ADEPT: ACTION-SENSITIVE WORLD ACTION MODELS WITH ADAPTIVE DELIBERATION FOR ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

World action models (WAMs) compare candidate actions through imagined futures, but effective manipulation requires predictions that distinguish action consequences and computation allocated to decisions that benefit from deliberation. Generic visual prediction can emphasize nuisance variation, recursive errors can obscure candidate differences, and progress-only scoring can favor inaccurate futures. We present ADEPT, an action-sensitive latent WAM that learns *what to predict* for action selection and *when to plan*. A learned token mask preserves action-dependent information; multi-step autoregressive training aligns latent prediction with the deployment horizon. A progress-informed gate invokes candidate evaluation, which combines predicted task progress with rollout consistency. Clean-only component controls and same-state candidate comparisons test representation, recursive training, and action ranking separately. ADEPT achieves 99.1% success on LIBERO and 88.1% Macro-7 on LIBERO-Plus after standard-LIBERO-only training, exceeding its frozen action prior by 16.6 percentage points. On RoboCasa365 it achieves 25.5% split-macro success. Across six real-robot tasks, ADEPT improves pooled success from 55.3% for matched frozen-prior controls to 90.3%. Adaptive deliberation reduces average decision latency by 31% relative to always-on planning in the profiled LIBERO-Spatial setting.

1 INTRODUCTION

Language-conditioned manipulation requires selecting actions whose consequences advance the task under partial observability. World action models (WAMs) address this requirement by predicting how candidate actions change the environment. In contact-rich settings, occlusion, geometric ambiguity, and small errors make such prediction important for avoiding long-horizon failures. Recent vision-language-action (VLA) models (Black et al., 2024; Physical Intelligence et al., 2025; Kim et al., 2024; Bjorck et al., 2025) provide action priors learned from large-scale visual and manipulation data. Reactive action generation, however, need not explicitly compare the predicted consequences of alternative candidates.

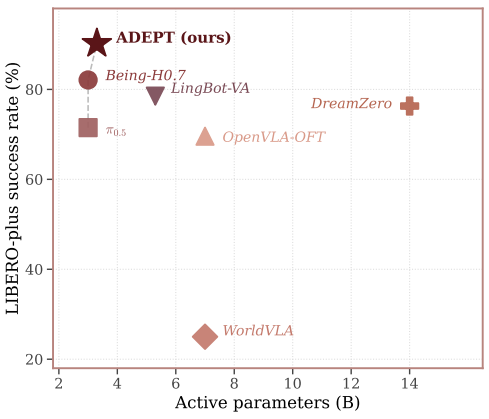


Figure 1: **Reported LIBERO-plus success and model size.** ADEPT-aug: 90.2%, 3.3 B parameters. Baselines follow their reported protocols.

The WAM literature spans joint video–action generation (Zhu et al., 2025; Ye et al., 2026b; Li et al., 2026a), compact latent prediction (Ye et al., 2026a; Yuan et al., 2026; Zheng et al., 2025b), and world-model-guided search or policy improvement (Guo et al., 2025; 2026). Three related challenges motivate our design: ❶ **Visual nuisance variation.** Generic visual prediction can devote capacity to nuisance variation rather than preserve action-dependent differences useful for candidate evaluation. ❷ **Recursive prediction errors.** Single-step supervision can leave latent dynamics

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.

unprepared for recursive inputs, allowing errors to accumulate during planning. ❸ **Deliberation cost and reliability.** Uniform planning allocates the same computation to states with different needs, overlooking opportunities to adapt inference cost to the current decision. Moreover, predicted progress alone can favor inaccurate imagined futures.

We present **ADEPT**, an action-sensitive world action model with adaptive deliberation for robotic manipulation. Its core idea is to learn **what to predict for action comparison** and **when to invoke planning**: preserve action-dependent future differences throughout imagined rollouts and allocate candidate evaluation according to decision-specific needs. ❶ **Control-oriented latent factorization (CLF)** learns a token mask through inverse dynamics, between-action predictive variation, and a mask budget to define the predictive state. ❷ **Multi-step autoregressive rollout training (MRT)** then supervises recursive predictions of this state over the deployment horizon, aligning dynamics learning with the rollouts used for candidate evaluation. ❸ **Adaptive deliberation (ADV)** uses a tentative action and model-based signals to choose between direct execution and further candidate evaluation. When invoked, it ranks candidates by predicted progress and an EMA consistency penalty before receding-horizon execution. The action prior proposes candidate actions; latent prediction and deliberation determine which candidate is executed.

The standard-LIBERO checkpoint reaches 88.1% Macro-7 on LIBERO-Plus without training, tuning, or checkpoint selection on its perturbations, versus 71.5% for the frozen backbone. Clean-only ablations isolate representation and rollout-training effects, while same-state, same-candidate comparisons test ranking against direct absolute and relative critics (Tables 5 and 6). The augmented variant reaches 90.2% on held-out configurations of six seen factor families and an unseen robot-initial-state family. Matched random gating tests computation allocation at the profiled 60% planning rate.

The contributions of this work are summarized as follows:

- An action-sensitive WAM linking CLF and MRT to retain action-dependent future differences throughout imagined rollouts, supported by clean-only representation ablations and controlled comparisons of autoregressive, single-step, and teacher-forced training.
- Adaptive deliberation (ADV) for action selection, evaluated through same-candidate ranking controls and matched-budget gating, with measured success–latency trade-offs.
- Extensive simulation and real-world validation: 99.1% on LIBERO, 88.1% on clean-trained LIBERO-Plus, and 25.5% split-macro success on RoboCasa365; six matched real-robot comparisons improve pooled success from 55.3% to 90.3%.

2 Related Work

World-Action Models for Manipulation. Video-action models jointly predict futures and actions (Zhu et al., 2025; Cen et al., 2025; Li et al., 2026a; Bi et al., 2025; Liao et al., 2025; Ye et al., 2026b; Kim et al., 2026b); latent models make prediction compact or optional (Ye et al., 2026a; Yuan et al., 2026; Gao et al., 2025; Huang et al., 2025; Zheng et al., 2025b; Kim et al., 2026a; Luo et al., 2026). LaWAM (Chen et al., 2026) conditions actions on latent subgoals; τ_0-WM (Zhou et al., 2026) ranks re-denoising consistency and rectifies candidates through simulation. CheckVLA (Liu et al., 2026b) checks chunks during execution and repairs unexecuted suffixes. ADEPT gates additional fixed-prior sampling, then ranks masked latent rollouts before execution. Multi-step supervision also appears in visuo-tactile prediction (Higuera et al., 2026).

Representations, Planning, and Adaptive Computation. Prior work includes task-informed abstractions (Zhang et al., 2021; Fu et al., 2021; Islam et al., 2023), decoder-free models (Hansen et al., 2022; 2024), future-answer prediction (Berg et al., 2025), and inverse dynamics (Han & Yilmaz, 2026). ActSWM (Gan et al., 2026) separates alternative-action rollouts and recovers actions through a fixed readout for game planning; Delta-JEPA (Zhang et al., 2026) decodes actions from latent displacements. ADEPT learns control-token selection over frozen features for adaptive short-horizon candidate evaluation in manipulation. Planning and adaptive computation are established (Chandra et al., 2025; Guo et al., 2026; 2025; Liu et al., 2026a; Chun et al., 2025; Graves, 2016). ProgressVLA (Yan et al., 2026) refines actions through differentiable prediction; ADEPT ranks fixed proposals. ACID (Seo et al., 2026) plans with inverse-cycle consistency; ADEPT uses inverse consistency for gating and EMA disagreement for ranking. VLA-ATTC (Li et al., 2026b)

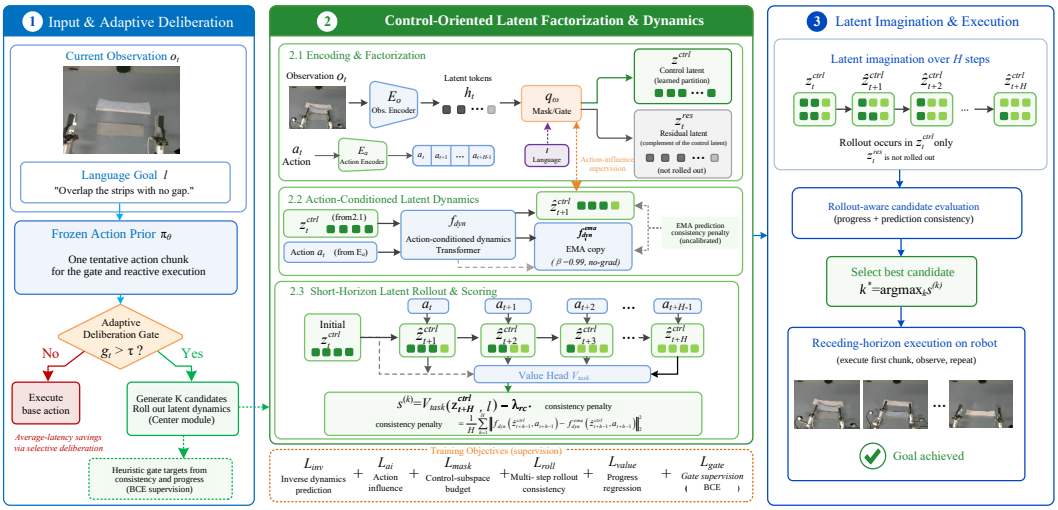


Figure 2: **Overview of ADEPT.** A mask separates the predictive state from the residual. A tentative frozen-prior action supplies one-step gate features. When invoked, deliberation generates K candidates and ranks their latent rollouts by progress and EMA consistency. Value/gate supervision stops gradients to the representation and dynamics.

learns pairwise preferences; our absolute/relative direct critics use ADEPT's protocol rather than reproducing VLA-ATTC.

3 Method

3.1 World-Action Formulation

We address language-conditioned manipulation under partial observability: at each step t, the agent maps a visual observation $\mathbf{o}_t$ and instruction l to a continuous action $\mathbf{a}_t$. ADEPT models candidate action sequences and their future latent states. Let π_θ denote its action prior and p_ϕ its action-conditioned transition model. The predictive state z_t^{ctrl} is defined by the latent factorization in §3.2. The joint distribution for candidate generation and prediction factorizes as

$$P(z^{ctrl}_{t+1:t+H}, \mathbf{a}_{t:t+H-1} \mid \mathbf{o}_{\le t}, l) = \pi_\theta(\mathbf{a}_{t:t+H-1} \mid \mathbf{o}_{\le t}, l) \cdot \prod_{k=0}^{H-1} p_\phi(z^{ctrl}_{t+k+1} \mid z^{ctrl}_{t+k}, \mathbf{a}_{t+k}). \quad (1)$$

We instantiate the action prior with a pretrained $\pi_{0.5}$ VLA and keep its parameters fixed. It samples candidates from full observations, while the transition model predicts their latent consequences. Freezing the prior fixes its parameters, not the executed policy: rollout-aware selection can change the chosen action and subsequent observations. ADEPT directly executes the tentative action or evaluates candidates before committing to the first action chunk.

3.2 Control-Oriented Latent Factorization

CLF defines the information retained for predicting candidate consequences. Its control-oriented state z^{ctrl} is learned to preserve action-dependent differences; recursive training propagates this state through the candidate window, and the scorer uses the resulting futures for action selection.

The encoder produces tokens $\mathbf{h}_t = \{h_t^1, \dots, h_t^N\}$, $h_t^i \in \mathbb{R}^d$. A lightweight gate predicts a soft per-token control mask $m_t^i = q_\omega(h_t^i, l) \in [0, 1]$, splitting the latent into a control-oriented component and a residual,

$$z_t^{ctrl} = \{m_t^i\, h_t^i\}_{i=1}^N, \qquad z_t^{res} = \{(1 - m_t^i)\, h_t^i\}_{i=1}^N. \quad (2)$$

The transition model rolls out this predictive state, $\hat{z}_{t+1}^{ctrl} = f_{dyn}(z_t^{ctrl}, a_t)$; the residual is excluded from candidate scoring. The resulting soft bottleneck constrains what the scorer can use. The

objectives below encourage action sensitivity and temporal consistency, and f_{dyn} is trained to output within the masked subspace so the mask need not be re-invoked during rollout.

Unanchored, the partition is arbitrary, so we impose three complementary objectives. *(i)* An *inverse-dynamics prediction* term requires consecutive action-sensitive latents to recover the executed action,

$$\mathcal{L}_{inv} = \mathbb{E}\Big[\big\|a_t - g_\psi(z_t^{ctrl}, z_{t+1}^{enc,ctrl})\big\|_2^2\Big]. \tag{3}$$

(ii) Action prediction alone does not encourage distinguishable candidate futures. Motivated by action influence (Klyubin et al., 2005; Mohamed & Rezende, 2015), fix the observation history and instruction $c_t = (\mathbf{o}_{\le t}, l)$, and hence z_t^{ctrl}. Consider the KL contrast for a single action, with c_t implicit below,

$$\mathcal{I}(a \to z_{t+1}^{ctrl} \mid z_t^{ctrl}) = D_{\mathrm{KL}}\Big[p(z_{t+1}^{ctrl} \mid z_t^{ctrl}, a) \;\Big\|\; p(z_{t+1}^{ctrl} \mid z_t^{ctrl})\Big], \tag{4}$$

with $p(z_{t+1}^{ctrl} \mid z_t^{ctrl}, a) = \mathcal{N}(f_{dyn}(z_t^{ctrl}, a), \sigma^2 I)$. The marginal is a Gaussian mixture over policy actions. Our implemented objective instead uses a Gaussian reference $q_{ref} = \mathcal{N}(\bar{\mu}_t, \sigma^2 I)$, with a mean estimated from M observational samples $\{a^{(j)}\} \sim \pi_\theta$. Its KL is

$$\mathcal{I}_{var}(a; z_t^{ctrl}) = \tfrac{1}{2\sigma^2}\big\|f_{dyn}(z_t^{ctrl}, a) - \bar{\mu}_t\big\|_2^2, \quad \bar{\mu}_t = \tfrac{1}{M}\textstyle\sum_j f_{dyn}(z_t^{ctrl}, a^{(j)}), \tag{5}$$

giving the between-action-variance surrogate loss

$$\mathcal{L}_{ai} = -\,\mathbb{E}_{a\sim\pi_\theta}\big[\mathcal{I}_{var}(a; z_t^{ctrl})\big]. \tag{6}$$

For the population mean, $\mathbb{E}_a\mathcal{I}_{var} = \operatorname{tr}\operatorname{Cov}_a[f_{dyn}(z_t^{ctrl}, a)]/(2\sigma^2)$; this is not the mixture KL in Eq. (4). The isolated variance term scales quadratically with prediction magnitude; rollout supervision anchors predictions but does not bound all candidate-action predictions. These observation–action objectives encourage action sensitivity, whereas reward-based bisimulation (Zhang et al., 2021) encodes behavioral and reward equivalence. Action sensitivity alone does not ensure task sufficiency: static goals can matter while irrelevant variables can respond to action. *(iii)* A budget regularizer

$$\mathcal{L}_{mask} = \left(\tfrac{1}{N}\sum_{i=1}^{N} m_t^i - \rho\right)^2 + \lambda_{ent}\tfrac{1}{N}\sum_{i=1}^{N}\Big[-m_t^i \log m_t^i - (1-m_t^i)\log(1-m_t^i)\Big] \tag{7}$$

encourages mean mask mass $\rho \in (0,1)$ and discourages degenerate masks. The three terms regulate action prediction, action sensitivity, and partition capacity. Soft masking retains all N tokens; ρ is an information budget, not a guaranteed token-count or compute reduction.

3.3 Action-Conditioned Multi-Step Rollout Consistency

With single-step supervision, autoregressive deployment can accumulate prediction error, obscuring the differences needed for action ranking. Our setting requires short-horizon prediction over a finite candidate-evaluation window before re-encoding.

We train f_{dyn} recursively in the control latent space to reduce this mismatch. From the current latent z_t^{ctrl} and a demonstration action sequence $a_{t:t+H-1}$, we unroll

$$\hat{z}_{t+k}^{ctrl} = f_{dyn}\big(\hat{z}_{t+k-1}^{ctrl},\, a_{t+k-1}\big), \quad k = 1, \dots, H, \qquad \hat{z}_t^{ctrl} \equiv z_t^{ctrl}, \tag{8}$$

and supervise the imagined trajectory against encoder targets,

$$\mathcal{L}_{roll} = \tfrac{1}{H}\sum_{k=1}^{H}\big\|\hat{z}_{t+k}^{ctrl} - z_{t+k}^{enc,ctrl}\big\|_2^2. \tag{9}$$

$\mathcal{L}_{roll}$ anchors recursive predictions to encoder targets over a finite window; it reduces recursive error without guaranteeing task relevance or eliminating shifts in states and candidate actions at deployment. Under receding-horizon control the agent executes the first chunk before re-encoding, so long-horizon benefits arise from repeated replanning, not modeling beyond the window.

3.4 Adaptive Deliberation for Action Selection

ADEPT uses its predicted latent trajectories to select actions. The decision rule specifies when to evaluate multiple candidates and how to score their predicted consequences.

Adaptive deliberation gate. Before execution, the action prior supplies a tentative action a_t for a one-step roll-in. A progress-informed scheduling gate uses the current state, predicted features, and available history to decide whether to evaluate further candidates,

$$g_t = \sigma\big(\mathbf{w}^\top \phi_t + b\big), \quad \phi_t = \big[\,\mathrm{pool}(z_t^{ctrl}),\ V_{task}(z_t^{ctrl}, l),\ \epsilon_{inv}^{(t)},\ \Delta_{ref}^{(t)},\ \Delta_{bg}^{(t)},\ \rho_{ctrl}^{(t)}\,\big], \tag{10}$$

whose six features encode the pooled state, predicted progress, and model-internal discrepancies. Using this one-step roll-in $\hat{z}_{t+1}^{ctrl} = f_{dyn}(z_t^{ctrl}, a_t)$, they include forward–inverse consistency $\epsilon_{inv}^{(t)} = \|\mathrm{InvDyn}(z_t^{ctrl}, \hat{z}_{t+1}^{ctrl}) - a_t\|_2^2$, a history-reference contrast $\Delta_{ref}^{(t)} = \|\hat{z}_{t+1}^{ctrl} - \bar{z}^{hist}\|_2$, a background-drift feature $\Delta_{bg}^{(t)}$, and control displacement $\rho_{ctrl}^{(t)} = \|\hat{z}_{t+1}^{ctrl} - z_t^{ctrl}\|_2$. Here $\bar{z}^{hist}$ is a cached EMA of past predictions, not a current-state action marginal.

The gate $\{\mathbf{w}, b\}$ is trained by BCE against a soft scheduling target that increases with standardized discrepancies and decreases with predicted progress. The learned gate distills an internal scheduling score; its output represents invocation preference rather than realized planning benefit. A threshold τ calibrated on held-out training data sets the average invocation budget: $g_t \le \tau$ executes the tentative action; otherwise, candidate evaluation runs. This controls average computation rather than worst-case latency.

Rollout-aware candidate evaluation. When the gate fires, the model rolls out a latent trajectory for each candidate sequence $A^{(k)} = (a_t^{(k)}, \ldots, a_{t+H-1}^{(k)})$. Progress prediction alone can favor an inaccurate imagined state. We therefore score terminal predicted progress with a trajectory-level EMA consistency penalty, so candidate evaluation depends on both predicted progress and model agreement,

$$s^{(k)} = \underbrace{V_{task}\Big(\hat{z}_{t+H}^{ctrl,(k)},\, l\Big)}_{\text{terminal predicted progress}} - \lambda_{rc}\ \underbrace{\frac{1}{H}\sum_{h=1}^{H} \Big\| f_{dyn}\big(\hat{z}_{t+h-1}^{ctrl,(k)},\, a_{t+h-1}^{(k)}\big) - f_{dyn}^{ema}\big(\hat{z}_{t+h-1}^{ctrl,(k)},\, a_{t+h-1}^{(k)}\big)\Big\|_2^2}_{\text{rollout consistency penalty}}, \tag{11}$$

where f_{dyn}^{ema} is an EMA copy of the dynamics model ($\phi^{ema} \leftarrow \beta\,\phi^{ema} + (1-\beta)\,\phi$). The penalty favors candidates that stay close to this slowly-updated reference—a self-consistency regularizer, not calibrated uncertainty. The head V_{task} regresses timestamp progress from offline demonstrations,

$$\mathcal{L}_{value} = \mathbb{E}\Big[\big(V_{task}(\mathrm{sg}[z_u^{enc,ctrl}],\, l) - y_u\big)^2\Big], \qquad y_u = \frac{u+1}{T}, \quad u = t + H < T, \tag{12}$$

where T is demonstration length, u is zero-indexed, and sg stops gradients. Training uses encoded demonstration states; candidate scoring uses imagined terminal states. This progress proxy is not a calibrated success probability and may be unreliable during failure, stalling, or recovery. No evaluation rollout, score, or failure labels supervise the heads. Both $\mathcal{L}_{value}$ and $\mathcal{L}_{gate}$ update only their respective heads: their inputs are detached from the encoder, mask, and dynamics. Thus only representation/dynamics learning is reward-free; planning and gating use demonstration-progress supervision. We select $k^\star = \arg\max_k s^{(k)}$ and execute the first chunk of $A_t^\star = A^{(k^\star)}$ before replanning.

4 Experiments

4.1 Datasets and Implementation

We evaluate transfer robustness, candidate scoring, and deliberation allocation on LIBERO (Liu et al., 2023), LIBERO-plus (Fei et al., 2025), RoboCasa365 (Nasiriany et al., 2026) (50 tasks), and six real-robot tasks (Franka, Custom Arm, ARX R5; 50 trials each). Deployment uses $H = 4$ and EMA decay $\beta = 0.99$; implementation details and Algorithm 1 are in App. B. ADEPT-clean is trained, tuned, and selected only on standard LIBERO. ADEPT-aug additionally trains its

mask, dynamics, and heads on the released generalized split covering six factor families; prior and encoder remain frozen. Evaluation variants, rollouts, scores, and failures are excluded from training. Value and gate heads use detached inputs with demonstration-progress and internal-discrepancy supervision.

4.2 Main Results

On near-saturated LIBERO, ADEPT reaches 99.1% (perfect Spatial/Object), versus 98.5% for the highest reported comparison entry. On all 10,030 LIBERO-plus variants, ADEPT-clean reaches 88.1% Macro-7 versus 71.5% for its frozen backbone (+16.6 pp), without exposure to LIBERO-plus perturbations. Among standard-LIBERO-only comparisons, it is comparable to ACoT-VLA (87.5% Macro-7) and above ABot-M0 (81.6%). It also exceeds augmented-trained OpenVLA-OFT+ (80.7%; Table 2). ADEPT-aug reaches 90.2%, a further 2.1 pp and above ACoT-VLA's augmented-training 88.5%, on held-out configurations of six seen families plus the unseen robot-initial-state family (Table 2). On RoboCasa365, ADEPT reaches 25.5% split-macro success versus 16.0% for the listed $\pi_{0.5}$ entry; all methods use the same three-split averaging, which differs from the official task-weighted Overall (per-split results: Table 19).

Table 1: Success (%) on LIBERO and RoboCasa365 (Nasiriany et al., 2026). RoboCasa365 uses Human300 and published baselines; split-macro equally averages Table 19's three split rates, unlike official task-weighted Overall (RoboCasa Team, 2026).

(a) LIBERO (four task suites)

Method	Spat.	Obj.	Goal	Long	Avg
OpenVLA (2024)	84.7	88.4	79.2	53.7	76.5
π_0 (2024)	96.8	98.8	95.8	85.2	94.2
UniVLA (2025)	95.4	98.8	93.6	94.0	95.4
GR00T-N1 (2025)	94.4	97.6	93.0	90.6	93.9
$\pi_{0.5}$ (2025)	97.4	98.4	96.2	95.4	96.85
X-VLA (2025a)	98.2	98.6	97.8	97.6	98.1
LingBot-VA (2026a)	98.5	99.6	97.2	98.5	98.5
ADEPT (Ours)	**100.0**	**100.0**	**98.0**	**98.5**	**99.1**

(b) RoboCasa365 (50-task multi-task)

Method	Split-macro
π_0 (2024)	13.9
$\pi_{0.5}$ (2025)	16.0
GigaWorld-Policy (2026a)	19.7
GR00T-N1.6 (2026)	20.7
GR00T-N1.5† (2025)	22.7
ADEPT (Ours)	**25.5**

Split-macro: equal mean of three split rates. †1.5× horizon, RoboCasa 1.0.1.

Table 2: Clean/augmented LIBERO-plus (Fei et al., 2025): all 10,030 variants. Macro-7 equally averages seven family rates; published methods are recomputed from their family rates (reported: OFT+ 79.6, ACoT-VLA 86.6/88.0). Backbone/ADEPT results are ours.

Method	Generalized-split training	Macro-7 (%)
$\pi_{0.5}$ (frozen backbone) (2025)	No	71.5
OpenVLA-OFT+ (2025)	Yes	80.7
ACoT-VLA (2026)	No / Yes	87.5 / 88.5
ADEPT-clean	**No**	**88.1**
ADEPT-aug	Six families; prior frozen	90.2

Table 3: LIBERO-plus family rates (%). Protocols and Macro-7 follow Table 2; robot initial state remains unseen. Backbone/ADEPT results are ours; other entries follow published protocols (ACoT-VLA and ABot-M0: standard-LIBERO-only; full table in App. F). Bold: best per column.

Method	Camera	Robot init.	Lang.	Light	Bg.	Noise	Layout	Macro-7
OpenVLA-OFT (2025)	56.4	31.9	79.5	88.7	93.3	75.8	74.2	71.4
AVA-VLA (2025)	55.5	25.9	85.6	95.5	88.9	78.0	74.1	71.9
$\pi_{0.5}$ (2025)	79.5	26.5	74.5	88.3	88.1	71.2	72.2	71.5
ABot-M0 (2026)	60.4	67.9	86.4	96.2	91.6	86.4	82.6	81.6
ACoT-VLA (2026)	72.6	**82.6**	**87.5**	97.7	96.5	87.8	**88.1**	87.5
ADEPT-clean	96.0	65.0	81.0	96.0	97.0	97.5	84.2	88.1
ADEPT-aug	**98.8**	68.0	83.1	**98.2**	**99.0**	**99.1**	85.4	**90.2**

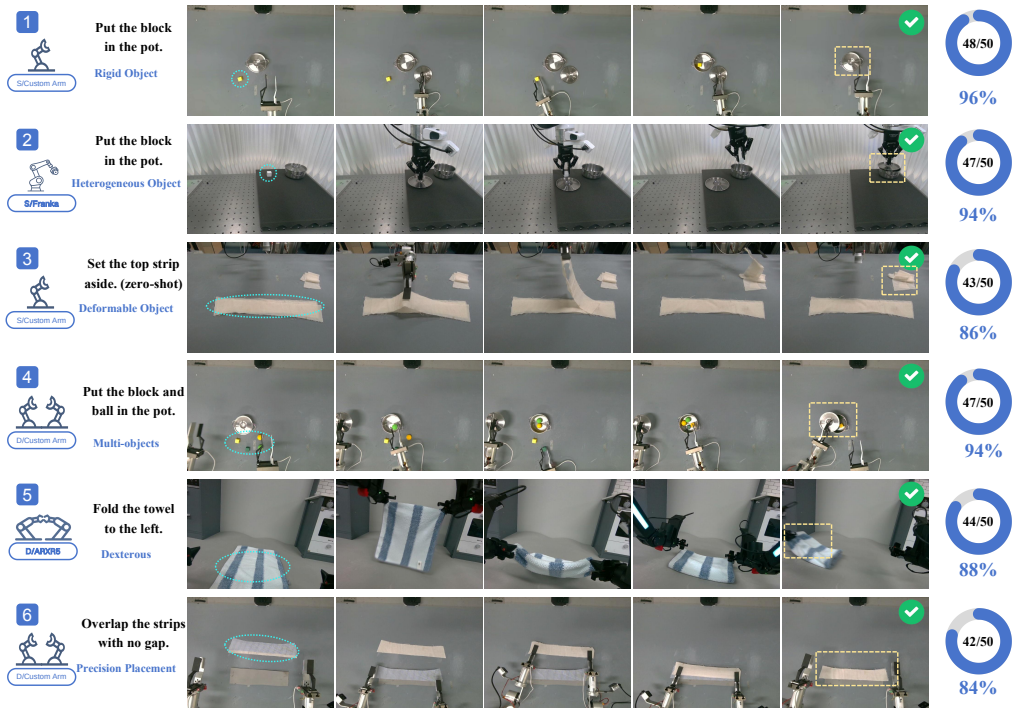


Figure 3: **Real-robot evaluation across diverse manipulation skills.** Six tasks—rigid, heterogeneous, deformable, multi-object, dexterous, precision-placement—on three embodiments (Franka, Custom Arm, ARX R5), single- and dual-arm. Rings report success over 50 trials; ADEPT attains 84–96% (mean 90.3%), with 86% on T3 without base-policy adaptation. For T3, "zero-shot" refers only to the unadapted base policy.

Table 4: Matched real-robot controls: successes over 50 full-reset trials per task. The frozen $\pi_{0.5}$ checkpoint and scene match ADEPT for each task in Figure 3. Pooled success increases from 55.3% to 90.3%.

Method	T1	T2	T3	T4	T5	T6	Pooled
Frozen $\pi_{0.5}$	35	33	10	31	32	25	166/300
ADEPT (Ours)	**48**	**47**	**43**	**47**	**44**	**42**	**271/300**
Gain (pp)	26	28	66	32	24	34	35.0

Where applicable, demonstrations adapt the base policy before freezing; T3 has no such adaptation and T5 uses ARX R5. Table 4 shows gains of 24–66 percentage points. Task-wise binomial intervals appear in App. F.2; pooled success summarizes these six tasks.

4.3 Ablation Studies

Action-sensitive representation and recursive prediction. Under standard-LIBERO-only training (Table 5), replacing CLF with full-latent prediction lowers Macro-7 from 88.1% to 78.4%; removing $\mathcal{L}_{ai}$ gives 81.0%. The mask limits retained information, while $\mathcal{L}_{ai}$ encourages differences between candidate futures. Their separate drops support learning which information to predict for selection. In the distinct four-suite protocol (Table 8), removing the mask objective or factorization costs 5.5 or 7.5 points, compared with 2.5 or 3.0 for the inverse or action-variance objective. These effects connect selective prediction to control; App. H tracks prediction scale alongside candidate discrimination across checkpoints.

At fixed deployment $H = 4$, single-step and teacher-forced training give 81.2% and 83.6%, versus 88.1% for recursive training (Table 5). Recursive training exposes dynamics to accumulated prediction errors without changing planning depth. The separate Spatial diagnostic gives 97.0% for

Table 5: Clean-only component attribution on LIBERO-Plus (Macro-7, %). All variants use standard-LIBERO training. Candidate evaluation uses $K = 5$ and deployment horizon $H = 4$; training horizons are shown separately. Δ is relative to Full.

Variant	Train → deploy	Macro-7	Δ (pp)
Frozen prior	n/a	71.5	−16.6
Full ADEPT-clean	$4 \to 4$	**88.1**	0.0
Full-latent prediction (no CLF)	$4 \to 4$	78.4	−9.7
Single-step training (no MRT)	$1 \to 4$	81.2	−6.9
Teacher-forced training	$4 \to 4$	83.6	−4.5
Always-on, progress only (no ADV)	$4 \to 4$	82.8	−5.3
Without action-variance loss	$4 \to 4$	81.0	−7.1

recursive $H = 4$, 92.0% for teacher forcing, and 94.0% for $H = 8$ (Table 9), so greater depth alone does not explain the gain. Figures 4–5 show representation and rollout behavior; outcome controls avoid comparing MSE across differently scaled latent spaces.

Candidate ranking and closed-loop control. Table 6 holds saved states and $K = 5$ candidates fixed. Branches restore simulator and controller state, execute $H = 4$ chunks, and share a continuation policy; outcome labels are independent of progress supervision (App. G). Full scoring improves pairwise accuracy from 75% to 80% and selected-candidate outcome from 82% to 85% over the evaluated relative critic. Removing $\mathcal{L}_{ai}$ lowers accuracy to 73%: encouraging candidate-dependent predictions affects their ordering, connecting representation learning to action selection. The selected-outcome gap is 3.0 points; App. H reports the paired full-minus-control differences at fixed tasks and checkpoints and specifies critic inputs, supervision, and training budgets.

Table 6: Same-state candidate ranking (%). Scorers share saved states, candidate arrays, and execution outcomes. Q_{sel} is selected-candidate outcome; regret is its gap to the shared 90.0% empirical best-candidate reference. Sample: $N = 2000$ anchors from 1000 source episodes; $R = 5$ repeats per candidate; 70.0% eligible anchors.

Scorer	Pairwise acc.	Q_{sel}	Regret (pp)
Direct absolute-progress critic	70.0	78.0	12.0
Relative critic	75.0	82.0	8.0
Progress-only rollout	66.0	74.0	16.0
Full scorer without $\mathcal{L}_{ai}$	73.0	80.0	10.0
Full rollout-and-consistency scorer	**80.0**	**85.0**	**5.0**

Table 7: Adaptive deliberation and candidate scoring on LIBERO-Spatial. Mean success over three seeds, 50 episodes each. *Delib. rate*: fraction of steps invoking planning; random gating matches Ours (60%). Latency: ms per decision on one A100; ratios use the 90 ms reactive baseline.

Variant	Avg	Delib. rate	Ratio	Latency
No delib. (prior only)	88.0	0%	1.00×	90.0
Always delib. (no gate)	90.0	100%	4.67×	420.0
Random gate (matched)	92.0	60%	3.20×	288.0
Gate w/o V_{task}	92.0	62%	3.29×	295.9
Gate w/o disagr. inputs	92.0	58%	3.14×	282.7
Full adaptive gate (Ours)	**98.0**	**60%**	**3.21×**	**289.3**
Rollout-aware value (gate fixed to full, r=60%):				
Task value only (λ_{rc}=0)	86.0	60%	3.20×	287.8
Consistency only (no V_{task})	92.0	60%	3.21×	288.6
Direct critic (no rollout)	90.0	60%	2.72×	244.6
Full rollout-aware (Ours)	**98.0**	**60%**	**3.21×**	**289.3**

Table 8: CLF objective ablation: four-suite mean (%), replan= 5, $n = 200$, shared $\pi_{0.5}$ backbone. Δ from Full; descriptive differences under a protocol distinct from Table 1.

Variant	Avg
Full (Ours)	**97.5**
w/o $\mathcal{L}_{inv}$	95.0 (−2.5)
w/o $\mathcal{L}_{ai}$	94.5 (−3.0)
w/o $\mathcal{L}_{mask}$	92.0 (−5.5)
Bisim. mask	93.5 (−4.0)
w/o factoriz.	90.0 (−7.5)

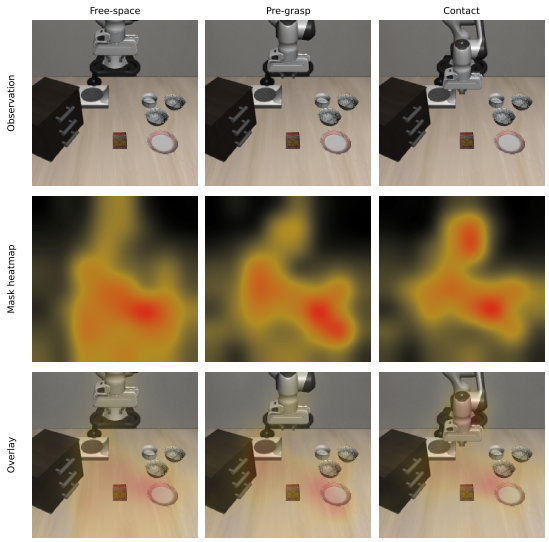


Figure 4: Control-mask saliency $m_t^i = q_\omega(h_t^i, l)$ in free-space, pre-grasp and contact phases (observation / mask / overlay). The mask emphasizes the gripper and black bowl on the stove.

Table 9: Rollout-consistency ablation on LIBERO-Spatial (%). Supervision and horizon vary; Δ from autoregressive H=4.

Variant	Avg
(a) No $\mathcal{L}_{roll}$	89.0 (−8.0)
(b) Single-step (H=1)	91.0 (−6.0)
(c) H=2	96.0 (−1.0)
(d) H=4 **(Ours)**	**97.0**
(e) H=8	94.0 (−3.0)
(f) H=4, teacher-forcing	92.0 (−5.0)

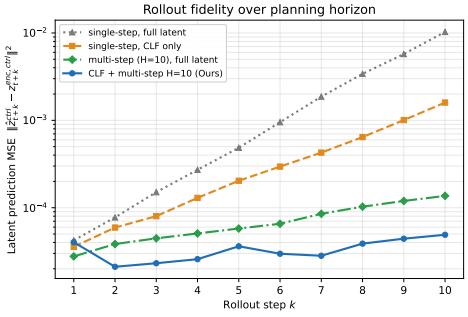


Figure 5: Latent rollout MSE vs. step k for the $H = 10$ diagnostic; default planning uses $H = 4$.

Closed-loop controls (Table 7) fix the gate rule, prior, budget, and progress supervision. Full scoring reaches $98.0 \pm 2.0\%$, versus $90.0 \pm 2.0\%$ for the direct absolute-progress critic, 86% for progress-only rollout, and 92% for consistency alone. Progress supplies a task-directed preference; consistency penalizes disagreement between dynamics models (Eq. (11)). The joint gain supports regularized prediction, since progress-only rollout underperforms the critic. EMA and ensemble both reach 98%, versus 92% for kNN density and 86% without a penalty (App. C), supporting consistency without demonstrating calibrated uncertainty or EMA superiority. These controllers can visit different states, unlike the shared-candidate comparison.

Deliberation allocation and computational budget. At 60% invocation, adaptive gating reaches $98.0 \pm 2.0\%$ at 289.3 ms, versus $92.0 \pm 2.0\%$ at 288.0 ms for random gating (Table 7). A separate paired study at a 60% target budget gives learned/heuristic success of $96.0\%/95.0\%$, a 1.0-point paired gap (App. H). The gates differ in inputs and feature normalization; this compares the two controllers without isolating learning alone. Always-on planning reaches 90% at 420 ms; adaptive gating reduces latency by 31%, while remaining $3.21\times$ the reactive baseline.

Invocation rises from 55% in transit to 84% during grasping over 200 trajectories (Figure 6); these post-hoc associations do not establish phase-specific benefit. With checkpoints fixed, $15/30/60\%$ invocation gives $92/96/98\%$ success at $141/190/289.3$ ms. In contrast, the 50-episode sweep shows no monotonic gain beyond $K = 5, H = 4$ (App. C). More candidates remain constrained by the prior; longer rollout can accumulate error. This motivates adjusting invocation separately from search width and depth.

5 Conclusion

ADEPT connects action-sensitive prediction with selective planning. Clean-only and same-state controls test representation, recursive training, and candidate ranking; controller and real-robot evaluations assess closed-loop utility. **Limitations.** Planning is constrained by prior support and $H = 4$. Demonstration progress can misrepresent failures or recovery, and action sensitivity does not ensure task sufficiency. Unnormalized action variance remains scale-sensitive; EMA disagreement is not calibrated uncertainty. The gate distills a scheduling heuristic. Selective invocation reduces average cost, while each invocation incurs candidate generation, rollout, and scoring.

AI USE STATEMENT

AI tools were used to assist with grammatical editing of the manuscript. The authors reviewed all revisions and take responsibility for the final manuscript.

REFERENCES

Jacob Berg, Chuning Zhu, Yanda Bao, Ishan Durugkar, and Abhishek Gupta. Semantic world models. *arXiv preprint arXiv:2510.19818*, 2025. URL https://arxiv.org/abs/2510.19818.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.

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. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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. *arXiv preprint arXiv:2506.21539*, 2025.

Akshay L. Chandra, Iman Nematollahi, Chenguang Huang, Tim Welschehold, Wolfram Burgard, and Abhinav Valada. Diwa: Diffusion policy adaptation with world models. *arXiv preprint arXiv:2508.03645*, 2025.

Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, Wenhao Tang, Zhiyuan Li, Weilin Liu, Zhuyu Yao, Boxun Li, Yuanbo Xu, and Chao Yu. LaWAM: Latent world action models for efficient dynamics-aware robot policies. *arXiv preprint arXiv:2606.15768*, 2026. URL https://arxiv.org/abs/2606.15768.

Inkook Chun, Seungjae Lee, Michael S. Albergo, Saining Xie, and Eric Vanden-Eijnden. Dynamic test-time compute scaling in control policy: Difficulty-aware stochastic interpolant policy. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

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. *arXiv preprint arXiv:2510.13626*, 2025.

Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning (ICML)*, 2021.

Zhenfeng Gan, ZiTong Zeng, Jiajun Cheng, Yeke Song, Yongyi Tang, and Xueqian Wang. ActSWM: Action-sensitive world models for long-horizon planning in open-world games. *arXiv preprint arXiv:2607.26712*, 2026. URL https://arxiv.org/abs/2607.26712v2.

Shenyuan Gao, Siyuan Zhou, Yilun Du, Jun Zhang, and Chuang Gan. Adaworld: Learning adaptable world models with latent actions. *arXiv preprint arXiv:2503.18938*, 2025.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.02199*, 2016.

Wenkai Guo, Guanxing Lu, Haoyuan Deng, Zhenyu Wu, Yansong Tang, and Ziwei Wang. Vla-reasoner: Empowering vision-language-action models with reasoning via online monte carlo tree search. *arXiv preprint arXiv:2509.22643*, 2025.

Yanjiang Guo, Tony Lee, Lucy Xiaoyang Shi, Jianyu Chen, Percy Liang, and Chelsea Finn. Vlaw: Iterative co-improvement of vision-language-action policy and world model. *arXiv preprint arXiv:2602.12063*, 2026.

Yuci Han and Alper Yilmaz. Enhancing policy learning with world-action model. *arXiv preprint arXiv:2603.28955*, 2026.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning (ICML)*, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*, 2024.

Carolina Higuera, Sergio Arnaud, Byron Boots, Mustafa Mukadam, Francois Robert Hogan, and Franziska Meier. Visuo-tactile world models. *arXiv preprint arXiv:2602.06001*, 2026. URL https://arxiv.org/abs/2602.06001.

Yuhang Huang, Jiazhao Zhang, Shilong Zou, Xinwang Liu, Ruizhen Hu, and Kai Xu. Ladi-wm: A latent diffusion-based world model for predictive manipulation. *arXiv preprint arXiv:2505.11528*, 2025.

Riashat Islam, Manan Tomar, Alex Lamb, Yonathan Efroni, Hongyu Zang, Aniket Rajiv Didolkar, Dipendra Misra, Xin Li, Harm Van Seijen, Remi Tachet Des Combes, and John Langford. Principled offline RL in the presence of rich exogenous information. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14390–14421, 2023. URL https://proceedings.mlr.press/v202/islam23a.html.

Dongwon Kim, Gawon Seo, Jinsung Lee, Minsu Cho, and Suha Kwak. Planning in 8 tokens: A compact discrete tokenizer for latent world model. *arXiv preprint arXiv:2603.05438*, 2026a.

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. In *Conference on Robot Learning (CoRL)*, 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.

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. *arXiv preprint arXiv:2601.16163*, 2026b.

Alexander S. Klyubin, Daniel Polani, and Chrystopher L. Nehaniv. Empowerment: A universal agent-centric measure of control. In *Proceedings of the 2005 IEEE Congress on Evolutionary Computation (CEC)*, volume 1, pp. 128–135. IEEE, 2005. doi: 10.1109/CEC.2005.1554676.

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. *arXiv preprint arXiv:2601.21998*, 2026a.

Wenhao Li, Xiu Su, Yichao Cao, Hongyan Xu, Xiaobo Xia, Shan You, Yi Chen, and Chang Xu. VLA-ATTC: Adaptive test-time compute for VLA models with relative action critic model. *arXiv preprint arXiv:2605.01194*, 2026b. URL https://arxiv.org/abs/2605.01194v2.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

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 (NeurIPS)*, 2023.

Yuanzhe Liu, Jingyuan Zhu, Yuchen Mo, Gen Li, Xu Cao, Jin Jin, Yifan Shen, Zhengyuan Li, Tianjiao Yu, Wenzhen Yuan, Fangqiang Ding, and Ismini Lourentzou. Palm: Progress-aware policy learning via affordance reasoning for long-horizon robotic manipulation. *arXiv preprint arXiv:2601.07060*, 2026a.

Yushan Liu, Peibo Sun, Xintao Chao, Zhenyang Yang, Yifan Xie, Lingfeng Zhang, Shoujie Li, Chenyu Tang, Fang Chen, Xiao-Ping Zhang, and Wenbo Ding. CheckVLA: Execution-time verification with action-conditioned world model for long-horizon mobile manipulation. *arXiv preprint arXiv:2607.26789*, 2026b. URL https://arxiv.org/abs/2607.26789.

Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-H0.7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026. URL https://arxiv.org/abs/2605.00078.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, pp. 2125–2133, 2015.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations (ICLR)*, 2026.

NVIDIA. NVIDIA Isaac GR00T N1.6. https://research.nvidia.com/labs/gear/gr00t-n1_6/, 2026. NVIDIA GEAR Lab. Accessed: 2026-05-20.

NVIDIA, Johan Bjorck, et al. GR00T N1.5: An open foundation model for generalist robots. https://research.nvidia.com/labs/gear/gr00t-n1_5/, 2025. NVIDIA Technical Report.

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

RoboCasa Team. RoboCasa365 leaderboard. https://robocasa.ai/leaderboard.html, 2026. Updated 2026-05-19; accessed 2026-05-20.

Gawon Seo, Dongwon Kim, and Suha Kwak. ACID: Action consistency via inverse dynamics for planning with world models. *arXiv preprint arXiv:2607.02403*, 2026. URL https://arxiv.org/abs/2607.02403.

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*, 2025.

Lei Xiao et al. AVA-VLA: Improving vision-language-action models with active visual attention. *arXiv preprint arXiv:2511.18960*, 2025.

Hongyu Yan, Qiwei Li, Jiaolong Yang, and Yadong Mu. Progressvla: Progress-guided diffusion policy for vision-language robotic manipulation. *arXiv preprint arXiv:2603.27670*, 2026.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, Feng Xiong, Xing Wei, Zhiheng Ma, and Mu Xu. ABot-M0: VLA foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026. URL https://arxiv.org/abs/2602.11236.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, Min Cao, Peng Li, Qiuping Deng, Wenjun Mei, Xiaofeng Wang, Xinze Chen, Xinyu Zhou, Yang Wang, Yifan Chang, Yifan Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Jim Fan, and Joel Jang. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026b.

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.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations (ICLR)*, 2021.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, Hongzhu Yi, Guoqing Chao, Xingchen Chen, Tiankun Yang, Chenxi Bao, Tao Yu, Jingjing Zhou, and Jungang Xu. Delta-JEPA: Learning action-sensitive world models via latent difference decoding. *arXiv preprint arXiv:2606.31232*, 2026. URL https://arxiv.org/abs/2606.31232.

Jinliang Zheng, Jianxiong Li, et al. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025a.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, Avnish Narayan, You Liang Tan, Guanzhi Wang, Qi Wang, Jiannan Xiang, Yinzhen Xu, Seonghyeon Ye, Jan Kautz, Furong Huang, Yuke Zhu, and Linxi Fan. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025b.

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. ACoT-VLA: Action chain-of-thought for vision-language-action models. *arXiv preprint arXiv:2601.11404*, 2026. URL https://arxiv.org/abs/2601.11404v2. CVPR 2026.

Pengfei Zhou, Shengcong Chen, Di Chen, Jiaxu Wang, Rongjun Jin, Bingwen Zhu, Yike Pan, Songen Gu, Kuanning Wang, Shufeng Nan, et al. τ_0-WM: A unified video-action world model for robotic manipulation. *arXiv preprint arXiv:2606.01027*, 2026. URL https://arxiv.org/abs/2606.01027.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

A Interpretation of the Action-Variance Surrogate

This appendix clarifies the relation between the conceptual action-influence quantity in Eq. (4) and the tractable objective in Eq. (5). The latter encourages action-dependent predictive variation; it does not establish causal identification, task sufficiency, or an injective mapping from actions to latent outcomes. The representation remains an action-influence-motivated inductive bias whose utility is evaluated empirically.

A.1 Conditional Mixture and Gaussian Reference

Fix the complete conditioning context $c = (\mathbf{o}_{\leq t}, l)$, and let $z = z_t^{ctrl}$ be its masked latent state. Let $A \sim \pi(\cdot \mid c)$ be an observational draw from the frozen action prior. All expectations and distributions below are conditional on this same c, with the conditioning suppressed where unambiguous. Write

$$p_a(z') = \mathcal{N}(\mu_a, \sigma^2 I), \qquad \mu_a = f_{dyn}(z, a), \qquad p_{\mathrm{marg}}(z') = \mathbb{E}_A[p_A(z')], \tag{13}$$

with $\sigma > 0$ and finite $\mathbb{E}_A\|\mu_A\|^2$. The marginal is generally a Gaussian mixture, even though every action-conditioned component is Gaussian. Define $\bar{\mu} = \mathbb{E}_A[\mu_A]$ and a reference $q_{ref} = \mathcal{N}(\bar{\mu}, \sigma^2 I)$. This reference matches the mean but does not match the marginal covariance:

$$\mathrm{Cov}_{p_{\mathrm{marg}}}(Z') = \sigma^2 I + \mathrm{Cov}_A(\mu_A). \tag{14}$$

Thus a mean-matched Gaussian with covariance $\sigma^2 I$ cannot be substituted for the exact marginal in the KL divergence without changing the objective.

A.2 What the Surrogate Measures

Proposition A.1 (Gaussian-reference decomposition)**.** *At each fixed context c, the expected KL divergence to q_{ref} satisfies*

$$\mathbb{E}_A D_{\mathrm{KL}}(p_A \| q_{ref}) = \frac{\mathrm{tr}\,\mathrm{Cov}_A(\mu_A)}{2\sigma^2}, \tag{15}$$

$$= I(A; Z' \mid c) + D_{\mathrm{KL}}(p_{\mathrm{marg}} \| q_{ref}). \tag{16}$$

Here $I(A; Z' \mid c) = \mathbb{E}_A D_{\mathrm{KL}}(p_A \| p_{\mathrm{marg}})$ is computed under the policy-induced joint distribution at fixed c. Conditioning only on z would not fix the action distribution, since the prior also observes information outside the masked latent.

Proof. Equal-covariance Gaussian KL gives $D_{\mathrm{KL}}(p_a \| q_{ref}) = \|\mu_a - \bar{\mu}\|^2/(2\sigma^2)$. Averaging yields Eq. (15). For the second identity, split the log ratio as $\log(p_a/q_{ref}) = \log(p_a/p_{\mathrm{marg}}) + \log(p_{\mathrm{marg}}/q_{ref})$. Integrating against p_a and averaging over actions leaves the first term as mutual information and the second as $D_{\mathrm{KL}}(p_{\mathrm{marg}} \| q_{ref})$. □

The additional marginal-to-reference term depends on the learned representation. Consequently, maximizing between-action variance is not equivalent to maximizing the exact mixture-marginal KL or conditional mutual information. It can also increase through predictive scale or task-irrelevant action-dependent variation. The rollout loss anchors predictions to encoded observations, but does not establish task relevance.

A.3 Prediction Scale and a Bounded Alternative

For predictions $f_c(z, a) = c f_{dyn}(z, a)$, the isolated variance loss satisfies $\mathcal{L}_{ai}[f_c] = c^2 \mathcal{L}_{ai}[f_{dyn}]$. Thus, positive between-action variation makes this term unbounded below as c increases. The rollout loss constrains predictions on supervised demonstration transitions; it does not by itself bound all candidate actions. A bounded alternative can be defined in normalized coordinates without identifying it with the unnormalized objective used in the benchmark tables. For $M \geq 2$ independent policy actions at fixed context, let

$$u_j = \frac{f_{dyn}(z, a^{(j)})}{\max(\epsilon, \|f_{dyn}(z, a^{(j)})\|_2)}, \qquad \bar{u} = \frac{1}{M}\sum_j u_j, \qquad \mathcal{L}_{ai}^{unit} = -\frac{1}{2\sigma^2 M}\sum_j \|u_j - \bar{u}\|_2^2, \tag{17}$$

where $\epsilon > 0$ and $\sigma > 0$.

Proposition A.2 (Bounded normalized separation)**.** *For every finite set of predictions,* $-1/(2\sigma^2) \leq \mathcal{L}_{ai}^{unit} \leq 0$.

Proof. Since $\|u_j\|_2 \leq 1$, $M^{-1}\sum_j \|u_j - \bar{u}\|_2^2 = M^{-1}\sum_j \|u_j\|_2^2 - \|\bar{u}\|_2^2 \in [0,1]$. Multiplication by $-1/(2\sigma^2)$ gives the claimed interval. □

This removes unbounded improvement by enlarging prediction scale, but retains the need to test task relevance. With the same M draws used for both the mean and deviations, the expected centered second moment is $(M-1)\,\mathrm{tr}\,\mathrm{Cov}(U)/M$; it is a finite-sample variance surrogate rather than an unbiased covariance estimator.

Scale in the joint objective. An unbounded separation term does not by itself establish that the joint objective is unbounded. Fix the encoder, mask, demonstration transitions, and sampled actions, and consider $f_c = cf_{dyn}$ with $c > 0$. For the first supervised rollout step, write $f = f_{dyn}(z, a)$ and $y = z_{t+1}^{enc,ctrl}$. Define $S = H^{-1}\mathbb{E}_{\mathcal{D}}\|f\|^2$, $R = H^{-1}\mathbb{E}_{\mathcal{D}}\langle f, y\rangle$, and $T = H^{-1}\mathbb{E}_{\mathcal{D}}\|y\|^2$. Let V be the unscaled empirical centered second moment used in $\mathcal{L}_{ai}$. The first-step rollout contribution and the weighted separation term then satisfy

$$\ell_1(c) = \frac{1}{H}\mathbb{E}_{\mathcal{D}}\|cf - y\|^2 + 0.02\mathcal{L}_{ai}[cf] = \left(S - \frac{0.02V}{2\sigma^2}\right)c^2 - 2Rc + T.$$

Thus $S > 0.02V/(2\sigma^2)$ is sufficient to prevent unbounded improvement along this fixed scaling direction, since the remaining rollout terms are nonnegative and the inverse and mask terms are fixed. This condition compares demonstration prediction energy with candidate-action variation; frozen encoder targets alone do not ensure it when the action supports differ. For $H > 1$, scaling changes recursive inputs, so the complete rollout loss is not generally quadratic in c, and a negative coefficient above does not establish joint divergence. The reported ramp-up and gradient clipping regulate optimization but do not impose an output-norm bound. The normalized objective has a scale-independent bound; neither that bound nor this directional analysis certifies stability of the unnormalized benchmark objective over all parameters.

A.4 Scope of the Representation Claim

A low inverse-dynamics loss encourages consecutive latents to retain information about the executed action on the training distribution. Even zero inverse loss on observed transitions does not prove that all candidate-action consequences are distinguishable, nor that static goals, obstacles, and task relations needed for planning are retained. Positive or large between-action variance also permits multiple distinct actions to share an outcome; increasing variance alone does not enforce injectivity.

Bisimulation representations (Zhang et al., 2021) define relevance through behavioral and reward equivalence, whereas the present objectives use action predictability and a mask budget. These are different biases, without a proved inclusion relationship. In particular, task-critical variables may be weakly action-sensitive, while irrelevant variables may respond strongly to action. ADEPT's action prior proposes candidates from full observations; its latent dynamics predict their short-horizon consequences for action selection. This architecture and the reported ablations provide evidence of utility, not a causal or task-sufficiency guarantee.

Finally, the soft mask reweights the original token set. Its average activation budget does not by itself reduce token count or imply proportional FLOP savings. Realized computation savings depend on the implemented dynamics and execution path.

B Implementation Details

B.1 Architecture Specifications

ADEPT uses a pretrained $\pi_{0.5}$ vision-language-action policy (Physical Intelligence et al., 2025) as its action prior π_θ. The action prior and visual encoder remain frozen while the mask, latent dynamics, and prediction/decision heads are trained. The gating function q_ω is a two-layer MLP that consumes per-token visual features concatenated with a pooled language embedding and outputs a

soft scalar mask. The latent dynamics f_{dyn} is an L-layer Transformer operating on the control tokens, conditioned on action embeddings via FiLM modulation. The task value head V_{task} is a lightweight MLP; the deliberation gate uses an affine map followed by a sigmoid, as in Eq. (10).

Tensor and planning units. The soft mask preserves N tokens of width d, so the predictive state has shape $N \times d$. Pooling maps it to $\mathbb{R}^d$; the five scalar gate features give $\phi_t \in \mathbb{R}^{d+5}$ and one sigmoid output. The mask MLP produces one scalar per token, while the value head produces one progress score. L denotes Transformer depth; M counts training-time reference actions, whereas $K = 5$ counts deployment candidate sequences. Each sequence contains $H = 4$ rollout chunks. These quantities have different roles: H does not specify the number of low-level commands within a chunk, and M is not identified with K.

B.2 Training Setup

We train using the AdamW optimizer with gradient-norm clipping at 1.0. The learning rate follows a cosine-decay schedule with a peak of 5×10^{-5} and $10{,}000$ warmup steps; we use a global batch size of 256 with 2-step gradient accumulation (effective batch 512) and train for $30{,}000$ steps on 8 A100 GPUs. The factorization loss coefficients are λ_{ai}=0.02, λ_{inv}=0.1, and λ_{mask}=0.05 (with mask-entropy weight λ_{ent}=0.01), the candidate-score consistency weight is λ_{rc}=0.5, and the mask budget is ρ=0.25; the dynamics and factorization loss terms are linearly ramped over the first $5{,}000$ steps. The rollout horizon used for $\mathcal{L}_{roll}$ matches the test-time lookahead horizon H=4.

Objective and parameter blocks. Writing the inverse-dynamics parameters as ψ, mask parameters as ω, and transition parameters as ϕ, the representation objective is

$$\mathcal{L}_{repr} = \mathcal{L}_{roll} + 0.1\mathcal{L}_{inv} + 0.02\mathcal{L}_{ai} + 0.05\mathcal{L}_{mask}. \tag{18}$$

The parameter dependencies of these terms are (ω, ϕ) for rollout and action-variance losses, (ω, ψ) for inverse dynamics, and ω for the mask regularizer. The action prior and visual encoder are fixed. Progress and scheduling heads minimize $\mathcal{L}_{value}$ and $\mathcal{L}_{gate}$ on detached inputs, separately from $\mathcal{L}_{repr}$. EMA parameters follow a moving average rather than gradient descent. The variance reference uses $M = 8$ policy draws and $\sigma = 1.0$, so the effective weight of predictive separation is $0.02/(2\sigma^2) = 0.01$; σ is a scale parameter, not an uncertainty estimate. The same M draws form the reference mean and centered deviations. Squared distances sum over latent scalars; $\mathcal{L}_{ai}$ averages over draws and batch elements, while $\mathcal{L}_{roll}$ averages over batch elements and the $H = 4$ rollout steps. The M reference draws and $K = 5$ deployment candidates have distinct roles (App. H).

Data provenance and evaluation tiers. The clean LIBERO setting uses only standard LIBERO training demonstrations. The clean LIBERO-Plus evaluation reuses the pre-existing standard-LIBERO checkpoint, trained, tuned, and selected without LIBERO-Plus perturbations, and evaluates all $10{,}030$ variants. The augmented setting additionally trains the mask, latent dynamics, and heads on the released generalized split covering six perturbation families; this does not update the frozen $\pi_{0.5}$ backbone and includes no robot-initial-state shift. Its results concern held-out configurations of six seen families plus one unseen family, rather than uniformly unseen shift families. Neither setting uses evaluation variants, rollouts, scores, failure labels, or success labels to train or tune the system. RoboCasa uses Human300 demonstrations. Real-robot supervision comes from offline teleoperation demonstrations; where applicable these adapt the base policy before it is frozen, while Task 3 uses no task-specific base-policy adaptation.

Progress target and gradients. For a demonstration of length T with zero-based index u, V_{task} regresses timestamp progress $y_u = (u+1)/T$ at encoded states $z_u^{enc,ctrl}$; rollout-terminal alignment uses $u = t + H < T$. All experiments use this target. It is not a discounted benchmark return or a calibrated success probability. At deployment the same head scores predicted terminal latents, so generalization from observed demonstration states to imagined states is a limitation. Progress need not represent recoverability in stalled, failed, or reversing trajectories. Value and gate losses use stop-gradient latents and update only their own heads, never the encoder, mask, or dynamics. Only representation/dynamics learning is reward-free; the planner and gate use demonstration progress supervision.

Gate target and reference. The gate parameters $\{\mathbf{w}, b\}$ are trained with binary cross-entropy against

$$\tilde{g}_t = \sigma\big(z_{inv} + 0.5 z_{ref} + 0.5 z_{bg} - z_{val}\big), \tag{19}$$

where the per-batch standardized training-target terms correspond to $\epsilon_{inv}^{(t)}$, $\Delta_{ref}^{(t)}$, $\Delta_{bg}^{(t)}$, and $V_{task}(z_t^{ctrl}, l)$. This is a heuristic soft target, not a measured label of the benefit from deliberation. Write $\bar{z}_t^{hist}$ for the reference available before decision t, and $\hat{z}_{t+1}^{tent} = f_{dyn}(z_t^{ctrl}, a_t)$ for the one-step prediction under the tentative action in Algorithm 1. The reference used by the current gate is read before its update for the next decision:

$$\bar{z}_{t+1}^{hist} = \kappa \bar{z}_t^{hist} + (1-\kappa)\hat{z}_{t+1}^{tent}, \qquad \kappa = 0.99. \tag{20}$$

This update reuses the tentative-action roll-in even when deliberation selects a different executed chunk; it is a prediction-history reference. It is distinct from the dynamics-parameter EMA used in candidate scoring. Because this average mixes states and times, it is not an estimate of an action marginal at fixed current state. We therefore denote the feature $\Delta_{ref}^{(t)} = \|\hat{z}_{t+1}^{tent} - \bar{z}_t^{hist}\|_2$ and interpret it as contrast to a cached historical reference, distinct from $\bar{\mu}_t$ in Eq. (5). It reuses the one-step roll-in and adds no dynamics call for the reference contrast itself.

Tuning and checkpoints. A single gradient-scale-matched loss-weight configuration is reused. The threshold is calibrated once using held-out training data; K and H are selected on clean LIBERO-Spatial. The final 30,000-step ADEPT checkpoints remain fixed for the deployment-budget sweep, which changes the operating threshold rather than retraining models.

B.3 Evaluation Protocol

Evaluation units and comparisons. Our main standard-LIBERO evaluation uses 50 trials per task and three seeds; comparison entries for other methods follow their cited reports. The focused Spatial controller study (Table 7) uses 50 episodes per seed over three shared seeds; the planning-budget sweep uses 50 episodes per configuration. The $\pi_{0.5}$ Spatial score of 97.4% in Table 1 is a reported benchmark comparison, whereas the 88.0% prior-only score in Table 7 is our direct-execution control in the focused study. Controller gains are computed within each study; absolute scores and uncertainty estimates are not pooled across protocols. The focused study reports mean and sample standard deviation over seeds, not a confidence interval for the complete task distribution.

The corresponding ADEPT Spatial results also identify different studies: 100.0% in the main benchmark (Table 1), 98.0% in the three-seed controller comparison (Table 7), 97.0% in the rollout-supervision diagnostic (Table 9), and 94.0% at $K = 5, H = 4$ in the 50-episode planning sweep (Table 11). The 97.5% Full result in Table 8 is instead a four-suite mean under its replan$= 5$, $n = 200$ objective-ablation protocol. These summaries retain their own evaluation units and are compared only with controls from the same study.

Clean LIBERO-Plus evaluates all 10,030 variants. Macro-7 equally averages the seven factor-family rates, giving 88.1% for ADEPT-clean and 90.2% for ADEPT-aug. OpenVLA-OFT+'s 80.7% is recomputed on this same basis; its published 79.6% instance-weighted aggregate uses a different denominator. RoboCasa split-macro equally averages the three displayed split rates for every method. Each real-robot method uses 50 fully reset trials per task, with task-wise Wilson intervals. Candidate-ranking metrics use shared restored-state executions and are reported as paired full-minus-control differences in App. H; the Spatial seed deviations and real-robot intervals do not quantify their uncertainty.

Execution paths in the ablations. The w/o ADV variant in Table 10 removes selective invocation and the consistency penalty but retains multi-candidate latent rollout and progress-based selection. It can therefore differ from the frozen prior. Removing factorization likewise retains planning, now using full latents. In contrast, prior-only execution in Table 7 bypasses candidate evaluation. With the same checkpoint, observation preprocessing, action interface, sampling, and execution settings, this path implements direct execution of the action prior. Freezing parameters alone does not make these three controllers equivalent.

Algorithm 1 ADEPT: Adaptive Deliberation for Action Selection

Require: Frozen action prior π_θ, dynamics model f_{dyn}, EMA dynamics model f_{dyn}^{ema}, value head V_{task}, inverse-dynamics head g_ψ, cached historical reference $\bar{z}^{hist}$ and background-drift feature $\Delta_{bg}^{(t)}$, gate parameters $\{\mathbf{w}, b, \tau\}$, candidates K, horizon H, consistency weight λ_{rc}

Encode current observation: $z_t^{ctrl} \leftarrow \mathrm{Encode\&Mask}(\mathbf{o}_{\le t}, l)$
Sample tentative action from the action prior: $a_t \sim \pi_\theta(\cdot \mid \mathbf{o}_{\le t}, l)$
One-step roll-in: $\hat{z}_{t+1}^{ctrl} \leftarrow f_{dyn}(z_t^{ctrl}, a_t)$
Compute model features using the roll-in and available history:
$\epsilon_{inv} \leftarrow \|g_\psi(z_t^{ctrl}, \hat{z}_{t+1}^{ctrl}) - a_t\|_2^2$ {forward–inverse consistency}
$\Delta_{ref} \leftarrow \|\hat{z}_{t+1}^{ctrl} - \bar{z}^{hist}\|_2$ {historical-reference contrast}
$\rho_{ctrl} \leftarrow \|\hat{z}_{t+1}^{ctrl} - z_t^{ctrl}\|_2$ {control-latent displacement}
Compute progress prediction at current state: $V_t \leftarrow V_{task}(z_t^{ctrl}, l)$
Compute gate: $g_t \leftarrow \sigma\big(\mathbf{w}^\top[\mathrm{pool}(z_t^{ctrl}),\, V_t,\, \epsilon_{inv},\, \Delta_{ref},\, \Delta_{bg},\, \rho_{ctrl}] + b\big)$
if $g_t \le \tau$ **then**
  Fast path: execute a_t directly {no deliberation}
else
  Deliberate: sample K action sequences $\{A^{(k)}\}_{k=1}^K \sim \pi_\theta$, with $A^{(k)} = (a_t^{(k)}, \ldots, a_{t+H-1}^{(k)})$
  for $k = 1, \ldots, K$ **do**
    $\hat{z}_t^{ctrl,(k)} \leftarrow z_t^{ctrl}$
    for $h = 1, \ldots, H$ **do**
      $\hat{z}_{t+h}^{ctrl,(k)} \leftarrow f_{dyn}\Big(\hat{z}_{t+h-1}^{ctrl,(k)},\, a_{t+h-1}^{(k)}\Big)$
    end for
    $V^{(k)} \leftarrow V_{task}\Big(\hat{z}_{t+H}^{ctrl,(k)},\, l\Big)$ {terminal progress prediction}
    For each h, let $\tilde{z}_{t+h}^{(k)} = f_{dyn}^{ema}(\hat{z}_{t+h-1}^{ctrl,(k)}, a_{t+h-1}^{(k)})$.
    $C^{(k)} \leftarrow \frac{1}{H}\sum_{h=1}^{H} \|\hat{z}_{t+h}^{ctrl,(k)} - \tilde{z}_{t+h}^{(k)}\|^2$ {consistency penalty}
    $s^{(k)} \leftarrow V^{(k)} - \lambda_{rc}\, C^{(k)}$ {rollout-aware score}
  end for
  Select $k^\star = \arg\max_k s^{(k)}$; execute the first chunk of $A_t^\star = A^{(k^\star)}$.
end if
Cache the tentative roll-in for the next decision: $\bar{z}^{hist} \leftarrow \kappa \bar{z}^{hist} + (1-\kappa)\hat{z}_{t+1}^{ctrl}$, $\kappa = 0.99$.

B.4 Full Test-Time Procedure

Algorithm 1 summarizes candidate generation, control-latent rollout, scoring, and receding-horizon action selection. The gate uses the current latent state, predicted progress, forward–inverse consistency, historical-reference contrast, background drift, and control-latent displacement. Gate inputs are available at decision time; the actual next observation is unavailable before action execution. The K-candidate, H-step evaluation branch runs when the gate activates. Prior-only execution bypasses the evaluator entirely; an inactive gate is obtained only after computing the one-step gate features. Neither is the always-on, progress-only variant called w/o ADV in Table 10.

Remarks. EMA disagreement is a relative model-consistency penalty, not calibrated rollout uncertainty. Replacing the EMA model with a stop-gradient copy of the current model would give identical numerical predictions and hence zero discrepancy; stop-gradient changes derivatives, not forward values. During training, $\phi^{ema} \leftarrow \beta\phi^{ema} + (1-\beta)\phi$ with $\beta = 0.99$ after each dynamics optimizer step; this is separate from the decision-time history update. Receding-horizon execution commits only the first chunk, then repeats prediction and selection on the next available observation. It mitigates myopia through repeated replanning; it does not model task structure beyond the rollout window.

C Additional Ablations and Hyperparameter Studies

C.1 Supporting Component and Horizon Controls

The main clean-only attribution in Table 5 fixes deployment depth. Table 10 instead reports the original standard-LIBERO and augmented LIBERO-Plus controls. Its no-MRT variant changes both training and deployment to $H = 1$, whereas its no-ADV variant retains progress-only rollout and removes both selective invocation and consistency scoring. The gate-only and penalty-only rows separate these two changes. These comparisons are interpreted within their own training protocol.

Table 10: Ablations of CLF (control-oriented latent factorization), MRT (multi-step rollout training), and ADV (adaptive gate + rollout-aware value). Removing CLF rolls out full latents; removing MRT uses single-step dynamics. *Partial* removes one ADV sub-mechanism. The LIBERO-plus column uses augmented training.

Variant	CLF	MRT	ADV	LIBERO Avg	LIBERO-plus Avg
Base ($\pi_{0.5}$, no world model)	×	×	×	96.85	71.5
w/o CLF (rollout on full latent)	×	✓	✓	95.5 (−3.6)	79.4 (−10.8)
w/o MRT (single-step training, H=1)	✓	×	✓	96.4 (−2.7)	82.8 (−7.4)
w/o ADV (always-on, task-value only)	✓	✓	×	97.0 (−2.1)	84.6 (−5.6)
ADV sub-ablations (CLF and MRT kept):					
w/o deliberation gate only	✓	✓	*partial*	98.2 (−0.9)	87.5 (−2.7)
w/o consistency penalty (λ_{rc}=0)	✓	✓	*partial*	97.6 (−1.5)	86.0 (−4.2)
Full (Ours)	✓	✓	✓	**99.1**	**90.2**

Tables 8 and 9 report the objective and training-horizon diagnostics in the main text. The objective study uses a separate $n = 200$ protocol; its 97.5% Full result is not the main benchmark aggregate. In the Spatial horizon study, recursive $H = 4$ training reaches 97.0% versus 92.0% for teacher forcing at the same horizon. The clean-transfer control in Table 5 additionally holds the deployment window fixed when comparing single-step training.

C.2 Planning Hyperparameters

Table 11 sweeps the number of candidate action sequences K and planning horizon H on LIBERO-Spatial. At $H = 4$, $K = 1/3/5/10/20$ gives $90/92/94/92/92\%$. At $K = 5$, $H = 1/3/4/5/10$ gives $92/90/94/90/94\%$. These small-sample point estimates show no monotonic gain beyond the selected $K = 5, H = 4$ configuration. They do not establish equivalence or statistical significance between configurations. The default horizon matches rollout training; longer horizons are outside that training depth and can accumulate model error. Candidate sequences are observational samples from the frozen policy, so increasing K cannot guarantee recovery when useful actions lie outside its support.

Table 11: Success rate (%) on LIBERO-Spatial under varying planning budget. Each cell uses five trials over ten tasks (50 episodes total). Bold marks the adopted $K = 5, H = 4$ setting, selected on clean LIBERO-Spatial and matched to the rollout-training horizon. Small differences are descriptive; no universal binomial significance threshold is assumed.

H \ K	1	3	5	10	20
1	88.0	92.0	92.0	88.0	90.0
3	82.0	90.0	90.0	90.0	90.0
4	90.0	92.0	**94.0**	92.0	92.0
5	90.0	92.0	90.0	92.0	92.0
10	88.0	86.0	94.0	94.0	90.0

Table 12: Per-objective ablation of control-oriented latent factorization on standard LIBERO and augmented LIBERO-Plus (Macro-7). Rows share the frozen backbone, encoder, and benchmark-specific training data. The evaluation uses replan$= 5$ and $n = 200$. Deltas are descriptive differences from Full, without a fixed significance cutoff.

Variant	LIBERO Avg	LIBERO-plus Avg
Full (Ours)	**97.5**	**90.2**
w/o $\mathcal{L}_{inv}$	95.0 (-2.5)	83.0 (-7.2)
w/o $\mathcal{L}_{ai}$	94.5 (-3.0)	81.5 (-8.7)
w/o $\mathcal{L}_{mask}$	92.0 (-5.5)	76.0 (-14.2)
Bisim. mask (Zhang et al., 2021)	93.5 (-4.0)	79.0 (-11.2)
w/o factoriz.	90.0 (-7.5)	73.5 (-16.7)

Table 13: Three-seed variance for the adaptive-deliberation and rollout-aware-value ablation (LIBERO-Spatial, n=50 per seed; shared seeds across rows). Mean $\pm$ std over seeds; The last column indicates whether the sign of the per-seed gap to Full is stable; no statistical test is implied.

Variant	S1	S2	S3	Mean$\pm$std	Gap sign stable
No delib. (prior only)	86.0	88.0	90.0	88.0$\pm$2.0	✓
Always delib. (no gate)	88.0	90.0	92.0	90.0$\pm$2.0	✓
Random gate (matched)	90.0	92.0	94.0	92.0$\pm$2.0	✓
Gate w/o V_{task}	90.0	92.0	94.0	92.0$\pm$2.0	✓
Gate w/o disagr. inputs	90.0	92.0	94.0	92.0$\pm$2.0	✓
Task value only (λ_{rc}=0)	84.0	86.0	88.0	86.0$\pm$2.0	✓
Direct critic (no rollout)	88.0	90.0	92.0	90.0$\pm$2.0	✓
Full (Ours)	**96.0**	**98.0**	**100.0**	**98.0$\pm$2.0**	—

C.3 Factorization Objectives under Distribution Shift

The factorization objectives are evaluated on standard LIBERO and the augmented LIBERO-Plus setting in Table 12. All rows within each benchmark share the training data and backbone, so their within-column differences isolate objective changes under that protocol. These ablations use the augmented LIBERO-Plus protocol. Near-saturated standard-LIBERO scores show smaller point-estimate differences than the distribution-shift evaluation.

Removing $\mathcal{L}_{ai}$ or $\mathcal{L}_{inv}$ reduces augmented LIBERO-Plus success by 8.7 and 7.2 percentage points, respectively. Removing the mask objective or factorization yields larger decreases of 14.2 and 16.7 points. The bisimulation-mask baseline is 11.2 points lower. These results support the utility of the combined factorization objectives in this setting; they do not establish causal identification, task sufficiency, or that every pairwise difference is statistically significant. The aggregate levels are not interchangeable with the main benchmark or other ablation protocols.

C.4 Seed Variance of the Deliberation Ablation

Table 13 reports the three-seed adaptive-deliberation study on LIBERO-Spatial, with 50 episodes per seed. All rows share seed IDs; ADEPT exceeds the frequency-matched random gate by six percentage points on each seed. This checks consistency across the reported seeds. A shared seed does not itself establish episode-wise pairing, and these aggregate scores do not support a universal six-point significance criterion. Formal paired testing requires the corresponding episode outcomes and task/seed structure.

The ranking is consistent across the three reported seeds. The result supports selective allocation relative to the matched random gate under this protocol, without proving that each triggered decision has positive causal benefit. The no-penalty configuration is lower on every seed, supporting the utility of the consistency term. Three-seed standard deviations describe these runs and are not confidence intervals for the full task distribution.

C.5 Alternative Consistency Signals

We compare EMA disagreement with an ensemble and a k-nearest-neighbor (kNN) density signal under matched LIBERO-Spatial evaluation ($n = 50$). EMA and ensemble achieve 98%, kNN 92%, and no penalty 86%. These point estimates support the EMA consistency signal; they do not demonstrate calibrated uncertainty or superiority over ensembles. Reconstruction against the observed future cannot score an unexecuted candidate at decision time, whereas ensembles and density estimates can be applied without that future.

C.6 Direct Action-Sequence Critic

A direct critic scores a candidate from the current control latent and the action sequence itself, without latent rollout: $V_{direct}(z_t^{ctrl}, a_{t:t+H-1}, l)$ is a lightweight MLP over the pooled control latent, the language embedding, and a step-embedded action sequence. It is trained with the demonstration-progress label and stop-gradient isolation of V_{task} (App. B), with the backbone, dynamics, and all other heads frozen. Deployment changes the candidate scorer while retaining the gate, candidate budget K, and frozen action prior; the direct critic uses neither candidate rollout nor an EMA-consistency term. Under the shared three-seed protocol of Table 13, the direct critic reaches $90.0 \pm 2.0\%$ versus $98.0 \pm 2.0\%$ for rollout-aware scoring, within the band of the task-value-only and matched random-gate configurations (86.0–92.0, Table 7). The full rollout-and-consistency scorer therefore outperforms this direct critic, but progress-only rollout reaches 86.0%, below the direct critic's 90.0%. Latent prediction alone does not account for the gain: the results support its joint use with consistency-aware scoring. The comparison holds supervision and the proposal mechanism fixed while changing both prediction and regularization. It does not isolate rollout alone. The lightweight critic uses the same prior, candidate budget, gate rule, and hardware; unlike VLA-ATTC (Li et al., 2026b), it learns absolute progress rather than pairwise preferences and uses pooled masked features rather than the full observation context. It therefore does not reproduce VLA-ATTC's relative critic or exclude stronger direct-scoring architectures. Identical settings do not imply identical visited states or candidate arrays after the controllers diverge. Skipping the K-candidate rollout lowers the per-step cost to 244.6 ms ($2.72\times$) at the same 60% activation.

Absolute and relative comparisons. Table 7 changes the scorer inside a complete closed-loop controller; its absolute-progress MLP receives pooled control features, language, and embedded actions, with demonstration-timestamp supervision. Table 6 instead evaluates absolute and relative critics on the same saved candidate arrays and outcome labels. Relative-critic accuracy uses pairwise preferences, while its selected-candidate outcome uses tournament selection. Sharing candidate support controls which actions can be selected; it does not by itself match scorer capacity or training supervision. This diagnostic is therefore a comparison with the evaluated critics, not a reproduction of VLA-ATTC's complete controller or a general claim against direct scoring.

Here "no rollout" refers to removal of the multi-candidate, H-step rollout; the retained gate still computes its tentative-action one-step features. It is therefore a control for candidate scoring, not a controller with no dynamics computation.

D Efficiency and Deployment Analysis

D.1 Per-Step Latency Breakdown

Table 14 reports the per-step cost decomposition. Full deliberation costs $420\,\mathrm{ms}$ on the reported LIBERO-Spatial/A100 profiling setup. At approximately 60% activation, mean latency is $289.3\,\mathrm{ms}$, about $1.45\times$ faster than always-on deliberation and about $3.21\times$ the $90\,\mathrm{ms}$ reactive latency. The inverse latency is a replanning rate (approximately 3.5 replans/s), not the low-level robot control frequency. Candidate sampling contributes substantially to the deliberate branch.

The fixed-checkpoint deployment sweep reports activation/success/latency as $15\%/92\%/141\,\mathrm{ms}$, $30\%/96\%/190\,\mathrm{ms}$, and $60\%/98\%/289.3\,\mathrm{ms}$. Prior-only execution without deliberation gives 88% at $90\,\mathrm{ms}$. Thus the default is one accuracy–latency operating point, not a claim of negligible or universally acceptable overhead. The learned gate can activate during transit; phase statistics alone do not identify where the selected action changes or how much deliberation benefits each state.

Table 14: Per-step latency breakdown in ms. ADEPT uses the reported single-A100 LIBERO-Spatial profiling setup. "Prior only" disables the evaluator; "worst case" forces full deliberation; "adaptive" uses the default threshold. Rates are replanning rates, not control-loop frequencies. External model totals are cross-paper reference values with different profiling configurations.

Method	Params	Enc	VLA	Gate	Rollout	Score	Total	Replans/s
$\pi_{0.5}$	3B	–	90.0	–	–	–	90.0	11.1
Cosmos Policy	2B	–	–	–	–	–	1500	0.7
GE-Act	2.1B	–	–	–	–	–	95.0	10.5
LingBot-VA	5.3B	–	–	–	–	–	210.0	4.8
DreamZero	14B	–	–	–	–	–	143.0	7.0
Ours (prior only)	3.3B	44.0	46.0	–	–	–	90.0	11.1
Ours (worst case)	3.3B	43.6	302.2	–	43.9	30.3	420.0	2.4
Ours (with adaptive gate)	3.3B	44.9	196.7	1.5	31.4	14.8	289.3	3.5

D.2 Comprehensive Comparison with Prior World-Action Models

Table 15 compares model sizes and reported results for context. ADEPT performs latent prediction without pixel decoding at deployment. Its standard-LIBERO success is 99.1%; the clean-only and augmented LIBERO-Plus results must be distinguished (88.1% and 90.2% Macro-7). The latter benefits from six seen perturbation families. Cross-paper differences in training, checkpoint, aggregation, and hardware do not isolate the effect of latent rather than pixel-space prediction. In particular, this table cannot establish that predictive structure matters more than model capacity or that every video-trained policy must generate video at deployment.

Table 15: Contextual comparison of world-action models. ADEPT's LIBERO-Plus entry is the augmented setting (Macro-7), and its latency is the adaptive operating point. External entries use their respective training, evaluation, aggregation and profiling protocols. "Video gen." denotes video generation at deployment; "–" denotes unavailable entries.

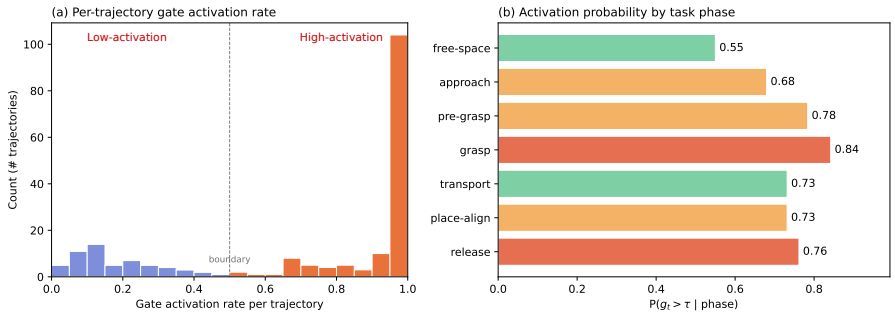
Method	Params	LIBERO	Plus (aug.)	Latency (ms)	Video gen.
$\pi_{0.5}$	3B	96.85	71.5	90	no
Cosmos Policy	2B	98.5	–	1500	yes
GE-Act	2.1B	–	–	95.0	no (GE-Sim: yes)
LingBot-VA	5.3B	98.5	78.5	210.0	yes
DreamZero	14B	–	76.3	143.0	yes
Ours	3.3B	**99.1**	**90.2**	289.3	no

Being-H0.7 (Luo et al., 2026), also shown in Figure 1, reports 82.1% on LIBERO-Plus after standard-LIBERO-only training and 84.8% after LIBERO-Plus fine-tuning, using a 3 B model. These are its published protocol-specific aggregates; they are not recomputed as Macro-7 here. The teaser is contextual and does not establish a common-protocol performance–size frontier.

D.3 Deliberation Gate Activation Statistics

Gate activations are recorded across 200 LIBERO trajectories from the four task suites, using the reported threshold $\tau = 0.55$. Figure 6 shows heterogeneous per-trajectory activation rates, with an approximately 60% overall activation rate. Phase labels are assigned post hoc and are not training supervision. The conditional activation rate is 55% in transit, rises through approach/pre-grasp (68–78%), peaks at grasp (84%), and remains elevated through transport, placement alignment and release (73–76%).

These are conditional frequencies, not shares of all activated steps. The higher grasp rate supports phase-dependent allocation, but transit activation is substantial and failures can occur outside grasping. The results do not support "transit never benefits" or "only critical phases trigger," nor do they establish the causal benefit of individual triggers. The histogram alone also cannot assign each trajectory to a physical routine/contact-rich category without corresponding trajectory annotations.



Gate decisions at $\tau = 0.55$

Time (s)	Phase	Gate score g_t	On / off
0.8	Transit	0.31	Off
2.4	Transit	0.62	On
4.2	Pre-grasp	0.73	On
5.0	Grasp	0.87	On
7.6	Transport	0.48	Off
9.4	Place-align	0.69	On

Figure 6: Gate activation at $\tau = 0.55$. Top panels: observed statistics over 200 LIBERO trajectories, showing per-trajectory activation rates (left) and phase-conditional firing probabilities (right). Grasp exceeds transit by 29 percentage points (84% vs. 55%); the reported overall rate is approximately 60%. Bottom: gate decisions for six time, phase, and score inputs. Phase labels are post hoc. Activation frequency does not establish phase-specific benefit.

E Robustness and Generalization

E.1 Failure Mode Analysis

Table 16 reports the failure breakdown for the augmented ADEPT setting on the Camera and Robot conditions. Each entry reports a share of failures and its implied absolute failure rate per 100 episodes. Because overall failures differ, a larger share need not mean more absolute failures. Camera failures decrease from 20.5 to 1.2 per 100 episodes and Robot failures from 73.5 to 32.0. Reported planning-failure rates decline from 6.4 to 0.1 on Camera and 27.9 to 2.9 on Robot. Perception-failure rates also decline; on Robot, the rounded execution-failure rate is similar (20.6 versus 20.5).

These descriptive categories are consistent with improved candidate selection, but do not independently identify which module caused the gain. The mask does not guarantee nuisance invariance, and a frozen action decoder can still execute different trajectories after reranking. These are augmented results, not a clean-only failure study. Absolute entries are rounded products of total failure rate and category share, so their rounded sums may differ slightly from the total.

Table 16: Failure-mode breakdown for augmented ADEPT on the Camera and Robot LIBERO-Plus conditions. Each cell shows the *share* of a model's failures attributed to that mode (%) $\rightarrow$ the *absolute* rate (failures per 100 episodes). Because ADEPT fails far less often overall (bottom row), a larger share can coexist with far fewer absolute failures.

	Camera split		**Robot split**	
Failure mode	$\pi_{0.5}$	**Ours**	$\pi_{0.5}$	**Ours**
Perception	42% $\rightarrow$ 8.6	30% $\rightarrow$ 0.4	34% $\rightarrow$ 25.0	27% $\rightarrow$ 8.6
Planning	31% $\rightarrow$ 6.4	6% $\rightarrow$ 0.1	38% $\rightarrow$ 27.9	9% $\rightarrow$ 2.9
Execution	27% $\rightarrow$ 5.5	64% $\rightarrow$ 0.8	28% $\rightarrow$ 20.6	64% $\rightarrow$ 20.5
Total (fail./100)	20.5	1.2	73.5	32.0

Table 17: Full LIBERO comparison (%). Compiled from public reports; protocols differ (pretrain / wrist camera / proprio / fine-tune), so the numbers are *not* a strict single-leaderboard comparison.

Method	Params	Spatial	Object	Goal	Long	Avg.
Diffusion Policy	0.15B	78.3	92.5	68.3	50.5	72.4
Octo (ft)	–	78.9	85.7	84.6	51.1	75.1
OpenVLA (ft)	7B	84.7	88.4	79.2	53.7	76.5
SmolVLA-0.25B	0.25B	87.0	93.0	88.0	63.0	82.8
SmolVLA-2.25B	2.25B	93.0	94.0	91.0	77.0	88.8
π_0-FAST	3B	90.0	86.0	95.0	73.0	86.0
MolmoAct-7B-D	7B	87.0	95.4	87.6	77.2	86.8
GR00T-N1	2.2B	94.4	97.6	93.0	90.6	93.9
π_0	3.3B	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$-KI	3.3B	98.0	97.8	95.6	85.8	94.3
VLA-0	3B	97.0	97.8	96.2	87.6	94.7
GO-1 Air	–	94.0	96.8	96.2	91.2	94.6
GO-1	–	96.2	97.8	96.0	89.2	94.8
UniVLA	–	95.4	98.8	93.6	94.0	95.4
$\pi_{0.5}$ (2025)	3B	97.4	98.4	96.2	95.4	96.85
OpenVLA-OFT	7B	97.6	98.4	97.9	94.5	97.1
DeepThinkVLA	2.9B	96.6	99.0	96.4	96.2	97.0
LeRobot $\pi_{0.5}$ (ft)	3.3B	97.0	99.0	98.0	96.0	97.5
X-VLA	–	98.2	98.6	97.8	97.6	98.1
LingBot-VA	5.3B	98.5	99.6	97.2	98.5	98.5
ADEPT (Ours)	3.3B	**100.0**	**100.0**	**98.0**	**98.5**	**99.1**

F Additional Results

F.1 Full Benchmark Comparisons

For completeness, we report the full versions of the three benchmark comparisons abridged in the main text: standard LIBERO (Table 17), the LIBERO-Plus robustness suite across all seven distribution-shift conditions (Table 18), and the per-split RoboCasa365 results (Table 19).

Table 17 reports the full standard-LIBERO comparison. Because the entries are compiled from public reports under differing protocols (pretraining, wrist camera, proprioception, fine-tuning), the table is not a strict single-leaderboard ranking and we read only coarse differences. ADEPT attains the highest average (99.1%), with perfect Spatial and Object suites (100.0%), edging the strongest prior results (LingBot-VA 98.5%, X-VLA 98.1%). The suites are near-saturated, so the field is tightly bunched; ADEPT's average edge over the strongest prior (LingBot-VA) comes from the Spatial and Object suites, where it is perfect, while the two are level on the Long-horizon split (both 98.5%).

Table 18 reports seven family-level rates and aggregates for clean and augmented LIBERO-Plus. Augmented ADEPT obtains 90.2% Macro-7, versus the frozen backbone's 71.5%, ACoT-VLA's 88.5% after training on the generalized split ($96.6/70.4/79.7/95.1/97.1/95.9/85.0$; reported aggregate 88.0), and OpenVLA-OFT+'s recomputed 80.7% Macro-7. Clean ADEPT obtains 88.1% with no LIBERO-Plus perturbation exposure; the listed standard-LIBERO-only comparisons include ACoT-VLA (87.5% Macro-7; reported 86.6) and ABot-M0 (81.6%; reported 80.5). ADEPT-clean is higher than ACoT-VLA on Camera, Background, and Noise and lower on Robot, Language, Light, and Layout. The clean-only family rates are also reported in Table 22. In the augmented setting, the six non-Robot families have training exposure at the family level; the Robot initial-state family remains unseen (68.0% versus 26.5% backbone success). Under augmented training, the Lang. result is 83.1%, below OpenVLA-OFT+'s 85.8% and above ACoT-VLA's 79.7%.

The aggregate column uses the respective reporting conventions. In particular, OpenVLA-OFT+'s repository 79.6% is instance-weighted, whereas ADEPT's 90.2% is Macro-7. We mark that value explicitly and use the matched Macro-7 figures above for comparisons; the $90.2 - 79.6$ subtraction is not a valid same-metric gain.

Table 18: Full LIBERO-Plus family-level results (%). Frozen-backbone and ADEPT results are our evaluations; other baseline values follow their respective published protocols. "Rep. agg." denotes each method's reported aggregate, not a common metric: † is the instance-weighted OFT+ aggregate (79.6); its Macro-7 is 80.7. "Macro-7" recomputes the equal-weight family mean from the listed rates; "aug." marks training on the generalized split. Bold: best per column. ADEPT rows report Macro-7, with clean and augmented training separated.

Method	Camera	Robot	Lang.	Light	Backgr.	Noise	Layout	Rep. agg.	Macro-7
OpenVLA	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6	16.3
NORA	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0	40.5
WorldVLA	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0	25.6
UniVLA	1.8	46.2	69.6	69.0	81.0	21.2	31.9	42.9	45.8
π_0	13.8	6.0	58.8	85.0	81.4	79.0	68.9	53.6	56.1
π_0-FAST	65.1	21.6	61.0	73.2	73.2	74.4	68.8	61.6	62.5
$\pi_{0.5}$ (ours, frozen)	79.5	26.5	74.5	88.3	88.1	71.2	72.2	—	71.5
OpenVLA-OFT_w	10.4	38.7	70.5	76.8	93.6	49.9	69.9	55.8	58.5
OpenVLA-OFT_m	55.6	21.7	81.0	92.7	91.0	78.6	68.7	67.9	69.9
RIPT-VLA	55.2	31.2	77.6	88.4	91.6	73.5	74.2	68.4	70.2
OpenVLA-OFT	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6	71.4
AVA-VLA	55.5	25.9	85.6	95.5	88.9	78.0	74.1	70.1	71.9
ABot-M0 (2026)	60.4	67.9	86.4	96.2	91.6	86.4	82.6	80.5	81.6
ACoT-VLA (2026)	72.6	**82.6**	**87.5**	97.7	96.5	87.8	**88.1**	86.6	87.5
OpenVLA-OFT+ (aug.)	92.8	30.3	85.8	94.9	93.9	89.3	77.6	79.6†	80.7
ACoT-VLA (aug.) (2026)	96.6	70.4	79.7	95.1	97.1	95.9	85.0	88.0	88.5
ADEPT (aug.)	**98.8**	68.0	83.1	**98.2**	**99.0**	**99.1**	85.4	—	**90.2**
ADEPT (clean)	96.0	65.0	81.0	96.0	97.0	97.5	84.2	—	88.1

On RoboCasa365 (Table 19), ADEPT reaches 25.5% split-macro success, versus 16.0% for the listed $\pi_{0.5}$ baseline. The aggregation is recomputed uniformly from each method's displayed split rates. It gives equal weight to Atomic-Seen, Composite-Seen, and Composite-Unseen; the official Overall instead weights these splits by 18, 16, and 16 tasks, respectively (RoboCasa Team, 2026). The two metrics are not interchangeable. On Composite-Unseen, ADEPT reaches 8.0% versus 2.9% for the strongest listed comparison, while absolute success on unseen compositions remains low. These selected published comparisons provide context rather than a claim of leaderboard leadership.

Table 19: RoboCasa365 split success rates (%) and their equal-weight split-macro mean, recomputed from the displayed rates for all methods. This differs from the official 50-task Overall. Atomic-S: Atomic-Seen; Comp.-S/U: Composite-Seen/Unseen. †1.5× horizon, RoboCasa 1.0.1.

Method	Atomic-S	Comp.-S	Comp.-U	Split-macro
GR00T-N1.5†	50.7	14.8	2.7	22.7
GR00T-N1.6	51.1	9.4	1.7	20.7
GigaWorld-Policy 0.1	44.4	11.8	2.9	19.7
$\pi_{0.5}$	39.6	7.1	1.2	16.0
π_0	34.6	6.1	1.1	13.9
Diffusion Policy	15.7	0.2	1.3	5.7
ADEPT (Ours)	**52.1**	**16.3**	**8.0**	**25.5**

F.2 Task-Wise Real-Robot Success Intervals

Table 20 reports Wilson score intervals computed from the success counts in Table 4. ADEPT improves the observed success rate on every task, with gains spanning 24–66 percentage points. Each interval treats the 50 full-reset outcomes for one method and task as Bernoulli trials under that task's evaluation protocol. These are marginal intervals for fixed tasks, without a paired significance test or an interval for generalization across tasks.

Table 20: Task-wise real-robot success rates and 95% Wilson intervals (%). Each method uses 50 trials per task. Gains are differences in observed success rates, in percentage points.

Task	Frozen $\pi_{0.5}$	ADEPT	Gain (pp)
T1	70.0 [56.2, 80.9]	96.0 [86.5, 98.9]	26
T2	66.0 [52.2, 77.6]	94.0 [83.8, 97.9]	28
T3	20.0 [11.2, 33.0]	86.0 [73.8, 93.0]	66
T4	62.0 [48.2, 74.1]	94.0 [83.8, 97.9]	32
T5	64.0 [50.1, 75.9]	88.0 [76.2, 94.4]	24
T6	50.0 [36.6, 63.4]	84.0 [71.5, 91.7]	34

F.3 Additional Robot Platforms and Manipulation Tasks

We further collect real-world demonstrations across three embodiments under four single-/dual-arm data collection configurations (Table 21), varying the teleoperation interface, visual observation layout, and action representation. These comprise a 7-DoF Franka Panda single-arm system, the 6-DoF Custom Arm in both single- and dual-arm configurations, and the ARX R5 dual-arm platform. Together, they cover single-arm manipulation, bimanual coordination, deformable-object handling, and container-based object manipulation.

Table 21: **Robot Setups and Data Collection Configuration.**

Setup	Robot	Interface	Action Space	RGB Observation	Image Size
Franka (single)	7-DoF single arm	FACTR leader arm	Joint positions and gripper state	Wrist view and side view	640×480
Custom Arm (single)	6-DoF single arm	Quest 3 VR	End-effector pose and gripper state	Wrist view and high view	640×480
Custom Arm (dual)	Dual 6-DoF arms	Quest 3 VR	Left/right end-effector poses and gripper states	Left wrist, right wrist, and high view	640×480
ARX R5 (dual)	Dual 6-DoF arms	Quest 3 VR	Left/right end-effector poses and gripper states	Left wrist, right wrist, and high view	640×480

For the Franka Panda single-arm setup, demonstrations are collected through a FACTR-style homomorphic leader arm. The operator directly moves the leader arm, and the corresponding joint commands are recorded for the Franka Panda follower arm. The visual observation contains a wrist RGB view and a side RGB view, both resized to 640×480.

For the single-arm setup, demonstrations are collected with a teleoperation interface that records the end-effector pose and gripper state. The visual observation contains a wrist RGB view and a high-view RGB observation. This setup complements the Franka data by providing single-arm trajectories in Cartesian end-effector space.

For the dual-arm setup, demonstrations are collected on the Custom Arm and ARX R5 platforms using two VR controllers. The operator controls the left and right end effectors independently, and the system records synchronized left-arm and right-arm actions. The visual observation is captured by three D405 RGB cameras, including the left-wrist view, the right-wrist view, and a high-view camera, with each RGB image recorded at 640×480 resolution.

All demonstrations are stored as language-conditioned trajectories. Each trajectory contains the task instruction, multi-view RGB observations, robot states, action commands, and gripper states. Training progress labels are derived from offline-teleoperation timestamps; task-success judgments are used for evaluation and are not benchmark-reward supervision for the value/gate heads. For bimanual demonstrations, the left-arm and right-arm action streams are temporally synchronized with the three RGB camera streams before being used for training and evaluation.

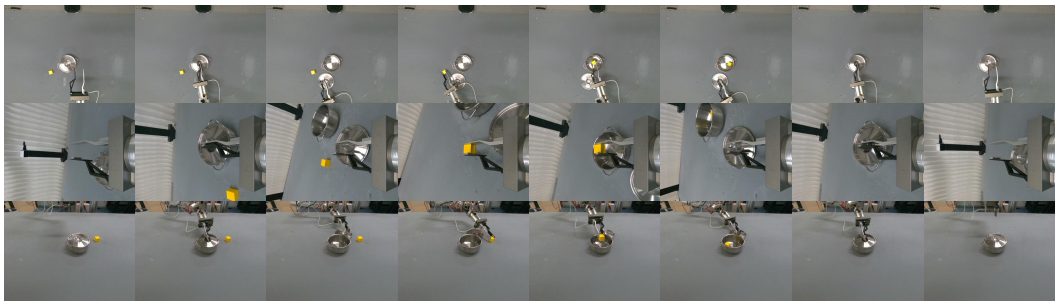


Figure 7: **Single-arm rigid-object pot insertion (Custom Arm).** The robot opens the lid, inserts the rigid block, and closes the lid.

The collected trajectories are organized by robot setup and task type. Single-arm tasks are mainly used to evaluate precise reaching, grasping, placement, and articulated-object interaction, while dual-arm tasks are used to evaluate coordinated manipulation, deformable-object handling, and fine-grained spatial alignment.

F.4 Task Suite

The task suite contains six real-world manipulation tasks. Each task is described by a short imperative instruction, followed by a step-wise procedure and a success criterion. This format matches the language-conditioned setting used by embodied policies and also makes it convenient to align each step with image panels in the supplementary figures.

(1) Single-Arm Rigid-Object Pot Insertion. Instruction: Put the block in the pot.

Procedure:

a. Place the pot with a closed lid and a rigid wooden block on the workspace.
b. Move the single arm to the lid handle.
c. Grasp the lid handle and open the lid.
d. Move the arm to the wooden block and grasp it.
e. Place the wooden block inside the pot.
f. Move back to the lid and close it.

Success Criterion: The rigid block is placed inside the pot, and the lid is closed at the end of the episode.

(2) Single-Arm Heterogeneous-Object Pot Insertion. Instruction: Put the block in the pot.

Procedure:

a. Place the pot with a closed lid and a heterogeneous target object on the workspace.
b. Move the single arm to the lid handle.
c. Grasp the lid handle and open the lid.
d. Move the arm to the target object and grasp it.
e. Place the target object inside the pot.
f. Move back to the lid and close it.

Success Criterion: The target object is placed inside the pot, and the lid is closed at the end of the episode.

(3) Single-Arm Cloth-Strip Removal. Instruction: Set the top strip aside.

Procedure:

a. Initialize the scene by stacking multiple cloth strips on the workspace.

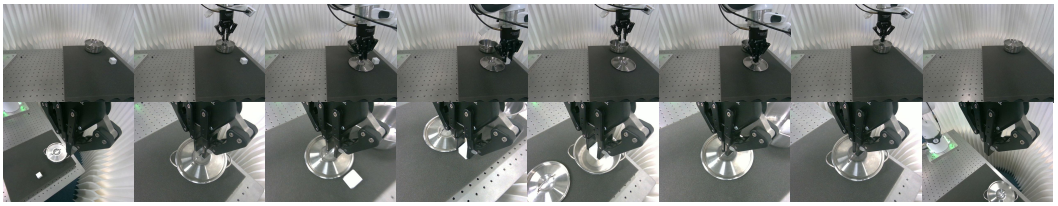


Figure 8: **Single-arm heterogeneous-object pot insertion (Franka).** The robot opens the lid, inserts the target object, and closes the lid.

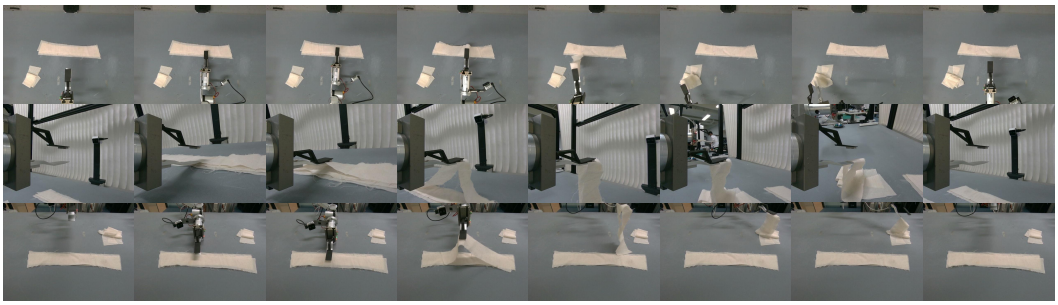


Figure 9: **Single-arm cloth-strip removal.** The robot moves the topmost cloth strip to the side without disturbing the lower strips.

b. Move the single arm from the home pose to the top cloth strip.
c. Align the gripper with a graspable region on the top strip.
d. Close the gripper to grasp only the top strip.
e. Lift the strip and move it to the target side region.
f. Release the strip and return the arm to a safe pose.

Success Criterion: The topmost cloth strip is removed and placed to the side without pulling away the lower cloth strips.

(4) Bimanual Multi-Object Pot Insertion. Instruction: Put the block and ball in the pot.

Procedure:

a. Place the pot, multiple wooden blocks, and balls on the workspace.
b. Use one arm to reach the lid handle and open the pot.
c. Use the other arm to grasp a target block or ball.
d. Place the grasped object inside the pot.
e. Repeat the grasp-and-place step until all target objects are inside.
f. Close the lid after the final object is inserted.

Success Criterion: All wooden blocks and balls are placed inside the pot, and the lid is closed at the end of the episode.

(5) Bimanual Towel Folding. Instruction: Fold the towel to the left.

Procedure:

a. Place the unfolded towel on the workspace.
b. Move both arms to the two grasping regions of the towel.
c. Close both grippers to hold the towel.
d. Lift the towel while maintaining the relative distance between the two arms.
e. Bring the two grasped regions together to fold the towel.

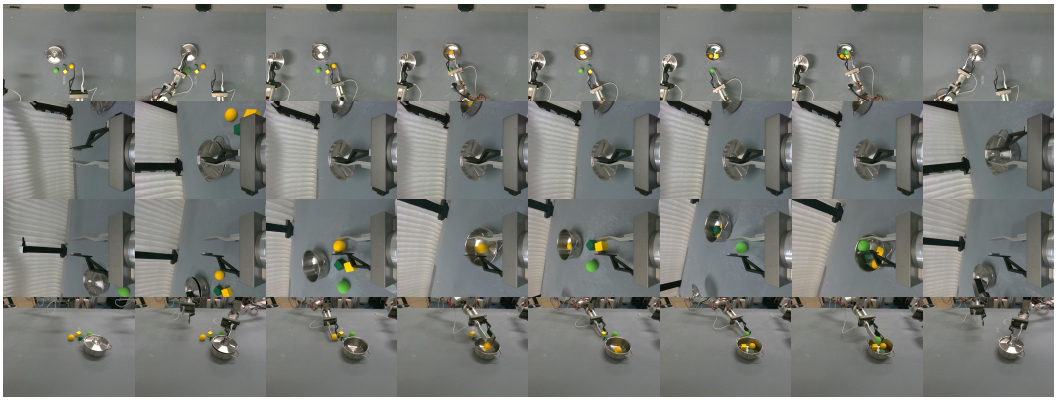


Figure 10: **Bimanual multi-object pot insertion.** The robot opens the lid, inserts multiple blocks and balls, and closes the lid.

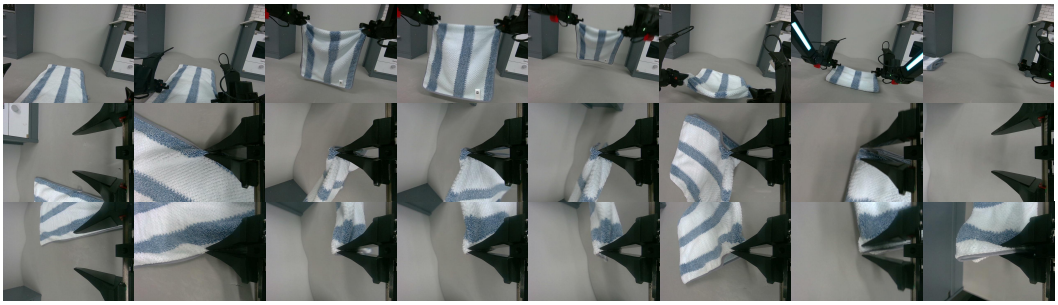


Figure 11: **Bimanual towel folding.** The robot folds the towel with two arms and places the folded towel to the side.

f. Place the folded towel in the target side region and release it.

Success Criterion: The towel is folded and placed to the side without severe twisting, dropping, or collapse.

(6) Bimanual Cloth-Strip Alignment. Instruction: Overlap the strips with no gap.

Procedure:

a. Place the foam board, the grid cloth strip, and the white cloth strip in the workspace.
b. Use both arms to grasp the grid cloth strip.
c. Place the grid cloth strip flat on the foam board.
d. Use both arms to grasp the white cloth strip.
e. Move the white strip above the grid cloth strip.
f. Align the white strip with the grid cloth and release it carefully.

Success Criterion: The grid cloth is placed on the foam board, and the white strip is aligned flush with the grid cloth with minimal offset or rotation.

F.5 Evaluation Protocol

For each task, we evaluate whether the robot completes the language instruction under the given initial scene. We report task success based on the final object state rather than only the executed motion. A rollout is marked successful only if all required objects are placed in the target configuration and no critical object is dropped, misplaced, or left in an incomplete state.

For deformable-object tasks, we additionally inspect whether the cloth or towel is excessively dragged, twisted, wrinkled, or misaligned. For pot-based tasks, we check whether the lid is correctly

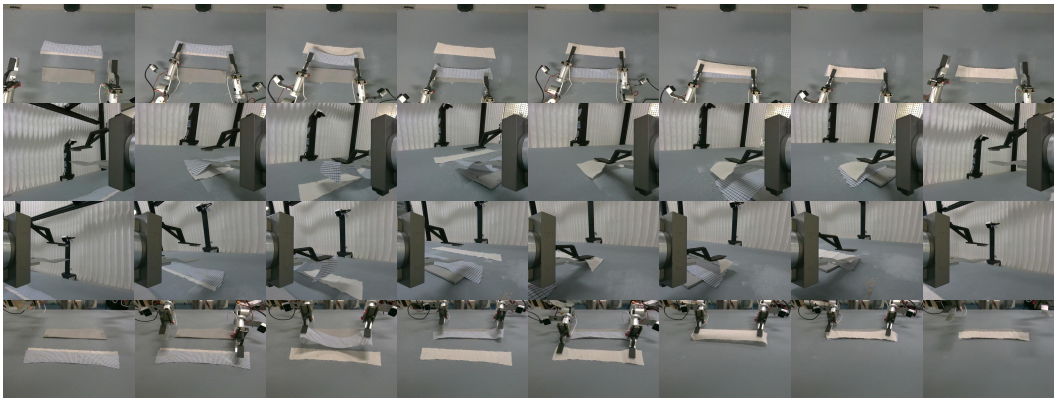


Figure 12: **Bimanual cloth-strip alignment.** The robot places the grid cloth on the foam board and aligns the white strip on top of it.

opened and closed, and whether all target objects are fully inside the pot. For the cloth-alignment task, we further evaluate the relative offset and angular deviation between the white strip and the grid cloth.

F.6 Matched Real-Robot Controls and Recording Evidence

For every task, the reactive control uses the same frozen $\pi_{0.5}$ checkpoint, scene protocol and 50 full-reset trials as its ADEPT counterpart. Tasks 1–6 respectively report frozen/ADEPT successes of $35/48$, $33/47$, $10/43$, $31/47$, $32/44$, and $25/42$, each out of 50 trials per method. These are two success counts, not success-over-trial fractions. The pooled totals are $166/300$ (55.3%) and $271/300$ (90.3%), a 35.0-point improvement. Task 3 is zero-shot with respect to task-specific base-policy adaptation. Task 5 is the ARX R5 towel-folding experiment. The default threshold is shared with the runs in Figure 3.

The supplementary video includes the ARX R5 towel-folding execution.

G Candidate Consequences and Clean-Transfer Diagnostics

Episode success evaluates the complete controller. A complementary same-state diagnostic separates candidate support from the accuracy of candidate ranking. The definitions below specify this diagnostic independently of the episode-level scores reported above.

G.1 Same-State Candidate Evaluation

At each held-out anchor i, save the simulator, controller, observation-history, and random states. Sample $K = 5$ candidates once from the frozen prior and give the identical sequences to every scorer. Scoring precedes observation of the execution outcomes. Restore the same state before executing each candidate for all $H = 4$ chunks, without replanning inside that branch. Anchors are selected independently of ADEPT's score and gate. Repeated stochastic branches use a common replicate schedule across candidates.

Let $Y_{ikr} \in \{0, 1\}$ denote benchmark success for candidate k from anchor i on repeat r. Where the four-chunk window does not determine success, each branch continues with the same fixed policy and remaining action budget. This tests the consequence of that prefix under the specified continuation. Neither the learned progress scorer nor demonstration timestamps define the outcome labels. Write $\hat{q}_{ik} = R^{-1} \sum_{r=1}^{R} Y_{ikr}$ and let $s_{ik}^{(m)}$ be scorer m's prediction.

For $D_i = \{(k, j) : k < j, \hat{q}_{ik} \neq \hat{q}_{ij}\}$, the state-level pairwise accuracy is

$$A_i^{(m)} = \frac{1}{|D_i|} \sum_{(k,j) \in D_i} \left[\mathbf{1}\{(s_{ik}^{(m)} - s_{ij}^{(m)})(\hat{q}_{ik} - \hat{q}_{ij}) > 0\} + \frac{1}{2} \mathbf{1}\{s_{ik}^{(m)} = s_{ij}^{(m)}\} \right]. \tag{21}$$

States with $D_i = \varnothing$ contribute to coverage and selection metrics but not ranking accuracy. Means are computed over eligible states within each task and then across tasks. Table 6 reports point estimates of pairwise accuracy, selected-candidate outcome, and regret. Candidate pairs from the same state share execution outcomes and are not independent observations; no significance test is inferred from the number of pairs. App. H specifies the population and the paired full-minus-control differences.

For maximal-score ties, use uniform selection over the tied candidates, with expected outcome $\hat{q}^{(m)}_{i,sel}$. The empirical best-candidate reference and regret are

$$Q_{oracle} = \frac{1}{N}\sum_i \max_k \hat{q}_{ik}, \qquad Q^{(m)}_{sel} = \frac{1}{N}\sum_i \hat{q}^{(m)}_{i,sel}, \qquad G^{(m)} = Q_{oracle} - Q^{(m)}_{sel}. \tag{22}$$

These expressions apply within a task before task-macro averaging. With $R = 1$, Q_{oracle} equals the fraction of anchors with at least one successful candidate. With multiple repeats it is an empirical ceiling subject to finite-sample maximization bias, not a known optimal success probability. Coverage is shared by all scorers using the same candidates. This common support permits comparisons among the direct critic, progress-only rollout, full scoring, and objective ablations; evaluation outcomes must remain excluded from training and tuning. Relative-critic pairwise accuracy uses its predicted preferences rather than differences between absolute progress values; selection uses its tournament over the saved candidates. Same-state evaluation controls candidate support and physical outcome measurement. Complete-controller experiments additionally measure execution and gate costs, but their visited states can diverge after different actions are selected. The two evaluations answer different questions.

G.2 Clean-Only Attribution and Predictive Error

A clean-transfer component comparison fixes standard-LIBERO demonstrations, train/validation partition, action prior, candidate settings, and final evaluation variants. Each variant's mask, dynamics, and heads are trained using only this clean partition. Full CLF/MRT/ADV, full-latent prediction, and fixed-gate or fixed-scoring variants are compared on the same seven held-out factor families. Family rates retain their own success counts and denominators; Macro-7 averages the seven rates equally. Augmented-training ablations answer a separate question and cannot be substituted for these clean-transfer contrasts.

To isolate recursive training, the single-step and teacher-forcing controls use the same $H = 4$ deployment window and fixed encoder/mask coordinates. A normalized rollout error can then divide prediction MSE by one common latent scale computed on training data. For CLF versus full-latent variants, outcome-ranking metrics provide a common comparison; dividing each model's error by its own variance does not align different latent spaces.

G.3 Diagnostic Comparisons

Table 22 gives clean-transfer family rates; the component and candidate-ranking results using the protocols above appear in main-text Tables 5 and 6.

Table 22: Clean-transfer family results (%). ADEPT-clean is trained only on standard LIBERO. Macro-7 equally averages the seven family-level success rates.

	Camera	Robot	Lang.	Light	Bg.	Noise	Layout	Macro-7
ADEPT-clean	96.0	65.0	81.0	96.0	97.0	97.5	84.2	88.1

Robot-initial-state shifts remain the main weakness, while appearance-related families have higher success rates. These results use the clean-trained checkpoint. The component comparisons in Table 5 evaluate representation, recursive training, and candidate selection under the same clean-only protocol.

The clean-only component results and same-state candidate-ranking results are reported in the main text (Tables 5 and 6). The single-step $1 \to 4$ and teacher-forcing $4 \to 4$ controls retain Full's deployment window. All scorers in the ranking diagnostic share anchor states and candidate arrays;

their physical outcomes come from restored-state execution with a fixed continuation policy. Pairwise ordering and selected-candidate outcome measure complementary aspects of action selection under common candidate support.

H Ranking Reliability, Gate Controls, and Prediction Scale

H.1 Ranking Population and Paired Differences

The same-state diagnostic in App. G separates proposal quality from the choice among proposals. Table 23 defines the population and execution protocol for Table 6. Each anchor is indexed by task, evaluation seed, episode, and anchor position; its state includes the simulator, controller, observation history, and random state. Candidates and execution repeats add two further indices. These indices define the record schema, while checkpoint steps identify the scoring configuration. The five evaluation seeds vary environments and action sampling at fixed trained checkpoints; they are not five independent training runs.

Table 23: Same-state ranking population and execution protocol. Task IDs use the canonical zero-based LIBERO-Spatial order. Candidate support and branch outcomes are shared by every scorer.

Field	Specification
Tasks and source episodes	Task IDs 0–9; evaluation seeds $11, 23, 37, 53, 71$; 20 episodes per task and seed, indexed 0–19: 1000 episodes total.
Anchors and source policy	Frozen $\pi_{0.5}$; two anchors per episode at one-third and two-thirds of the usable preterminal decision interval: $N_j = 200$, $N = 2000$.
Candidates and branches	$K = 5$, $H = 4$, $R = 5$; repeat seeds $101, 103, 107, 109, 113$ shared across candidates; $NKR = 50{,}000$ branch continuations.
Continuation and checkpoints	Same frozen $\pi_{0.5}$ checkpoint as the source; remaining original task budget after the prefix. Scoring heads use update 30,000; the no-$\mathcal{L}_{ai}$ control uses its own budget-matched checkpoint.
Eligible ranking population	140 anchors from 70 episodes per task: $1400/2000 = 70.0\%$ coverage, 700 eligible episodes, 11,200 discordant pairs.
Index and failure rules	Anchor positions $0, 1$, candidates 0–4, repeats 0–4. Restore failures invalidate both anchors of a source episode for all scorers; replace the whole episode before scoring. Execution failures count as zero success.
Aggregation and empty tasks	All anchors enter selection metrics; only outcome-discordant anchors enter pairwise accuracy. A task with zero eligible anchors has undefined accuracy and is excluded from the accuracy macro with its count reported; all 10 tasks here are eligible.

For an episode with T recorded decision states, usable anchor indices run from 0 through $U = T - H - 1$; choose $\lfloor U/3 \rfloor$ and $\lfloor 2U/3 \rfloor$. Episodes with fewer than two distinct usable anchors are replaced before scoring. The branch prefix and continuation together cannot exceed the task's original command budget. Source episodes, branch outcomes, and timing records are excluded from critic training, validation, and threshold calibration.

Direct-scoring controls. Both critics consume the same pooled control latent, pooled language embedding, and four proposed action chunks. Let their widths be d_c, d_l, and d_a (the flattened scalar width of one chunk). A shared linear action embedding maps d_a to 128; four learned positional embeddings of width 128 are added before concatenation. Thus the head input width is $p = d_c + d_l + 512$. The head is an MLP $p \to 512 \to 256 \to 1$ with GELU and 0.1 dropout after each hidden layer. Including the action and positional embeddings, each critic has exactly $512p + 128d_a + 132{,}737$ trainable parameters. This expression specifies the count without assuming a different frozen-backbone tensor width. The two critics match one another in input access, architecture, and training-window budget; this does not make their capacity equal to the full dynamics model.

At evaluation, relative preference compares candidates from the same anchor; training preferences express demonstration progress, not empirical success differences between restored-state candidates. This is a specified relative-progress control, not a reproduction of another method's full observation

Table 24: Direct critic supervision and optimization. Both heads use frozen ADEPT features and receive no evaluation-branch success labels. A window contains the current observation and $H = 4$ demonstration action chunks.

Configuration	Absolute-progress critic	Relative-progress critic
Target	Sigmoid output regresses endpoint demonstration progress $y = (t + H + 1)/T$.	Bradley–Terry probability $\sigma(s(w_1) - s(w_2))$ predicts which endpoint has greater normalized demonstration progress.
Loss and sampling	Mean squared error; uniform task, then episode, then valid window.	Binary cross-entropy; two independent windows from the same task, discard equal progress, balance pair orientation $1:1$. Training pairs may have different current states.
Data partition	Standard-LIBERO demonstrations; per-task episode-disjoint $90/10$ train/validation split using seed 0. No evaluation episodes or branch outcomes enter either partition.	
Effective batch	256 windows per microbatch, two accumulation steps: 512 windows/update.	128 pairs per microbatch, two accumulation steps: 256 pairs (512 windows)/update.
Training	AdamW, peak learning rate 5×10^{-5}, linear 10,000-step warm-up then cosine decay to zero; 30,000 updates; $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, weight decay 0.01, gradient-norm clip 1.0, initialization seed 0.	
Checkpoint rule	Monitor held-out loss every 1000 updates; use the fixed 30,000-update checkpoint for both heads, with dropout disabled.	
Candidate selection	Cache five scalar outputs; take the maximum.	Cache five utilities; compute ten pair probabilities and select the maximum mean win probability (soft Borda).

critic. Maximal-score ties use the uniform expected outcome as in App. G. Sharing proposals alone does not establish that every possible direct critic has been matched.

For fixed $J = 10$ tasks, define $\mathcal{E}_j = \{i \text{ in task } j : |D_i| > 0\}$ and

$$A^{(m)} = \frac{1}{J}\sum_{j=1}^{J}\frac{1}{|\mathcal{E}_j|}\sum_{i\in\mathcal{E}_j} A_i^{(m)}, \qquad Q^{(m)} = \frac{1}{J}\sum_{j=1}^{J}\frac{1}{N_j}\sum_{i \text{ in } j}\hat{q}_{i,sel}^{(m)}. \tag{23}$$

All anchors enter $Q^{(m)}$, including those without a discordant pair. For each control m, Table 25 reports the full-minus-control differences $\Delta A = A^{(\text{full})} - A^{(m)}$ and $\Delta Q_{sel} = Q^{(\text{full})} - Q^{(m)}$ in percentage points, formed from the anchor-level differences $A_i^{(\text{full})} - A_i^{(m)}$ and $\hat{q}_{i,sel}^{(\text{full})} - \hat{q}_{i,sel}^{(m)}$ under the task-macro weighting of Eq. (23). Every scorer ranks the same candidates against the same executed branches, so each difference is paired at the anchor level; the two anchors of an episode share its restored source state, so the episode is the unit for any dispersion estimate. For accuracy, both anchors in each of the 70 eligible episodes per task contribute (700 episodes); selection uses all 100 episodes per task (1000 episodes). Candidate pairs never serve as independent trials. The differences condition on fixed tasks, trained checkpoints, and saved branch outcomes; they do not include training-run variability or finite-R outcome uncertainty, and Table 25 reports them as point estimates without interval or significance statements.

Table 25: Paired ranking differences for fixed tasks and checkpoints. Entries give the full scorer minus the control in percentage points on the shared anchors, candidates, and branch outcomes of Table 6; accuracy uses the 700 eligible episodes and selected outcome all 1000 episodes.

Control	ΔA (pp)	ΔQ_{sel} (pp)
Direct absolute critic	10.0	7.0
Relative critic	5.0	3.0
Progress-only rollout	14.0	11.0
Without $\mathcal{L}_{ai}$	7.0	5.0

H.2 A Direct Heuristic Gate at Matched Budgets

The direct gate uses the four terms and coefficients in Eq. (19). For each raw feature f_v, compute μ_v^{tr} and s_v^{tr} as the population mean and standard deviation over a fixed training-statistics partition of 50,000 decision states (5000 per Spatial task). Freeze these values and use $z_v^{tr} = (f_v - \mu_v^{tr})/\max(s_v^{tr}, 10^{-6})$. The direct decision is

$$g_t^{\text{heur}} = \text{sigmoid}(z_{inv}^{tr} + 0.5z_{ref}^{tr} + 0.5z_{bg}^{tr} - z_{val}^{tr}), \qquad \text{invoke iff } g_t^{\text{heur}} > \tau_{\text{heur}}. \tag{24}$$

The learned gate in Eq. (10) additionally receives the pooled control latent and control displacement, and is fitted to the per-batch standardized targets in Eq. (19). The direct rule instead uses the frozen training-set statistics above. This comparison therefore evaluates two scheduling controllers at matched calibration budgets; it does not isolate learning from input access or normalization.

Both gates use the tentative-action roll-in and historical-reference update in Algorithm 1. A separate calibration partition has 10,000 states (1000 per task) from episode-disjoint training-side trajectories. Within each partition, sample tasks uniformly, then episodes, then decision indices with seed 20270923. Test episodes enter neither partition. For each gate separately, sort calibration scores in ascending order; the threshold for budget b is the score at one-based index $\lceil (1-b)10{,}000 \rceil$, namely 8500, 7000, or 4000 for $b = 0.15, 0.30, 0.60$. Exact score ties use an independent Bernoulli draw with probability $(10{,}000b - n_>)/n_=$, clipped to $[0, 1]$, where $n_>$ and $n_=$ count scores above and at the threshold. Thus raw feature units and gate score distributions do not require a universal numerical threshold. Statistics and thresholds stay fixed at test time.

The paired evaluation uses the same ten Spatial tasks, five evaluation seeds, and 20 initial-condition blocks per task and seed: 1000 paired episodes per budget, 6000 controller episodes across both gates and three budgets. Initial states and seed schedules are shared; candidate arrays may diverge when controllers visit different states. The prior retains its frozen source checkpoint; mask, dynamics, progress and learned-gate checkpoints are fixed at update 30,000, with $K = 5$, $H = 4$, and the full scorer. An initial-condition block contains one episode from each controller. Success uses equal task weights; invocation is the ratio of invoked to valid decisions, pooled across test episodes.

For paired success, write $D \in \{-1, 0, 1\}$ for learned minus heuristic success within an initial-condition block; Δ in Table 26 is the mean of D over the 1000 blocks of a budget, with equal task weights, in percentage points. Because the blocks share initial states and seed schedules, Δ is a paired difference rather than a difference of two independent success rates. It is reported as a point estimate without an interval or significance statement, and it conditions on the fixed checkpoints, calibration partition, and thresholds.

Table 26: Learned and direct heuristic scheduling at calibration-matched budgets. Invocation is decision-weighted; success is task-macro. Δ denotes paired learned-minus-heuristic success over the 1000 initial-condition blocks per budget.

Target	Gate	Invoke (%)	Success (%)	Δ (pp)
15%	Learned	15.2	91.0	0.0
	Heuristic	14.8	91.0	—
30%	Learned	30.1	94.0	1.0
	Heuristic	30.3	93.0	—
60%	Learned	60.0	96.0	1.0
	Heuristic	59.8	95.0	—

This paired comparison has its own population. The existing three-seed controller ablation and fixed-checkpoint budget sweep remain separate summaries with their own evaluation seeds and episode counts; their aggregate success rates cannot supply paired observations for this table, and the success rates of the separate studies need not coincide.

H.3 Scale Diagnostics for the Implemented Objective

The benchmark objective uses the unnormalized $\mathcal{L}_{ai}$ in Eq. (6); the bound in App. A.3 concerns a different, normalized objective. The diagnostic evaluates updates 5000, 15,000, and 30,000 on 4000 held-out transitions (400 per Spatial task), with fixed observation/action indices and candidate

arrays. Draw transitions uniformly by task, episode, and valid decision index with seed 20270923, without overlap with the 2000 calibration transitions (200 per task) used below. The training variance reference uses $M = 8$ draws, $\sigma = 1.0$, a squared-distance sum over latent scalars, and a mean over draws and batch elements. Rollout loss sums squared error over latent scalars and averages over batch elements and the $H = 4$ steps. The variance mean and deviations share the same M draws, giving the finite-sample factor $(M - 1)/M$ described in App. A.3.

Fix $s_0^2 = \mathbb{E}_{\mathcal{C}}\|z_{ref}^{ctrl}\|_2^2/d_z$ on the calibration partition at update 5000, where d_z is the number of latent scalar entries. Report target and prediction RMS relative to s_0, and divide both native losses by $d_z s_0^2$ solely for reporting. This rescaling changes neither the optimized objective nor its coefficients. On the shared $K = 5$ ranking candidates, report $K^{-1}\sum_{k=1}^{K}\|u_k - \bar{u}\|_2^2$, where $u_k = \hat{z}_k/\max(10^{-6}, \|\hat{z}_k\|_2)$ and $\bar{u} = K^{-1}\sum_k u_k$. Here K indexes deployment candidates, whereas M indexes the variance-reference draws. Each checkpoint retains its own mask and scoring path; ranking and selected outcome use the same physical branch outcomes as Table 6.

Table 27: Checkpoint scale and action-selection diagnostics. RMS uses the frozen amplitude reference s_0; losses retain their native definitions and are displayed in units of $d_z s_0^2$. Separation uses unit-normalized predictions over $K = 5$ candidates.

Diagnostic	Step 5000	Step 15,000	Step 30,000
Target RMS/s_0	1.000	1.000	1.050
Prediction RMS/s_0	0.950	1.050	1.050
Rollout loss $\mathcal{L}_{roll}/(d_z s_0^2)$	0.185	0.140	0.120
Variance loss $\mathcal{L}_{ai}/(d_z s_0^2)$	-0.045	-0.060	-0.080
Unit-normalized candidate separation	0.080	0.100	0.120
Pairwise accuracy (%)	70.5	75.5	80.0
Selected-candidate outcome Q_{sel} (%)	77.5	81.5	85.0

The frozen scale tracks amplitude drift; it does not align changing latent spaces. Rollout error is therefore a within-checkpoint diagnostic, while action ranking on shared physical branches supplies a common utility comparison. Interpretation requires agreement among amplitude, predictive error, and selection quality. A more negative variance loss caused only by growing norms does not establish better consequence prediction. Unit separation tests directional variation; ranking tests whether those differences aid selection. Neither establishes task sufficiency or calibrated uncertainty. An aligned-error control needs a fixed encoder and mask, and a normalized-loss variant needs a separate budget-matched training run; neither can certify the original objective retrospectively.