/40	$65.6_{+0.5}$	$93.2_{-0.3}$

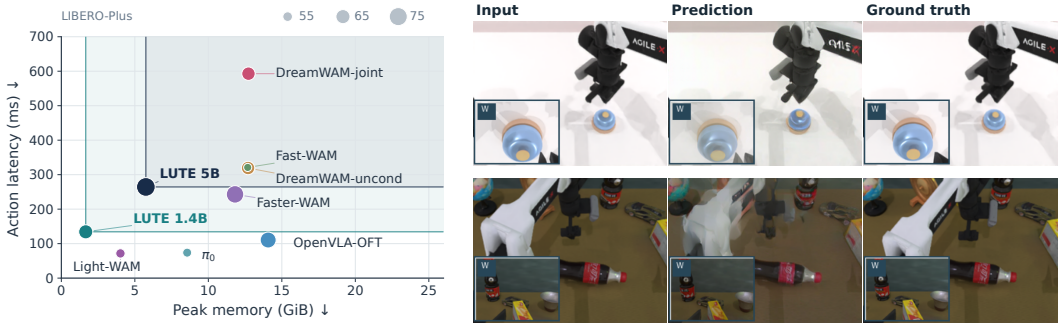

Figure 5: Efficiency tradeoff and visual prediction. (Left) Action-only inference latency and peak memory; bubble size indicates LIBERO-Plus success. (Right) Multiview observations, visual predictions, and matched ground truth for RoboTwin bell and bottle manipulation.

4.3 Ablation Studies

We use Lute 1.4B for controlled ablations with the same training recipe. Unless noted otherwise, variants are evaluated on LIBERO-Plus (Seed = 0). Axis allocation preserves total depth and symmetric Slow/Mid branches. The inference-mode study measures latency and memory at both model scales.

Tab. 3(a) places the selected Slow conditioning between Fast/Mid-only injection and conditioning every block. Early language access captures most of the all-block result. Tab. 3(d) shows that more dense processing is not automatically beneficial. Moving depth from Mid to Slow reduces both Plus and Long success; moving it to Fast improves Plus but slightly lowers Long. The fixed-depth comparison favors intermediate processing for Long and additional compact processing for Plus, making stage allocation a task-dependent choice within the shared hierarchy.

Tab. 3(b) separates compression quality from retained detail. Adding detail to pooling recovers little of its deficit, whereas R/W without detail remains close to the reference and learned convolution. Control is thus more sensitive to the tested compression operators than to the presence of a detail branch. Tab. 3(c) separates predictive learning from execution cost. Ending at Fast saves roughly one third of latency and half of peak memory at both scales, using the same jointly trained parameters. Combined with the action-only training comparison, this supports learning from future prediction while deploying the resulting policy through compact action computation.

4.4 Visualization

Deployment cost. Fig. 5 shows how model scale changes the deployment trade-off. The smaller model improves scene robustness over the lightweight baseline while using less memory; the larger model improves robustness over the speed-focused baseline at similar action-only latency and 40% lower memory. Scaling therefore buys stronger control while retaining a memory advantage in both comparisons. The right panel shows recognizable bell and bottle scenes across head and wrist views, with visible gripper-pose differences. The shared model combines compact action execution with spatially informative prediction when the return path is enabled.

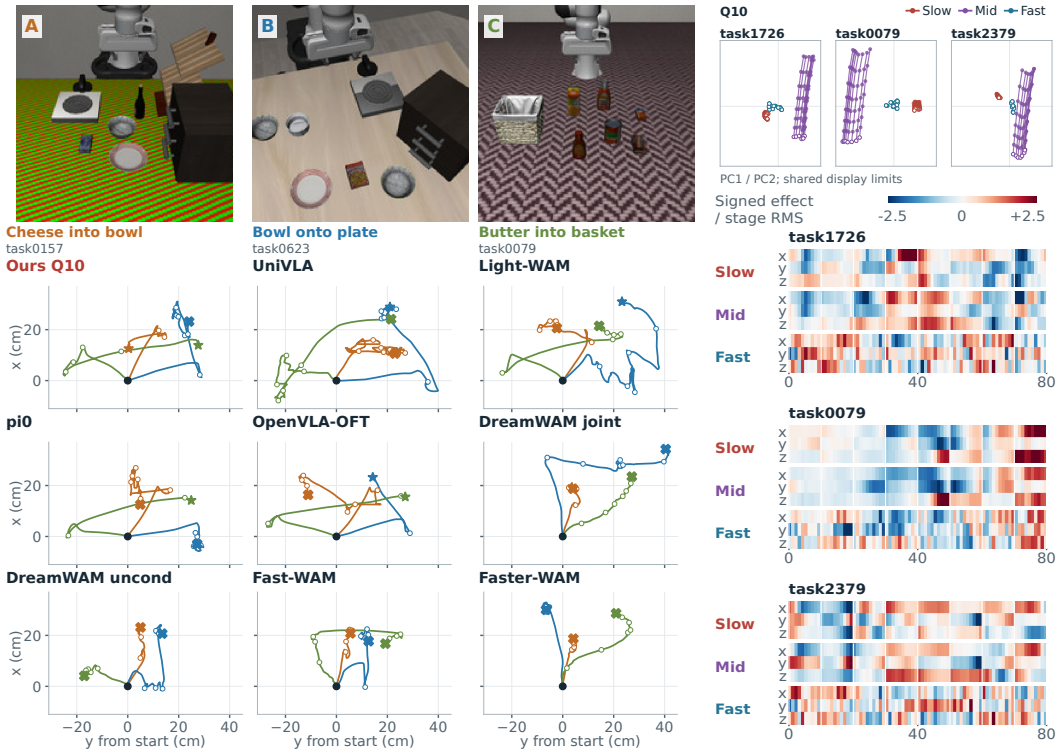

Figure 6: Closed-loop behavior and pathway dynamics. (Left) Task scenes and end-effector trajectories from matched initial conditions; each policy follows its own rollout. Colors identify tasks, circles positions, and stars/crosses success/failure. (Right) Readout projections and translation changes under pathway freezing. Lines connect samples within replans; heatmaps use stage RMS.

Fig. 6 reveals why path compactness alone poorly characterizes scene-robust control. In the matched texture task, Lute succeeds after a lateral reversal, while several shorter approaches terminate in failure. Its other successful path also includes a substantial change of direction; the failed bowl-placement run repeatedly revisits a local region. The distinction is whether successive adjustments advance the interaction or recur near the same position. Across policies, the same scene induces different approach directions and endpoints, while each policy changes its route across tasks. These patterns motivate evaluating closed-loop progress under perturbations alongside aggregate success, capturing the behavior through which robustness is expressed.

The internal probes expose different response structures within the homogeneous hierarchy. Mid traces extended paths in its own projection, while Slow and Fast remain more localized in theirs. Under freezing, Slow and Mid produce broader bands of similarly signed action changes, whereas Fast produces more frequent local sign changes. Visual-stage perturbations therefore tend to shift groups of action outputs together, while perturbing the action readout changes finer output structure. This division matches the visual context processed by Slow and Mid and the action tokens updated by Fast. Spatial placement and token access differentiate the shared block design during action formation, producing distinct patterns of coordinated and local output changes.

5 Conclusion

This paper finds that predictive training improves bimanual control even when execution stops before visual synthesis. Future WAM design should separate shared learning capacity from action-time computation. Retaining spatial detail supports world estimation alongside compact action processing. Information routing and readout location deserve attention alongside model scale. Evaluation should examine scene robustness, action cost, and predictive structure. Canonical success can conceal sensitivity to scene changes, while pixel similarity can conceal object-boundary loss. Lute's decision-making and future-estimation performance does not appear to saturate at the 5B parameter scale, suggesting further gains through scaling. Further discussion appears in App. A.

AI USE STATEMENT

In this work, we used generative AI tools to identify relevant literature, improve manuscript readability, and assist in verifying parts of the experimental code. We did not use generative AI tools to design or conduct experiments, interpret experimental results, or generate the initial manuscript draft. All AI-assisted outputs were reviewed and verified by the authors. We take responsibility for the final content of this work.

ETHICS STATEMENT

Lute did not conduct destructive tests on the real environment, nor did it involve human subjects. Furthermore, it is unable to generate content that is not action-oriented. All the training was conducted on open-source datasets and simulators.

REPRODUCIBILITY STATEMENT

All experiments fixed the random seed. All the codes (except for the weights) can be found in the appendix. After undergoing peer review, the complete training process and weights will be made publicly available.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to act anywhere with task-centric latent actions. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.014.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model, 2025. URL https://arxiv.org/abs/2506.21539.

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, 2025. URL https://arxiv.org/abs/2506.18088.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2023. URL https://arxiv.org/abs/2303.04137.

Pengcheng Fang, Hongli Chen, and Xiaohao Cai. Privileged foresight distillation: Zero-cost future correction for world action models, 2026. URL https://arxiv.org/abs/2604.25859.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: In-depth robustness analysis of vision-language-action models, 2025. URL https://arxiv.org/abs/2510.13626.

Daniel J. Felleman and David C. Van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral Cortex*, 1(1):1–47, 1991. doi: 10.1093/cercor/1.1.1.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos Policy: Fine-tuning video models for visuomotor control and planning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/748becc400a57c0e31cfe6a2e7951467-Abstract-Conference.html.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control, 2026a. URL https://arxiv.org/abs/2601.21998.

Ziang Li, Dongzhou Cheng, Yibin Wang, Shiyue Wang, Xiaoyang Xu, Lingxuan Weng, Juan Wang, and Jiaqi Wang. Light-WAM: Efficient world action models with state-fusion action decoding, 2026b. URL https://arxiv.org/abs/2606.08242.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation, 2024. URL https://arxiv.org/abs/2410.07864.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011.

RoboTwin Team. RoboTwin 2.0 leaderboard, 2026. URL https://robotwin-platform.github.io/leaderboard. Accessed September 9, 2026.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015. doi: 10.1007/978-3-319-24574-4_28.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models, 2025. URL https://arxiv.org/abs/2505.17016.

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.

Yuchuan Tian, Zhijun Tu, Hanting Chen, Jie Hu, Chao Xu, and Yunhe Wang. U-DiTs: Downsample tokens in u-shaped diffusion transformers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 51994–52013, 2024.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, et al. World action models are zero-shot policies, 2026. URL https://arxiv.org/abs/2602.15922.

Shanglin Yuan, Weiheng Zhao, Xin Shi, Haoyi Jiang, Xianda Guo, Liu Liu, Wenyu Liu, Wei Sui, and Xinggang Wang. DreamWAM: Beyond RGB future prediction for world action models, 2026a. URL https://arxiv.org/abs/2608.04996.

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

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable visuomotor policy learning via simple 3D representations, 2024. URL https://arxiv.org/abs/2403.03954.

Weili Zeng, Yitong Xing, Fulong Liu, Chengqun Yang, Antao Xiang, Feng Tian, Jingnan Gao, Jisong Cai, Xin Wang, Xiaomin Wu, Yao Mu, Xiaokang Yang, and Yichao Yan. Enfold: Folding world model imagination into predictive representations for ultra-efficient embodied control, 2026. URL https://arxiv.org/abs/2607.26657.

Chushan Zhang, Jinguang Tong, Xuesong Li, Yikai Wang, and Hongdong Li. Keep the future, drop the rollout: RIFT for world action models, 2026. URL https://arxiv.org/abs/2608.11521.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

Weiheng Zhao, Haoyi Jiang, Xin Shi, Liu Liu, Fan Huang, Zhizhong Su, Wei Sui, and Xinggang Wang. Faster-WAM: Efficient inference-time future conditioning for robust world action models, 2026. URL https://arxiv.org/abs/2608.04404.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025. URL https://arxiv.org/abs/2510.10274.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A Discussion and Limitations

Action success and visual prediction reveal different aspects of representation quality. Removing retained detail changes control success only slightly, while the prediction comparisons reveal substantial differences in object boundaries and scene layout. The hierarchy routes compact summaries through Fast and preserves complementary detail for visual synthesis. Action execution uses the compact path, while world estimation draws on spatial information retained by the shared backbone.

The gains become clearer as evaluation moves beyond canonical success. Similar LIBERO scores separate under scene perturbations, and predictive training benefits hard bimanual tasks more than easy ones. Meanwhile, extra dense processing reduces success in the fixed-depth ablation. These results favor strengthening compact representations through predictive learning and selective computation, then deploying them through the action pathway.

Limitations. Sharing the backbone couples the capacity available for action formation and visual prediction. It provides decision-oriented visual features, while allocating fewer total parameters to joint modeling than larger WAMs (e.g., w/ 1B Action Expert). Fine-grained future geometry remains a source of prediction error. Action generation also relies on the shared visual summary rather than a separate action backbone. This coupling may favor information useful for the immediate action chunk over state information needed across longer interactions, offering a possible explanation for sensitivity to starting conditions and remaining long-horizon errors.

As shown in Tab. 4 and Tab. 1, robot initialization is the weakest LIBERO-Plus category for both scales, and Long is their lowest-scoring canonical suite. Scaling improves both, yet the larger model's strong aggregate robustness coexists with a gap to leading methods on robot initialization and Long. These results identify starting-state variation and sustained task progress as priorities for improving the shared representation. Future work will investigate which visual information decisions require and which future prediction requires, testing how shared and task-specific capacity affect initial-state robustness, long-horizon control, and prediction fidelity.

Table 4: **LIBERO-Plus category breakdown.** Success rates (%) for methods compared in the main paper. Best results are bolded and second-best results are underlined. Aggregate scores retain the reported evaluation protocols.

Method	Camera	Noise	Layout	Light	Texture	Robot Init.	Language	Total
OpenVLA	0.80	15.20	28.50	8.10	34.80	3.50	23.00	15.60
OpenVLA-OFT	56.40	<u>75.80</u>	74.20	88.70	**93.30**	31.90	79.50	69.60
NORA	2.20	12.80	62.10	45.70	58.60	37.00	65.10	39.00
WorldVLA	0.10	10.90	38.00	43.70	17.10	27.90	41.60	25.00
UniVLA	1.80	21.20	31.90	69.00	81.00	46.20	69.60	43.90
π_0	13.80	**79.00**	68.90	85.00	81.40	6.00	58.80	53.60
RIPT-VLA	55.20	73.50	74.20	88.40	<u>91.60</u>	31.20	77.60	68.40
Light-WAM	<u>65.83</u>	61.48	61.82	72.84	24.86	23.74	63.93	54.38
Fast-WAM	16.26	38.54	61.38	79.77	52.60	44.13	66.82	51.36
DreamWAM	53.78	67.15	<u>80.72</u>	**96.67**	71.56	<u>63.61</u>	**94.80**	<u>75.47</u>
Faster-WAM	53.75	63.57	79.09	<u>96.27</u>	61.29	**71.57**	<u>94.67</u>	73.57
Lute 1.4B(A0.87B)	60.00	71.21	72.27	88.89	51.35	41.92	73.36	65.54
Lute 5.0B(A3.05B)	**67.08**	75.10	**81.36**	95.68	82.16	48.48	90.57	**76.83**

B Acting as a World Model

A deployable WAM must preserve useful world estimates as well as robust actions. The smaller Lute and Light-WAM achieve similar canonical success but differ in scene robustness, predictive structure, and joint inference cost. Tab. 4 and Fig. 7 connect category-level control results to the structure of predicted scenes.

The smaller Lute improves over Light-WAM in six of the seven LIBERO-Plus categories, with an aggregate relative gain of about 20%. The largest relative gains occur under texture changes and robot initialization, while noise, layout, lighting, and language variations also improve. Light-WAM retains the higher camera-perturbation score. The aggregate advantage thus reflects gains across several sources of variation, connecting the compact policy's success to broader scene robustness.

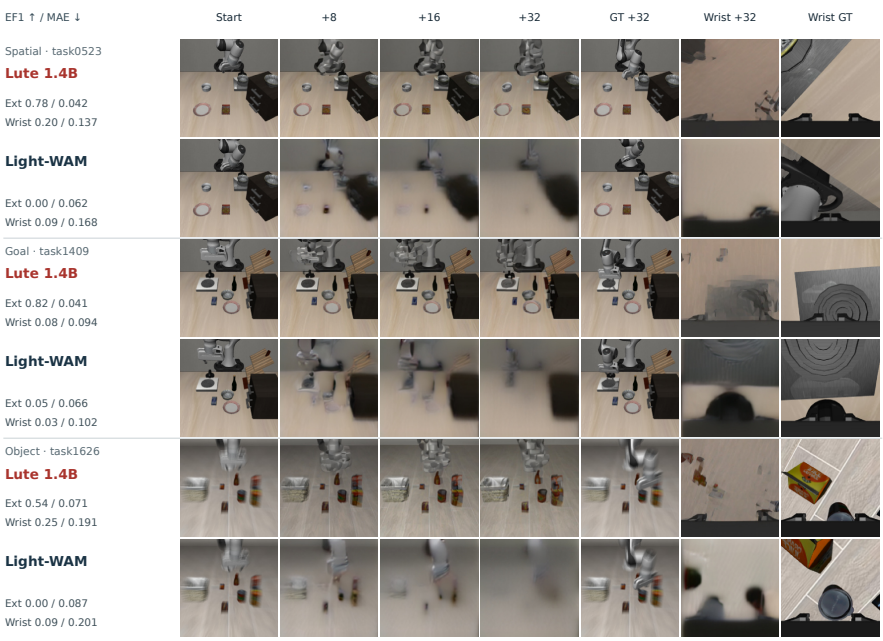

Figure 7: Future-state estimates from the smaller Lute and Light-WAM on Spatial, Goal, and Object examples. Columns show the starting external view, external predictions at $+8$, $+16$, and $+32$, external ground truth (GT) at $+32$, and the wrist prediction and GT at $+32$. Ext and Wrist identify the views. Edge F1 (EF1, higher is better) measures boundary agreement; mean absolute pixel error (MAE, lower is better) measures pixel discrepancy. Lute preserves recognizable scene structure as Light-WAM's object contours fade.

Table 5: Backbone inference cost for a 32-step action chunk and a visual prediction at $t + 4$, on the same GPU with cached text features and excluding visual encoding and decoding. Ratios normalize each cost to its minimum, set to $1.00\times$. Lower values are better.

Method	Latency (ms)	Ratio	Peak memory (GiB)	Ratio
Lute 1.4B	**137.79**	$1.00\times$	**2.97**	$1.00\times$
Lute 5B	254.83	$1.85\times$	9.50	$3.20\times$
Light-WAM	541.76	$3.93\times$	3.74	$1.26\times$
Faster-WAM	542.18	$3.93\times$	10.47	$3.53\times$
Fast-WAM	626.31	$4.55\times$	11.49	$3.87\times$
DreamWAM-joint	606.84	$4.40\times$	11.48	$3.87\times$

Fig. 7 compares the smaller Lute with Light-WAM across Spatial, Goal, and Object examples. Lute preserves recognizable plates, containers, and scene boundaries as the prediction horizon grows. Light-WAM retains broad color regions but progressively loses object contours. This difference appears most clearly in the external view, where stronger edge agreement accompanies lower pixel error for Lute; pixel error also improves in the wrist view. Modest pixel error can therefore hide a loss of the structure needed to distinguish objects and their arrangement. Wrist predictions retain more boundary information as well, with visible errors in object position and scale relative to the observed future.

With visual prediction enabled, Tab. 5 compares the backbone cost of delivering a 32-step action chunk and a visual prediction at $t + 4$. The smaller Lute reduces latency by about 75% and peak memory by about 21% relative to Light-WAM. The larger model reduces latency by about 53% relative to Faster-WAM while using about 9% less memory. Both scales also use less latency and memory than Fast-WAM and DreamWAM-joint for these target outputs, extending the efficiency advantage to visual prediction.

C Implementation Details

C.1 Architecture and training

The visual encoder produces a grid for each camera and observed frame. A learned projection maps its latent channels to the fixed width of the DiT backbone. Tab. 6 gives the interfaces used for the reported benchmark settings; the action horizon is the predicted chunk length, while Q in Sec. 3.1 is the number of actions executed before replanning.

Slow analysis attends within observed frames. Mid analysis alternates spatial and causal temporal attention. Fast combines compact visual tokens with noisy action tokens; actions read their source-frame visual representation and their own chunk, while visual tokens remain action-independent. After action readout, the noised future enters the return pathway. Mid return processes future visual queries without the analysis-stage spatial/temporal alternation. Camera identity uses a learned embedding; physical time enters three-dimensional rotary positional encoding (3D RoPE).

Table 6: Benchmark interfaces and hierarchy. Visual latents have 1024 channels before projection; grids are per camera and frame. Block indices are one-based.

Interface	LIBERO	RoboTwin
Cameras	2	3
State dim.	8	14
Action dim.	7	14
Action horizon	32	40
Latent grid	16×16	12×16
Training epochs	10	5

Stage	Blocks	LIBERO	RoboTwin
Slow analysis	1–3	16×16	12×16
Mid analysis	4–12	8×8	6×8
Fast	13–18	4×4	3×4
Mid return	19–27	8×8	6×8
Slow return	28–30	16×16	12×16

C.2 Orthogonal spatial analysis and synthesis

At each reduction, R_j partitions every channel of every visual grid into non-overlapping 2×2 spatial groups. For one group, let $x_{j,c} \in \mathbb{R}^4$ list the top-left, top-right, bottom-left, and bottom-right values of channel c. The module learns six angles per channel and resolution level. With planes $\mathcal{P} = ((1,2),(1,3),(1,4),(2,3),(2,4),(3,4))$, its matrix is

$$H_{\text{Haar}} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}, \qquad M_{j,c} = G_{\mathcal{P}_6}(\theta_{j,c,6}) \cdots G_{\mathcal{P}_1}(\theta_{j,c,1}) H_{\text{Haar}}. \tag{10}$$

Each $G_{ab}(\theta)$ is an identity matrix except for the (a,b) plane, where its 2×2 block is $\left[\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right]$. All angles start at zero, so the initial transform is Haar. The Slow–Mid and Mid–Fast operators have independent angles; within each operator, the same per-channel matrix is reused across positions, frames, and cameras.

Analysis assigns the first transformed coefficient to the summary and the remaining three to retained detail. Synthesis uses the transpose of that same learned matrix.

$$\begin{bmatrix} s_{j,c} \\ d_{j,c,1} \\ d_{j,c,2} \\ d_{j,c,3} \end{bmatrix} = M_{j,c} x_{j,c}, \qquad W_{j,c}(s_{j,c}, d_{j,c}) = M_{j,c}^{\top} \begin{bmatrix} s_{j,c} \\ d_{j,c,1} \\ d_{j,c,2} \\ d_{j,c,3} \end{bmatrix}. \tag{11}$$

Because products of Givens rotations are orthogonal, $W_{j,c}(R_{j,c}(x_{j,c})) = x_{j,c}$ and $\|x_{j,c}\|_2^2 = |s_{j,c}|^2 + \|d_{j,c}\|_2^2$ in exact arithmetic. The synthesis calculation reuses the analysis angles, so the two R/W levels add $12D$ learned scalars in total for backbone width D.

Let B, F, and V denote batch size, observed-frame count, and camera count; H, W, and D denote grid height, grid width, and backbone width. For an input grid of shape $B \times F \times V \times H \times W \times D$, one reduction outputs a summary of shape $B \times F \times V \times (H/2) \times (W/2) \times D$ and a detail tensor of shape $B \times F \times V \times (H/2) \times (W/2) \times 3 \times D$. The summary continues through the DiT blocks at the next resolution; detail is stored for the corresponding return operation. The second reduction repeats this operation on the Mid grid. A return operation combines its updated summary with the detail

retained at the matching resolution. At initialization these are Haar average and contrast directions; training rotates the basis while preserving their complementary relationship.

C.3 Future prediction and action inference

Each visual query is anchored at a source physical frame r and predicts the encoded observation at $r + Q$. Its action target is a rolling chunk beginning at the same source. The model samples one flow time for the action and future-visual targets of each training example, with $u = \mathrm{sigmoid}(\xi)$ and $\xi \sim \mathcal{N}(0, 1)$. Action and visual Gaussian noises are sampled independently. All three visual inputs to the return path are derived from the same noised future latent $Z_u^{+,(r)}$.

$$N_s^{(r)} = P_Z Z_u^{+,(r)}, \qquad N_m^{(r)} = R_s^S(N_s^{(r)}), \qquad N_f^{(r)} = R_m^S(N_m^{(r)}), \tag{12}$$

where R_j^S selects the summary output of the corresponding analysis operator. The return path also receives the source representation's retained $\Delta_m^{(r)}$ and $\Delta_s^{(r)}$. These details carry observed spatial information into the predictive calculation.

Let $H_{r,h}^{\mathrm{act}}$ be action-token hidden state h for the query at r. The return path projects the Qth state, $P_{aw} H_{r,Q}^{\mathrm{act}}$, across compact visual positions. Mid synthesis combines the source Mid detail with the resulting prediction input; after Mid return blocks, Slow synthesis adds source Slow detail before the latent-velocity head.

For validity indicator m_i and error vector $e_i \in \mathbb{R}^{d_e}$, the masked flow losses are

$$\mathcal{E}(e, m) = \frac{\sum_i m_i \|e_i\|_2^2}{d_e \max(1, \sum_i m_i)}, \qquad \mathcal{L} = \mathcal{E}(v_A - (A^* - \epsilon_A), M_A) + \mathcal{E}(v_Z - (Z^{+*} - \epsilon_Z), M_Z). \tag{13}$$

For the action term, i ranges over valid query and action positions and d_e is the action dimension. For the visual term, i ranges over valid source–future pairs and d_e counts all latent values of one multiview future target. The two separately normalized losses receive equal coefficients. When observations are encoded more densely than the deployment replan interval, training forms phase-consistent causal histories at that interval.

Action-only inference ends at the Fast velocity head. It evaluates the observation-analysis stages and the Fast action blocks for each action-flow step, then executes the selected prefix of the generated chunk. Joint prediction additionally evaluates the future-latent projections, the two W operators, and the return blocks. During standard closed-loop control, the next replan updates the visual history with measured observations. Recursive world-model evaluation instead uses predicted camera views as subsequent visual inputs, as described in Sec. 4.2.

C.4 Evaluation and visualization protocol

The trajectory comparison includes Lute, UniVLA, Light-WAM, π_0, OpenVLA-OFT, DreamWAM joint and unconditional, Fast-WAM, and Faster-WAM. Policies share a task, perturbation, seed, and initial state, then follow their own observations. Curves show measured end-effector positions relative to the start, with lateral motion and height in adjacent views.

Pathway probes replay matched recorded observations and stochastic inputs. At each replan, freezing replaces one selected representation with its first value; the replacement resets at the next replan. The Slow probe acts on a compact analysis representation while its complementary detail remains available to the return path. The Mid probe acts at an executed intermediate block, and the Fast probe acts at the action-velocity output. The mean action change over a chunk measures the shared response across action positions, while deviations from that mean show position-specific changes.

Hidden-state plots fit centered PCA separately by stage and connect samples within a replan. Action-change heatmaps use stage-specific root-mean-square normalization.

Prediction examples begin at recorded diagnostic states after evaluation warmup. Each capture saves a predicted visual future and its accompanying action chunk. The simulator executes a prefix of that chunk and supplies the matched future observation. Displays preserve camera identity and benchmark orientation and show decoded visual predictions beside the corresponding simulator observations. Fast-WAM uses its visual export path, Faster-WAM and DreamWAM joint use their joint samplers, and Light-WAM produces its visual future through a separate generation path.

D MORE VISUALIZATIONS

D.1 ADDITIONAL TRAJECTORIES AND JOINT PREDICTIONS

The gallery pairs end-effector paths with the camera frames recorded at corresponding points in each rollout. The later pages pair visual forecasts with the actions returned in the same inference call.

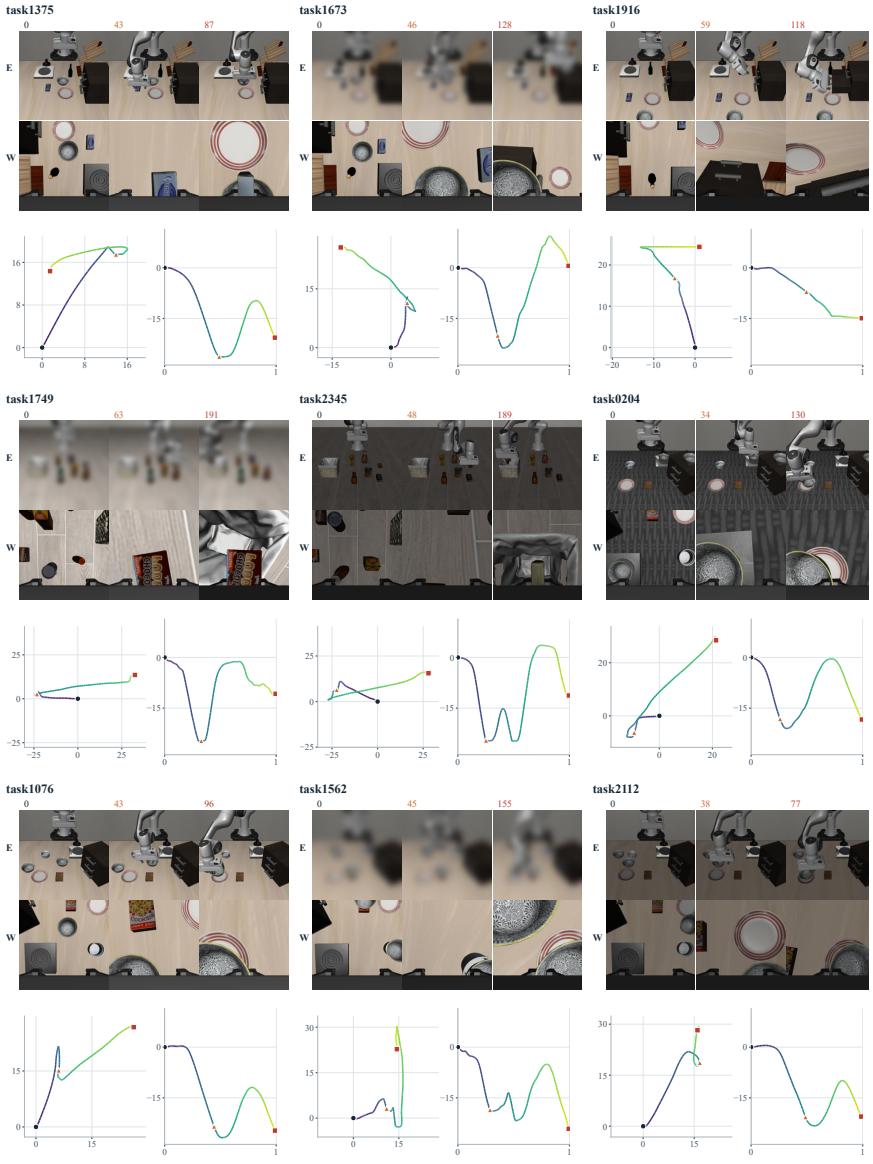

Figure 8: Nine successful Lute rollouts under varied visual conditions, including blurred observations, dim lighting, and changed table textures. External (E) and wrist (W) frames at the start, marked intermediate step, and last saved step align with circle, triangle, and square markers on the end-effector path and height profile. The trajectories combine lateral transport with task-dependent descent and lift. Color denotes rollout progress; planar windows vary by task.

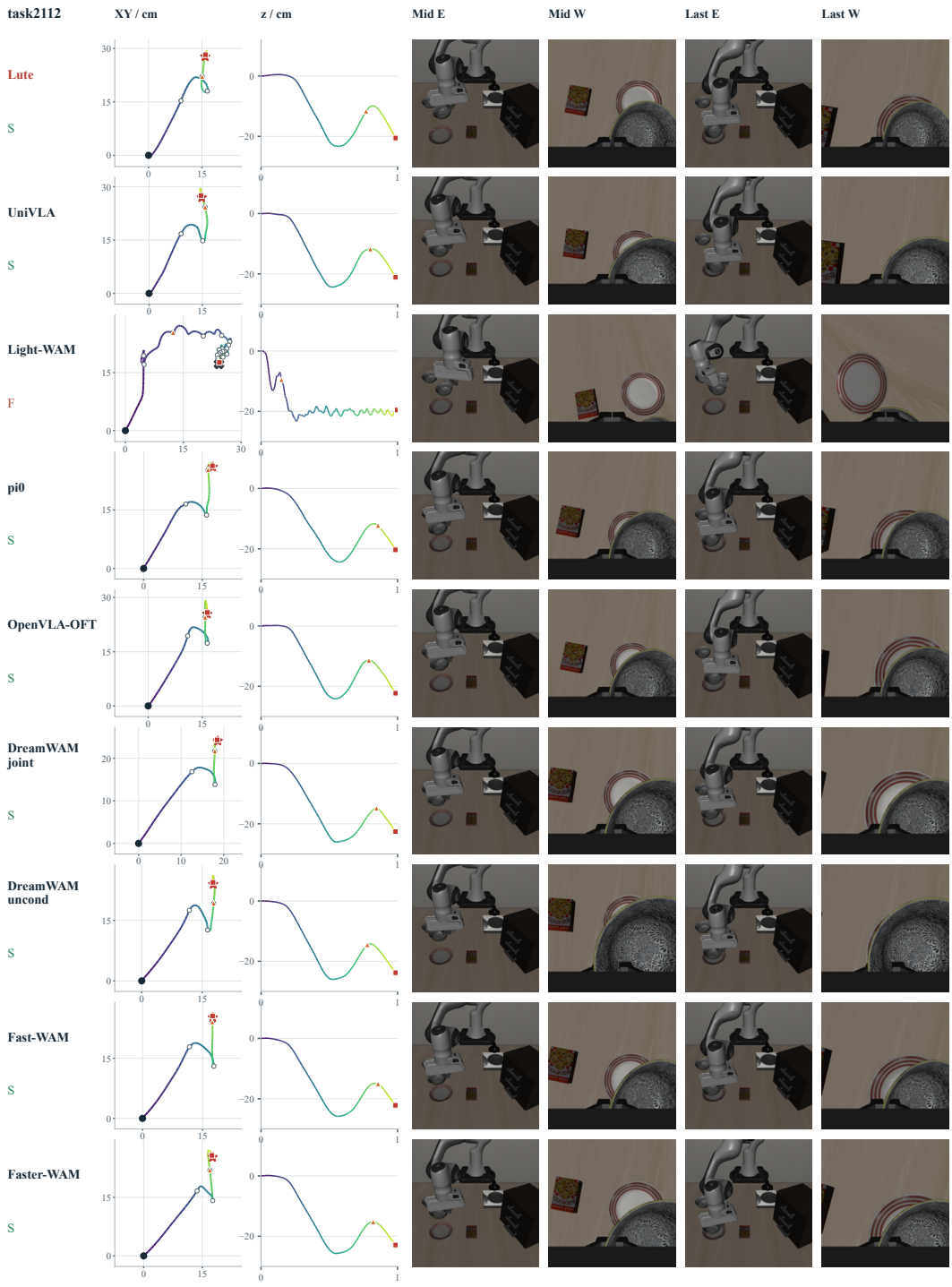

Figure 9: Closed-loop control under a lighting perturbation. Policies begin from the same task and simulator state, then follow their own observations. Lute and the other successful policies descend, lift, and descend again; Light-WAM shows repeated height oscillations after its initial descent and finishes with failure. Mid and Last external (E) and wrist (W) frames correspond to triangle and square markers. S/F denotes success/failure; XY and height are measured in centimeters, with planar windows set per row.

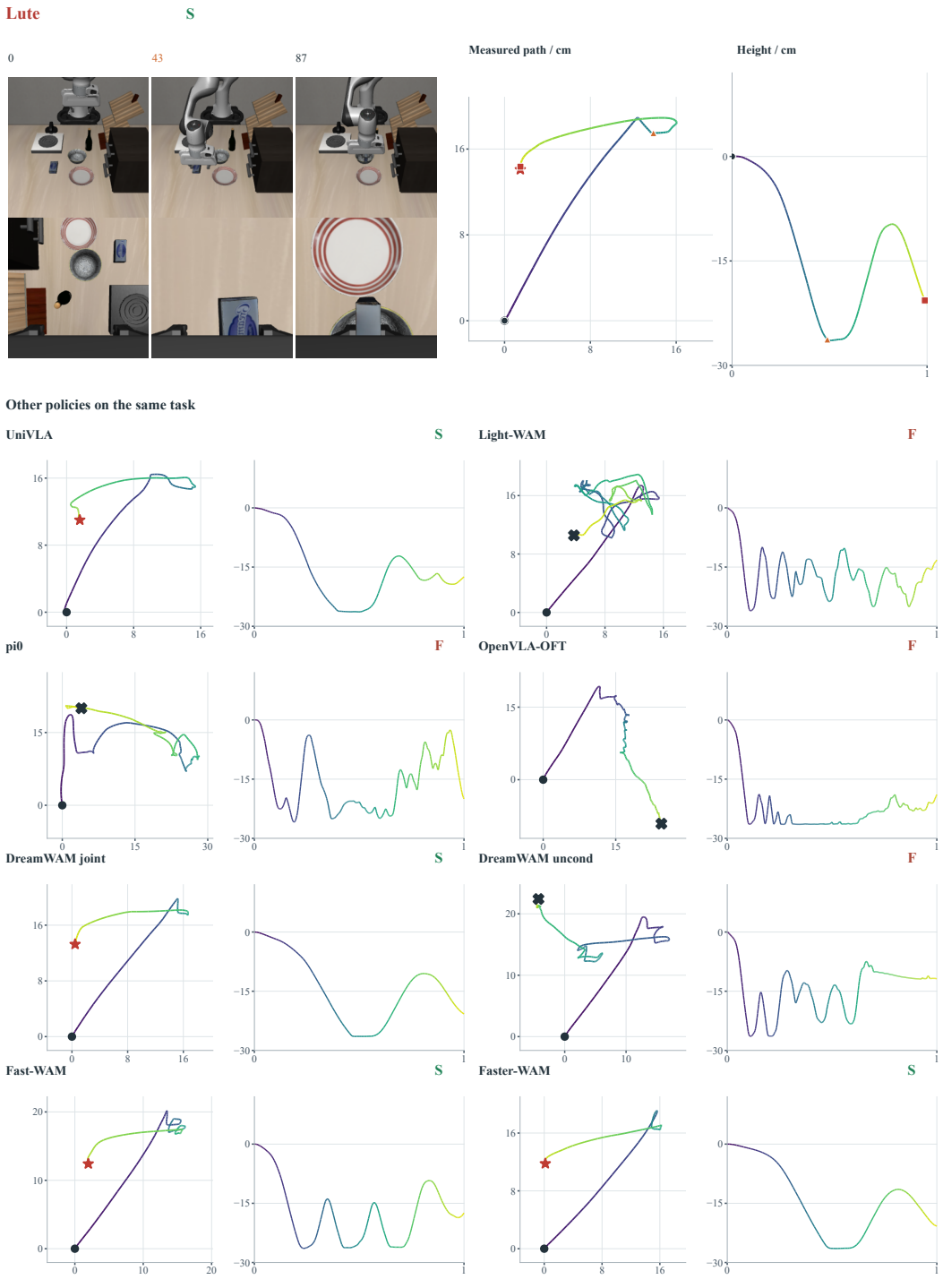

Figure 10: Closed-loop control under language variation. The upper panel pairs a successful Lute rollout with external and wrist observations at three marked steps. Eight comparison policies share the task and initial state. Successful paths include lateral transport following descent, while several failed paths show repeated vertical excursions or lateral drift. S/F and star/cross markers denote success/failure. Trajectories and height are measured in centimeters; planar windows vary by policy.

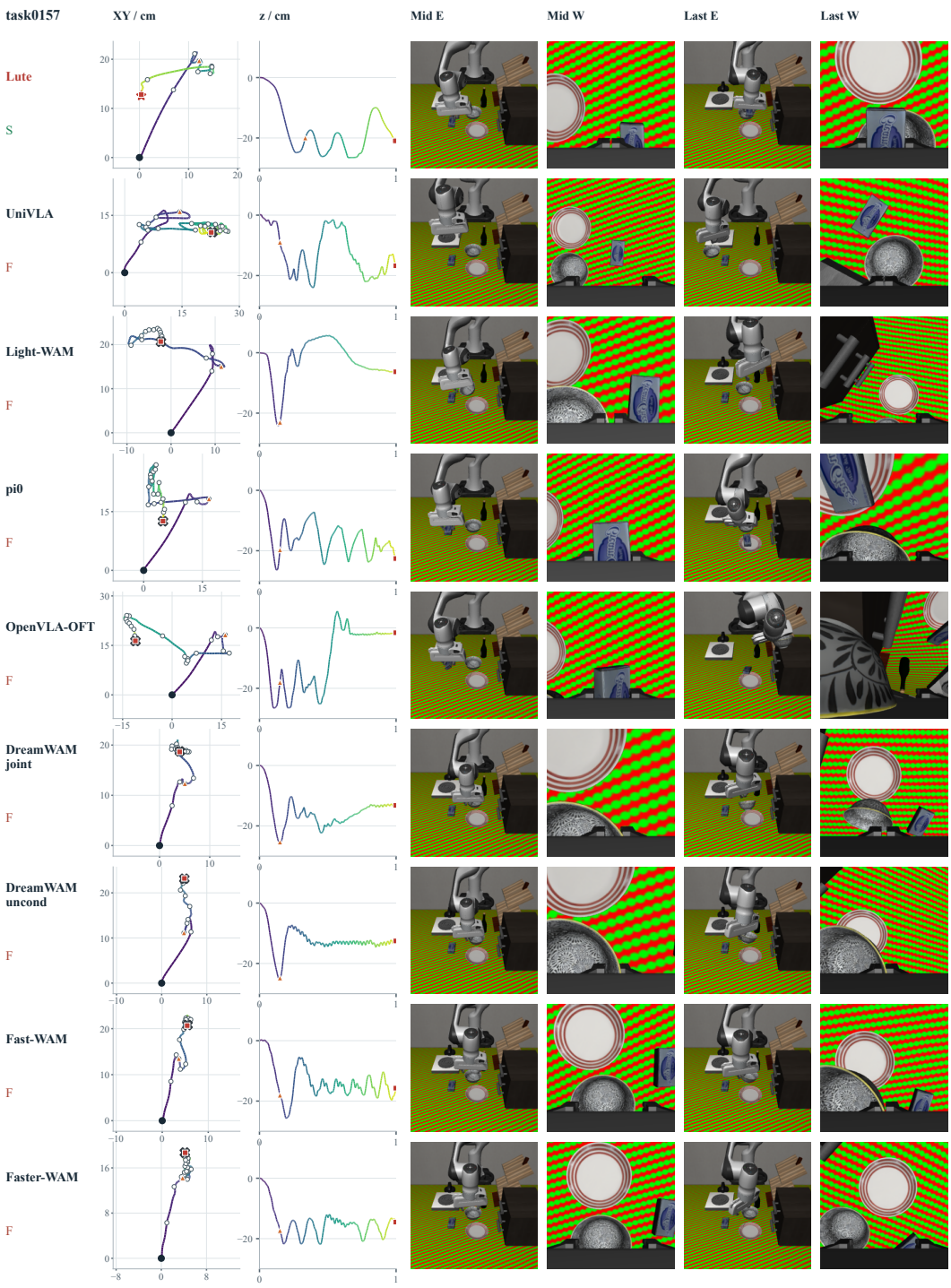

Figure 11: Closed-loop control with a high-contrast table texture. Lute succeeds in the displayed rollout, while all eight comparison policies fail from the same initial state. The paths and height profiles expose different responses to the changed scene, including repeated descents, local motion, and retreat. Mid and Last external (E) and wrist (W) frames align with triangle and square markers. S/F denotes success/failure; height uses a shared scale and planar windows vary by policy.

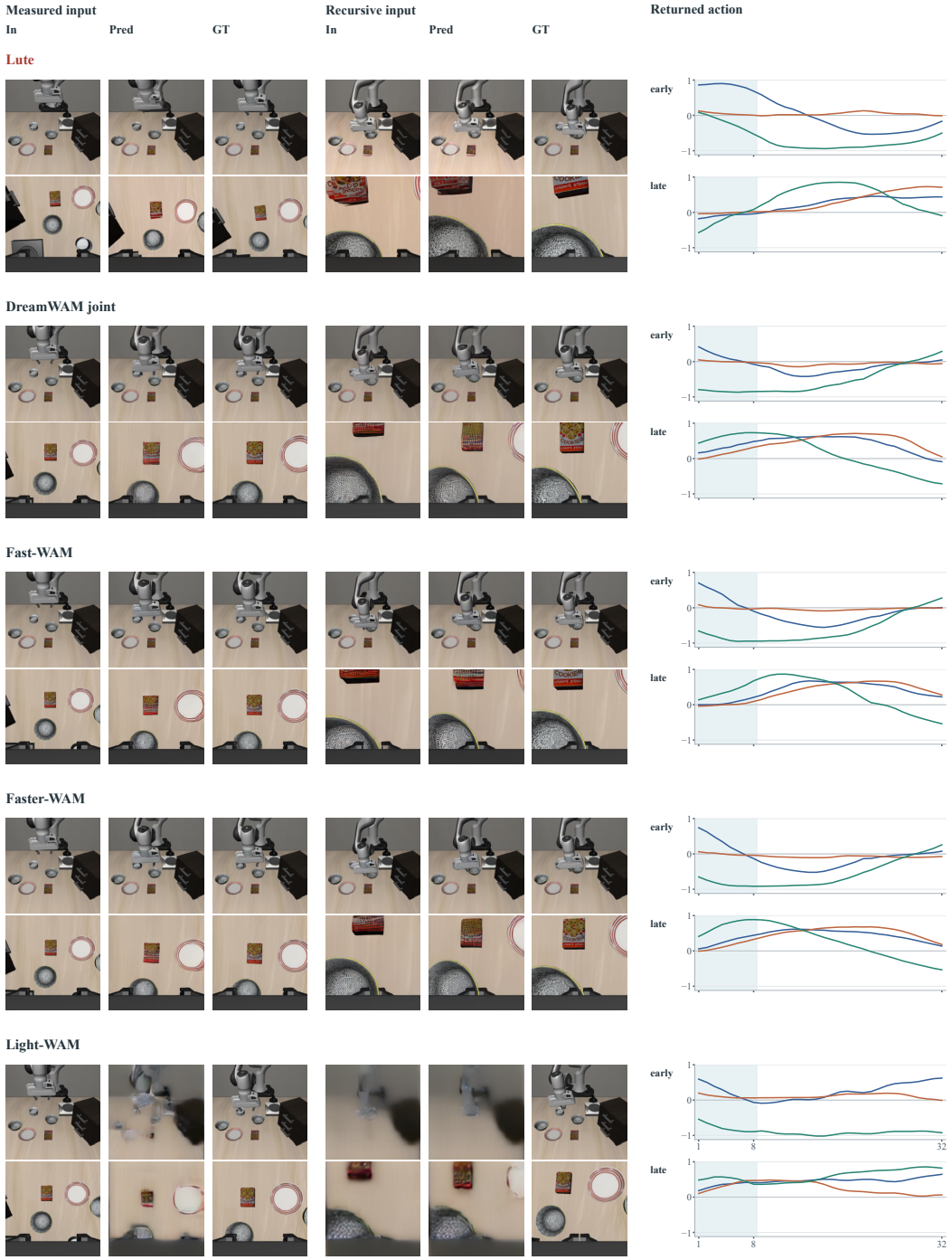

Figure 12: Recursive prediction and returned actions under language variation. Each method starts from the same scene. Measured input and Recursive input panels pair input images (In), next predictions (Pred), and observations after execution (GT), with external views above wrist views. Lute, DreamWAM joint, Fast-WAM, and Faster-WAM preserve recognizable tabletop objects; Light-WAM's external recursive input and prediction become diffuse. Right panels show early and late translation-command chunks; shading marks the executed prefix.

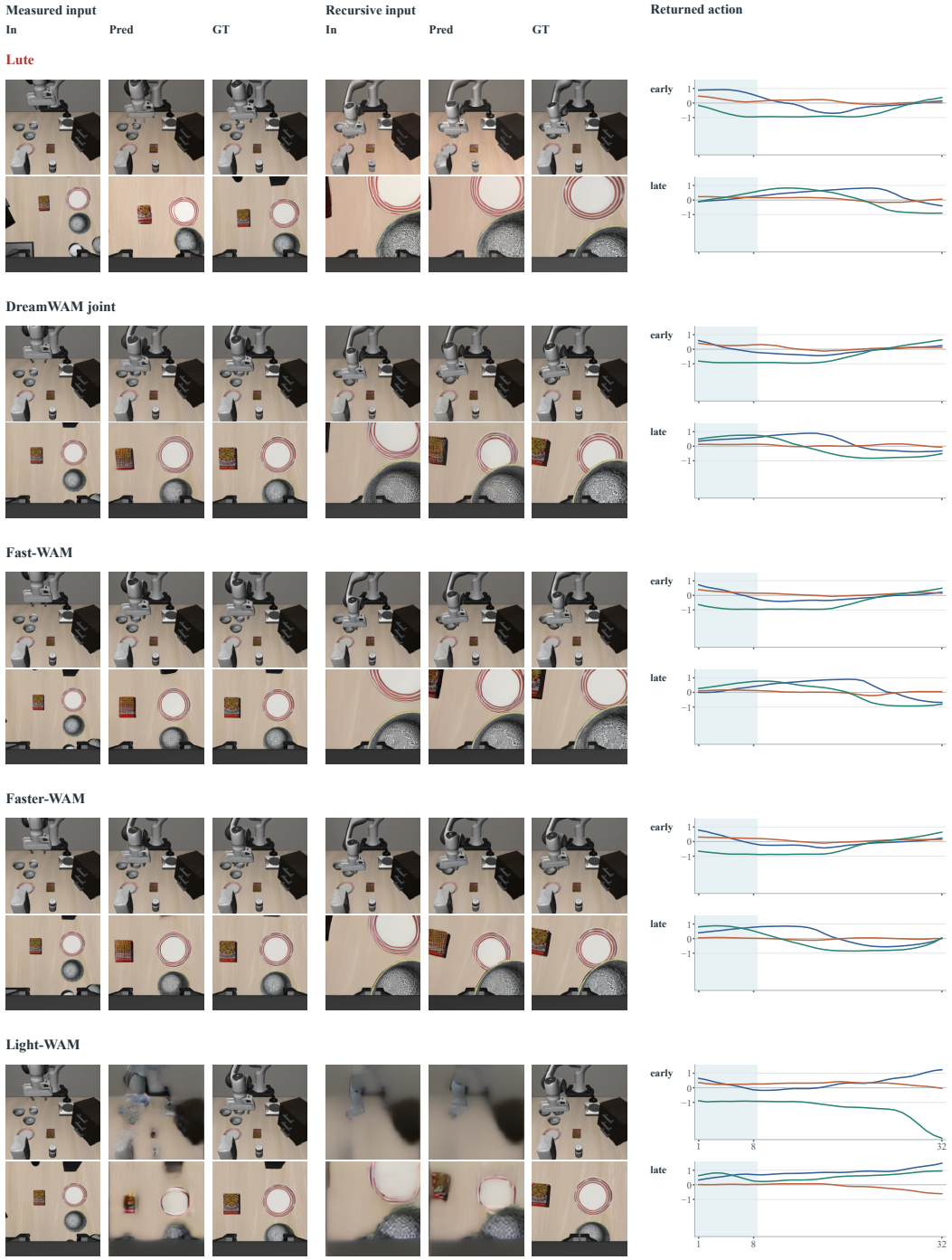

Figure 13: Recursive prediction and returned actions under object-layout variation. Methods share the initial scene, then execute their own actions and use predicted images in later replans. Input images (In), predictions (Pred), and observations after execution (GT) pair external and wrist views. Lute preserves the tabletop objects across both replans, while its wrist forecasts show changes in apparent plate and bowl scale. Light-WAM loses external-view object contours. Right panels show the returned translation-command chunks and shaded execution prefixes.



Figure 14: Multiview world estimates during bell and bottle manipulation. Columns compare input, prediction, and ground truth after execution; H, L, and R denote the head, left-wrist, and right-wrist cameras. The bell example retains the bell and surrounding tabletop arrangement. The two bottle examples preserve the bottle–gripper configuration and background objects at successive phases, with visible blur and pose differences in the later forecast. The views expose both scene structure and local interaction geometry.