ACTING AT THE MOMENT OF CONSEQUENCE: PHYSICAL FORESIGHT FOR VLA ACTION REFINEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models can produce coherent manipulation behavior yet fail during physical interactions that require precise control. Refining their actions requires identifying when local adjustments can still affect an operation's outcome. We introduce PREEMPT, a framework for selectively refining the actions of a fixed VLA policy. A V-JEPA-based anticipator foresees physical decision windows and initiates local correction. A latent world model predicts short candidate trajectories, and a completion-time evaluator relates these forecasts to the likelihood and timing of operation completion. These estimates guide smooth corrections constrained by the policy's intended motion, while predicted trajectory diversity guides further search allocation. We evaluate PREEMPT on LIBERO and RoboCasa with $\pi_{0.5}$, on LIBERO with OpenVLA-OFT, and on four real-robot tasks with an Agilex dual-arm robot. With $\pi_{0.5}$, PREEMPT increases success from 96.7% to 97.1% on LIBERO and from 64.9% to 68.6% on RoboCasa. Real-robot success rises from 72.5% to 82.5%, a gain of 10 percentage points. In a LIBERO-10 scheduling comparison, window-based search uses 58.4% fewer candidate evaluations than searching at every replan.

1 INTRODUCTION

A robot can select the correct object and follow a suitable approach path, yet fail to secure its grasp. The error may become apparent during lifting, after the opportunity for a small adjustment has passed. Vision-language-action (VLA) models trained on large, diverse datasets generalize across manipulation tasks (Kim et al., 2024; Physical Intelligence et al., 2025), but reliable execution also requires precise control as contact forms, grasps stabilize, and articulated objects move. We seek to improve execution at these physical transitions (Figure 1) while keeping the VLA's parameters and architecture fixed.

Inference-time refinement uses additional computation to improve the policy's actions. Value-guided steering and test-time verification select among policy proposals (Nakamoto et al., 2025; Kwok et al., 2025), while predictive methods evaluate their consequences before execution (Wu et al., 2025; Qi et al., 2026). DreamAvoid triggers short-horizon video prediction at critical phases and ranks proposals using learned progress values; CycleVLA combines progress-aware fine-tuning with VLM-based failure prediction and subtask backtracking (Fan et al., 2026; Ma et al., 2026). We study how short-horizon prediction can guide structured corrections to a VLA's intended motion, and when the additional computation is useful.

The opportunity for useful correction changes over time. During contact formation, small motion differences can alter the outcome of an attempt. Before or after this interval, candidate actions may lead to similar outcomes, leaving little basis for choosing among them. Search should therefore depend on both the stage of the interaction and the diversity of consequences predicted for the available actions.

Evaluating a correction requires reasoning beyond the prediction horizon. A five-step forecast may capture gripper alignment or initial contact, while stable control or a completed door motion emerges tens of steps later. Immediate completion and endpoint distance provide incomplete measures of this progress. An evaluator must relate the observed history and predicted trajectory to the same operation's eventual completion.

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

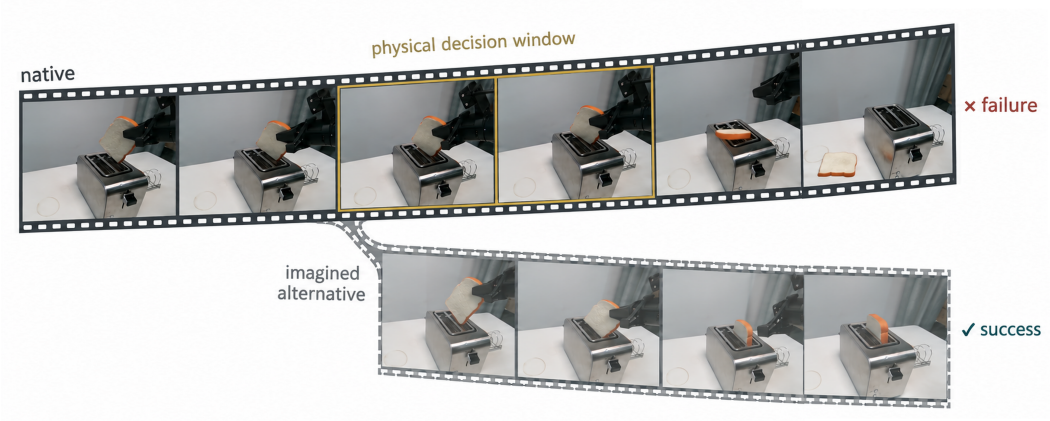

Figure 1: **A physical decision window for toast insertion.** The nominal motion approaches the toaster but releases the bread outside its slot. The highlighted interval marks an opportunity to refine the motion before the outcome is determined. The imagined alternative illustrates how a small adjustment can lead to successful insertion.

Corrections must also respect the policy's motion structure. An action sequence encodes direction changes, interaction phases, and orientations beyond its executed prefix. These later actions provide a reference for modifying the prefix coherently. Across replans, the search must account for previously executed corrections to limit accumulated motion drift.

We introduce PREEMPT, a framework for refining a fixed VLA's actions around critical physical interactions (Figure 2). A V-JEPA-based anticipator forecasts upcoming *physical decision windows* (PDWs), intervals in which local action changes can affect the current operation's outcome. A dual-view latent world model predicts candidate futures, whose diversity guides subsequent search. A *completion-time evaluator* combines observed history, object queries, candidate actions, and five-step predictions to estimate whether and when the operation will finish. Completed events, failed attempts, and censored observations provide a common temporal learning objective.

Using these predictions, PREEMPT searches over smooth spline corrections to the execution prefix, with the full intent segment defining the motion reference. Path and cumulative-offset costs preserve motion structure across replans. The selected correction is executed if it improves the regularized objective over the original prefix. New observations then update the history for the next decision.

Our contributions are:

1. We organize action refinement around physical decision windows, using predicted consequence diversity to guide search and observed interaction state to connect successive corrections.
2. We learn object-conditioned completion-time distributions from completed, failed, and censored attempts, relating short candidate futures to longer operation outcomes.
3. We develop twelve-dimensional spline search in the policy's intent coordinates, with structural costs that account for both local deformation and accumulated corrections.
4. We evaluate on LIBERO, RoboCasa, and an Agilex robot. With 0.335B auxiliary parameters, 9.3% of $\pi_{0.5}$'s size, PREEMPT improves real-robot success by 10 percentage points. Window-guided search uses 58.4% fewer candidate evaluations than searching at every replan in a LIBERO-10 comparison.

2 RELATED WORK

VLA adaptation and inference-time steering. OpenVLA, $\pi_{0.5}$, and OpenVLA-OFT provide transferable manipulation policies with different action-generation and adaptation strategies (Kim et al., 2024; Physical Intelligence et al., 2025; Kim et al., 2025). V-GPS steers a fixed policy using an offline-learned value function, while RoboMonkey combines sampling, proposal augmentation,

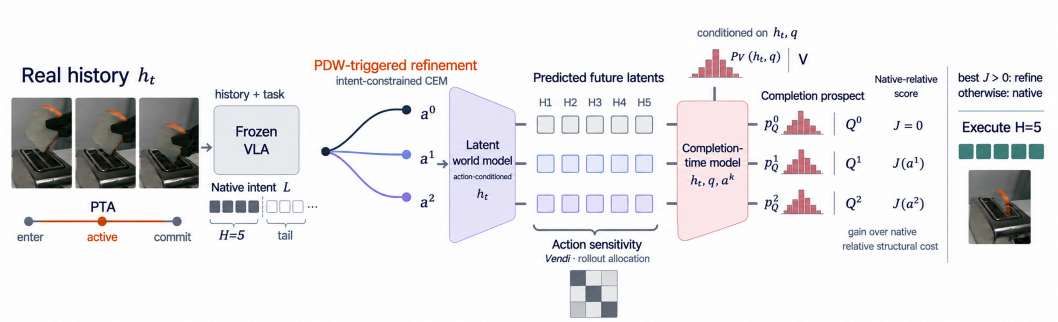

Figure 2: **Overview of PREEMPT.** PTA anticipates an upcoming physical decision window, initiating intent-constrained refinement of the frozen VLA's action prefix. Five-step latent predictions support Vendi-based search allocation and completion-time evaluation. CEM balances completion gain with structural cost relative to the nominal prefix; the selected five actions are executed before replanning.

and verification (Nakamoto et al., 2025; Kwok et al., 2025). PREEMPT refines a policy-generated sequence through smooth motion corrections. We train auxiliary prediction and evaluation models while keeping the VLA weights fixed.

Physical foresight and proactive correction. FOREWARN combines latent prediction with VLM-based assessment, and GPC uses predictive world models to rank and refine generative-policy proposals (Wu et al., 2025; Qi et al., 2026). DreamAvoid triggers short-horizon video prediction at critical phases and ranks sampled actions using learned progress values, supplemented by learning from boundary data (Fan et al., 2026). CycleVLA fine-tunes a progress-aware VLA and combines VLM-based failure prediction, subtask backtracking, and minimum Bayes risk decoding (Ma et al., 2026). PREEMPT brings together physical-window anticipation, completion-time evaluation of latent trajectories, and continuous search constrained by the policy's intended motion.

Latent dynamics and consequence diversity. DINO-WM and V-JEPA 2 use learned visual representations for predictive control (Zhou et al., 2024; Assran et al., 2025). Our predictor combines spatial feature aggregation with action-conditioned latent dynamics and SIGReg regularization (Balestriero & LeCun, 2025). Vendi Score measures the effective diversity of predicted trajectories and guides search allocation (Friedman & Dieng, 2023). The completion-time evaluator scores candidate outcomes, and the cross-entropy method (CEM) optimizes the corrections (de Boer et al., 2005).

3 PROBLEM FORMULATION

Let h_t contain visual observations, robot states, and executed actions up to control step t, and let ℓ denote the task instruction. A fixed VLA π generates an action sequence $A_t^0 = \pi(h_t, \ell)$. We take an *intent segment* I_t of length L and use its first H actions as the nominal prefix a_t^0. PREEMPT searches over locally modified prefixes a_t^k from the same observed state. The full intent segment provides motion geometry for the correction, while prediction and execution cover its first H steps.

A local operation is specified by $q = (\ell, \rho, o, g)$, comprising the instruction, operation type, object or articulated part, and target relation. An *attempt* is a continuous effort to complete that operation. For attempt j, the physical decision window is the interval in which local actions can affect its outcome:

$$D_j = [t_j^{\text{enter}}, t_j^{\text{commit}}] \subseteq \text{Attempt}_j. \tag{1}$$

Entry and commit mark the beginning and end of this interval. Operation completion is determined by physical evidence such as stable object control or the required door joint state.

PTA initiates correction for the relevant operation, and its observed interaction state connects subsequent replans. We use $H = 5$, with $L = 10$ for $\pi_{0.5}$ on LIBERO and RoboCasa and $L = 8$ for OpenVLA-OFT on LIBERO. On RoboCasa, the intent segment is the first ten actions of a fifty-step

policy output. Candidates share the same operation queries and observed history; execution updates the history and accumulated command offsets.

4 PREEMPT: PHYSICAL FORESIGHT AND INTENT-CONSTRAINED REFINEMENT

Figure 2 shows how PREEMPT combines physical decision windows, latent prediction, and completion-time evaluation to refine the actions of a fixed VLA.

4.1 ANTICIPATING PHYSICAL DECISION WINDOWS

The physical transition anticipator (PTA) predicts upcoming PDWs and tracks their progression from observed interaction history. It combines frozen robot-domain V-JEPA features, language, proprioception, gripper state, and executed actions. Projected features enter a unidirectional GRU with three scalar heads for window entry, active occupancy, and commit. We train on continuous episodes containing multiple annotated windows, using masked, class-weighted binary cross-entropy.

During execution, sustained high or low occupancy probability establishes or closes a window; the entry and commit heads refine its boundaries. This temporal decoding establishes a correction context for the relevant operation and object, which observed interaction state maintains across replans. Grasp refinement uses observed object control and gripper state; articulated-object refinement follows the opening or closing target.

4.2 PREDICTING CANDIDATE CONSEQUENCES

Our latent world model predicts the consequences of each candidate action prefix. A frozen ViT-B/14 extracts spatial patch features from external and wrist views. Learnable queries aggregate each view through cross-attention, producing a joint 768-dimensional representation. The predictor conditions on three observed states, synchronized proprioception and executed actions, and a proposed five-step prefix:

$$\widehat{Z}_t^k = F_\theta(z_{t-2:t}, p_{t-2:t}, a_{t-2:t-1}, a_t^k) = (\widehat{z}_{t+1}^k, \ldots, \widehat{z}_{t+5}^k). \tag{2}$$

Here, z and p denote visual latents and proprioceptive states. Training minimizes latent prediction errors at horizons one, three, and five under the recorded actions, with SIGReg regularization (Balestriero & LeCun, 2025).

The diversity of predicted trajectories indicates how strongly candidate actions affect the predicted outcome. For K trajectories from the same starting state and with a common length $H = 5$, we use the equally weighted trajectory kernel and Vendi Score (Friedman & Dieng, 2023):

$$G_{ij} = \exp\left[-\frac{1}{2\sigma^2 H}\sum_{h=1}^{H}\|\widehat{z}_{t+h}^i - \widehat{z}_{t+h}^j\|_2^2\right], \tag{3}$$

$$S_t = \exp\left[-\sum_{r=1}^{K}\lambda_r \log \lambda_r\right], \qquad \lambda_r = \mathrm{eig}_r(G/K). \tag{4}$$

S_t measures the effective number of distinct predicted trajectories. It approaches one when candidates produce similar futures and increases as their consequences diverge. After PTA triggers the initial predictions, this diversity determines the interval before the next search (Appendix G).

4.3 ESTIMATING OPERATION COMPLETION

Let τ_q be the control steps from the current decision to completion of operation q. We define completion scores that combine probability and timing:

$$V(h_t, q) = \mathbb{E}\left[e^{-\tau_q/\beta}\mathbf{1}_{\text{complete}} \mid h_t, q\right], \tag{5}$$

$$Q(h_t, q, a) = \mathbb{E}\left[e^{-\tau_q/\beta}\mathbf{1}_{\text{complete}} \mid h_t, q, a, \widehat{Z}_t(a)\right]. \tag{6}$$

Both estimates refer to the same completion event and are measured from the current decision time. V uses observed history, whereas Q additionally conditions on the full candidate prefix and its predicted trajectory.

A shared causal Transformer with two layers, four heads, and hidden width 192 encodes up to sixteen observed latent states sampled every two control steps, followed by the candidate's five consecutive predicted states. Each token contains the latent, its change, relative time, observation/prediction type, operation type, and object query. Predicted tokens also include the corresponding action. The current-time hidden state outputs a completion-time distribution p_V from history; the candidate-end state outputs p_Q from history and the full candidate sequence.

Completion time is discretized into bins with upper limits $u_b \in \{0, 5, 15, 30, 60, 120, 240\}$ and an additional non-completion category. With $\beta = 60$ control steps, both scores are obtained as $\sum_b e^{-u_b/\beta} p(b)$, with zero utility for non-completion. Training accounts for completed operations, observed failures at the end of an attempt, and right-censored observations. If $\mathcal{B}_i$ is the set of time bins compatible with the observed outcome and duration, the loss is

$$\mathcal{L}_{\text{time}}(p_i, \mathcal{B}_i) = -\log \sum_{b \in \mathcal{B}_i} p_i(b). \tag{7}$$

We apply completion-time supervision to the current state, candidate sequence, and predictions at future time steps, together with auxiliary supervision for object control. Appendix A describes the labels and model architecture.

Object-conditioned comparison. To account for the object relations involved in an operation, we evaluate the nominal prefix and every candidate with a shared query set $\mathcal{Q}_t$. Task-specific roles are encoded using frozen language features and residuals relative to their within-task mean (Appendix A). We define candidate advantage as the smallest query-wise completion gain:

$$\Delta Q_t(a) = \min_{q \in \mathcal{Q}_t} \big[Q(h_t, q, a) - Q(h_t, q, a_t^0) \big]. \tag{8}$$

For example, a double-door operation includes separate queries for the left and right doors.

4.4 Intent-Constrained Action Refinement

Motion coordinates. We integrate the policy's intent segment into reference positions and orientations (p_h^0, R_h^0). Gripper-state changes and motion reversals divide the segment into phases. Within each phase, adjacent and later policy actions define local translation and rotation directions. These coordinates use the full intent segment to guide corrections to its first five actions.

We parameterize a correction with two six-dimensional deformation control points. Cubic splines interpolate translation and rotation offsets along the reference path, giving a twelve-dimensional search space:

$$p_h^\theta = p_h^0 + \delta p_h(\theta), \qquad R_h^\theta = \exp\big([\delta\omega_h(\theta)]_\times\big) R_h^0, \quad h = 1, \ldots, H. \tag{9}$$

The deformation starts at zero, and consecutive corrected poses are converted back into executable increments. Candidates retain the nominal gripper commands and follow the translation-direction constraints of the policy's motion. Policy-generated alternatives in each domain calibrate the deformation ranges and sampling covariances.

Continuity across corrections. For the active object or operation, D_t accumulates the pose-command offsets of executed corrections. The structural cost vector $C_t(a)$ contains three terms: path deformation in intent coordinates, changes to the first and last actions of the prefix, and the incremental cost of the accumulated offset after adding this correction. These terms penalize path distortion, discontinuities between action prefixes, and drift across replans. We update D_t after the selected five-step prefix has executed.

Candidate selection. Let s_Q be a domain-calibrated scale of completion-score differences among policy samples. The search objective is

$$J_t(a) = \frac{\Delta Q_t(a)}{s_Q} - \boldsymbol{\lambda}^\top \big[C_t(a) - C_t(a_t^0)\big], \quad \boldsymbol{\lambda} = (0.05, 0.05, 0.05), \tag{10}$$

$$a_t^\star = \arg \max_{a \in \{a_t^0\} \cup \mathcal{A}_t} J_t(a). \tag{11}$$

The objective balances estimated completion gain with the change in motion cost relative to the nominal prefix, whose objective is $J_t(a_t^0) = 0$. We execute a correction only if its objective is positive; otherwise, we retain the nominal prefix.

We optimize this objective with four CEM rounds (de Boer et al., 2005), sampling 64 corrections per round and updating the search distribution from the top eight elites. All candidates share the same observed history and query set. The final round's samples form $\mathcal{A}_t$. After executing $a_t^\star$, PREEMPT updates history with new observations, accumulates the executed command offsets, and obtains a fresh policy intent for the next replan. Algorithm 1 and Appendix B give the full search procedure.

5 EXPERIMENTS

We evaluate manipulation success, window localization, prediction quality, and inference cost in simulation and on a physical robot.

5.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on LIBERO (Liu et al., 2023), RoboCasa (Nasiriany et al., 2024), and an Agilex dual-arm robot. LIBERO comprises four suites—Spatial, Object, Goal, and LIBERO-10—with ten tasks per suite. RoboCasa covers eight pick-and-place tasks and six tasks involving doors or drawers. We run 50 trials per task, totaling 2,000 LIBERO trials and 700 RoboCasa trials, following each benchmark's initial-scene protocols and success criteria. Section 5.3 describes the four real-robot tasks.

Policies and baselines. We apply PREEMPT to $\pi_{0.5}$ (Physical Intelligence et al., 2025) on all three platforms and to OpenVLA-OFT (Kim et al., 2025) on LIBERO, keeping the base-policy weights fixed. We compare with each policy's default action-chunk length, and also test replanning every five steps (H=5) for $\pi_{0.5}$ on RoboCasa and OpenVLA-OFT on LIBERO. PREEMPT searches over smooth corrections using four CEM rounds of 64 candidates and executes a five-step prefix. Appendix Table 10 summarizes published results for related inference-time methods.

Metrics. We report task success with equal weight per task, relative failure reduction, model size, and inference time. Appendices C and D.2 describe training data and calibration.

5.2 SIMULATION RESULTS

LIBERO. PREEMPT improves $\pi_{0.5}$'s overall success from 96.7% to 97.1% (Table 1), reducing the remaining failures by 12.1%. It improves success on Object, Goal, and LIBERO-10 while matching the 98.8% Spatial baseline. The largest gain is on Object, from 98.8% to 99.6%, showing that local refinement can recover failures even when the base policy is already highly successful.

With OpenVLA-OFT, the largest gain is on LIBERO-10, from 95.4% to 96.4%. Spatial improves from 93.6% to 94.0%, Goal remains at 98.0%, and Object decreases from 99.0% to 98.4%.

RoboCasa. PREEMPT completes 480 of 700 trials, compared with 454 for the base policy, increasing success from 64.9% to 68.6% (Table 2). It improves pick-and-place, single-door, and drawer manipulation, with the largest category gain on single-door tasks, from 69% to 77%. Double-door success is slightly lower, at 59% versus 61%. Replanning every five steps alone achieves 319 successes; adding PREEMPT at the same execution horizon raises this to 480 and reduces failures by 42.3%. The gain therefore comes from selecting corrected actions, beyond the change in replanning frequency.

Table 1: Success rate (%) on LIBERO. Our evaluations use 50 trials per task; Overall averages the four suites.

Method	Spatial	Object	Goal	LIBERO-10	Overall
Diffusion Policy (Chi et al., 2023)	78.3	82.5	68.3	50.5	69.9
Octo (Octo Model Team et al., 2024)	78.9	85.7	84.6	51.1	75.1
GR00T N1 (NVIDIA et al., 2025)	93.8	97.6	92.6	90.8	93.7
OpenVLA (Kim et al., 2024)	84.4	88.4	78.6	54	76.4
$\pi_{0.5}$	**98.8**	98.8	96.2	93	96.7
$\pi_{0.5}$ + PREEMPT	**98.8**	**99.6**	96.4	93.6	**97.1**
OpenVLA-OFT	93.6	99	**98**	95.4	96.5
OpenVLA-OFT, H=5	93	98.2	96	93.6	95.2
OpenVLA-OFT + PREEMPT	94	98.4	**98**	**96.4**	96.7

Table 2: Successful trials on RoboCasa. Failure reduction (Fail. red., %) is relative to $\pi_{0.5}$ with H=5.

Method	Pick & Place	Single Door	Double Door	Drawer	Overall
$\pi_{0.5}$, H=50	253/400	69/100	**61/100**	71/100	454/700
$\pi_{0.5}$, H=5	198/400	37/100	14/100	70/100	319/700
$\pi_{0.5}$ + PREEMPT	**269/400**	**77/100**	59/100	**75/100**	**480/700**
Fail. red. (%)	35.1	63.5	52.3	16.7	42.3

5.3 REAL-ROBOT RESULTS

We deploy $\pi_{0.5}$ and PREEMPT on an Agilex dual-arm robot for four breakfast-preparation tasks (Figure 3). Each task uses one manipulation arm with a seven-dimensional action: six end-effector motion coordinates and one gripper command. *Load Toast* requires inserting bread into a toaster; *Start Toaster* requires pressing its lever; *Pour Coffee* requires pouring from a pitcher into a cup; and *Place Toast* requires placing toast on a plate. We conduct 50 trials per task and method.

PREEMPT increases overall success from 72.5% to 82.5%, a 10-point gain and a 36.4% reduction in failures (Table 3). The largest gains occur in Pour Coffee, from 22/50 to 31/50, and Load Toast, from 35/50 to 43/50. Start Toaster improves from 42/50 to 45/50, while Place Toast remains at 46/50.

5.4 WINDOW LOCALIZATION AND LATENT PREDICTION

Localizing physical decision windows. We evaluate PTA on 196 held-out RoboCasa episodes containing 319 reference events. PTA matches 266 events under positive temporal overlap, yielding 83.39% recall and 71.51% precision. Recall is 58.31% at an IoU threshold of 0.5. Figure 4 illustrates the predicted occupancy and decoded interval during a manipulation sequence. Appendix F reports boundary errors and additional localization metrics.

Predicting action consequences. On 96 held-out LIBERO-10 episodes, the world model reduces latent prediction MSE from 0.02124 to 0.00889 at three steps and from 0.05438 to 0.01514 at five steps relative to copying the last observation. Shuffling actions also increases prediction error at these horizons, showing that the model uses action information to distinguish future trajectories (Appendix Table 18). Copying the last observation has the lowest one-step error.

5.5 MODEL SIZE AND INFERENCE COST

Compact models and decision cost. PREEMPT adds 0.335B auxiliary parameters to the 3.617B base policy, a 9.3% increase. In a four-round search profile on two-moka-pot placement, evaluating 256 candidates takes 592.31 ms on average and 742.16 ms at the 95th percentile on an A800, including policy inference and observation processing. For context, DreamAvoid reports 2.133 s per intervention and its GPC-RANK comparison reports 0.694 s on an RTX 4090 (Fan et al., 2026). Figure 5 summarizes these reported costs and model sizes.

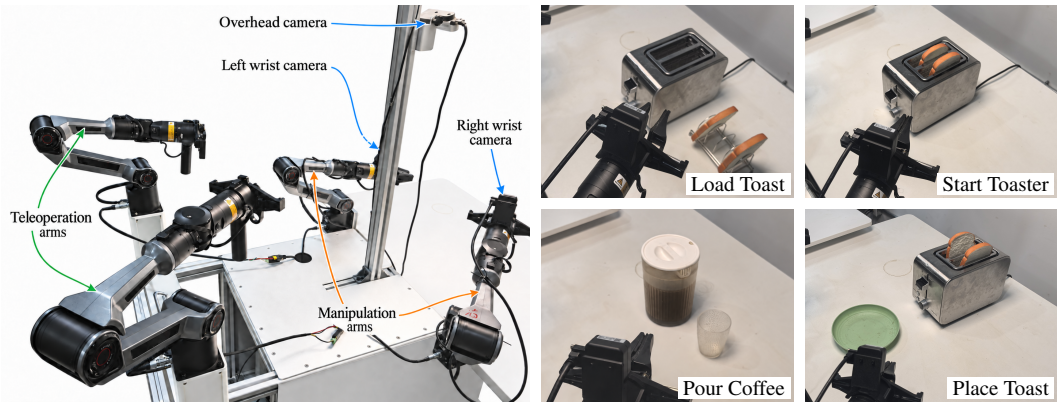

Figure 3: Agilex setup and four manipulation tasks

Table 3: Successful trials on the Agilex robot (50 per task). Failure reduction (Fail. red., %) is relative to $\pi_{0.5}$.

Method	Load Toast	Start Toaster	Pour Coffee	Place Toast	Overall
$\pi_{0.5}$	35/50	42/50	22/50	**46/50**	145/200
$\pi_{0.5}$ + PREEMPT	**43/50**	**45/50**	**31/50**	**46/50**	**165/200**
Fail. red. (%)	53.3	37.5	32.1	0.0	36.4

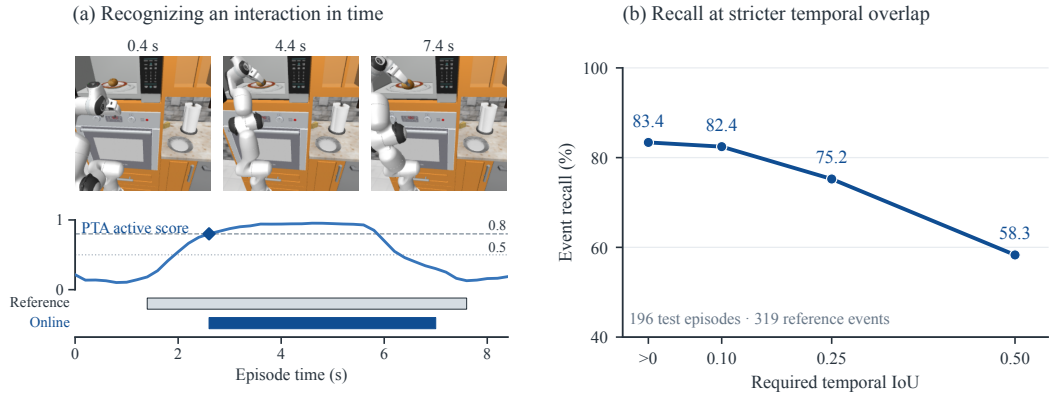

Figure 4: **Window localization.** (a) Observations, PTA scores, and reference and online windows during microwave-to-counter transfer. (b) Event recall at increasing temporal-IoU thresholds on 196 held-out episodes.

Window-guided computation. On 500 LIBERO-10 episodes, window-guided search uses 624.64 candidate evaluations per episode versus 1,502.72 for search at every replan, a reduction of 58.4% (Table 4). The corresponding success rates are 95% and 92%. Both settings use the same models, operation queries, search budget, and objective; Appendix D.2 describes the task scope. Across the four LIBERO suites, search occurs on 8.25% of control steps. Figure 6 illustrates this concentration of computation before object acquisition.

Table 4: Window-guided search on LIBERO-10 (500 episodes). Each search evaluates 256 candidates.

Search timing	Success (%)	Candidates/episode
Every replan	92.0	1,502.72
Physical decision windows	**95.0**	**624.64**

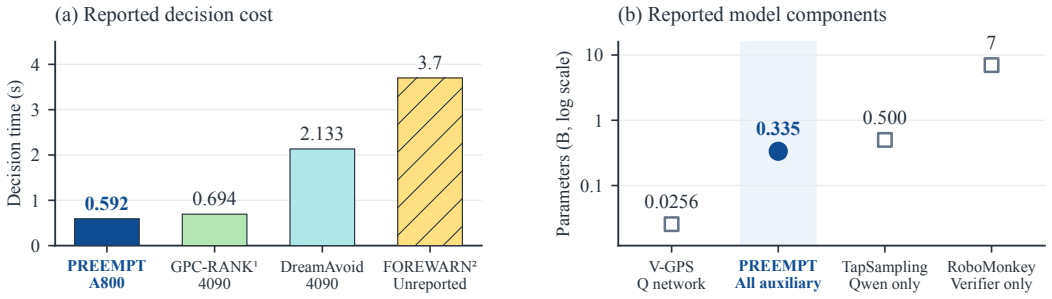

Figure 5: **Reported inference costs and model components.** (a) Decision costs on each study's hardware: [1]GPC uses DreamAvoid's implementation (both RTX 4090); [2]FOREWARN measures its steering core. (b) Parameters on a log scale: all auxiliaries for PREEMPT, the Q network for V-GPS, and language-model backbones for TapSampling and RoboMonkey.

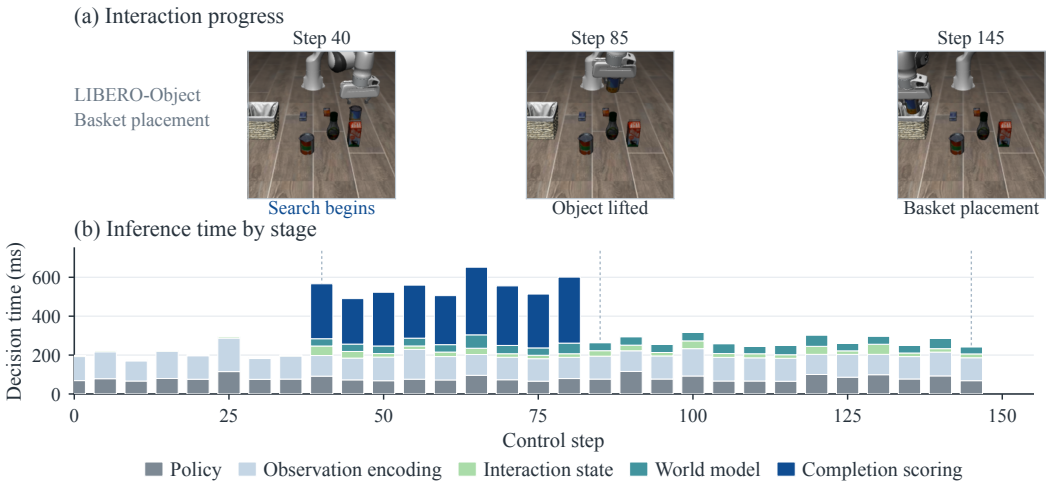

Figure 6: **Computation during object placement.** Observation frames align with the decision-time trace. Candidate scoring is concentrated before the lift, while policy inference and observation encoding continue throughout.

6 CONCLUSION

PREEMPT refines a fixed VLA's actions by concentrating search around physical interactions. Window anticipation and predicted consequence diversity guide when to search; completion-time evaluation connects short forecasts to operation outcomes; and intent-constrained optimization produces smooth corrections while accounting for earlier adjustments.

Experiments on LIBERO, RoboCasa, and an Agilex robot demonstrate action refinement with 0.335B auxiliary parameters, adding 9.3% to $\pi_{0.5}$'s size. Real-robot success increases from 72.5% to 82.5%. Window-guided search reduces candidate evaluations by 58.4% relative to searching at every replan in the LIBERO-10 comparison, with success rates of 95% and 92%, respectively.

AI USE STATEMENT

Generative AI assisted with manuscript editing, literature discovery, mathematical exposition, evaluation design, and figure and LaTeX preparation. The authors take responsibility for the final text, figures, and experimental results.

ETHICS STATEMENT

This work concerns predictive refinement of robotic manipulation. Physical deployment requires robot motion limits, emergency stopping, and operator supervision.

REPRODUCIBILITY STATEMENT

Sections 3–4.4 define the completion objective and action-refinement procedure. Appendices A–G describe model architectures, training data, calibration, evaluation protocols, and hyperparameters. Additional results include window-localization accuracy, latent prediction errors, and inference and memory measurements.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025. URL https://arxiv.org/abs/2511.08544.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2303.04137.

Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005. doi: 10.1007/s10479-005-5724-z.

Xianzhe Fan, Yuxiang Lu, Shenyuan Gao, Xiaoyang Wu, Ruihua Han, Manling Li, and Hengshuang Zhao. DreamAvoid: Critical-phase test-time dreaming to avoid failures in VLA policies. *arXiv preprint arXiv:2605.11750v1*, 2026. URL https://arxiv.org/abs/2605.11750v1.

Dan Friedman and Adji Bousso Dieng. The Vendi Score: A diversity evaluation metric for machine learning. *Transactions on Machine Learning Research*, 2023. URL https://arxiv.org/abs/2210.02410.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. URL https://arxiv.org/abs/2406.09246.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems*, 2025. URL https://arxiv.org/abs/2502.19645.

Jacky Kwok, Christopher Agia, Rohan Sinha, Matt Foutter, Shulu Li, Ion Stoica, Azalia Mirhoseini, and Marco Pavone. RoboMonkey: Scaling test-time sampling and verification for vision-language-action models. *arXiv preprint arXiv:2506.17811*, 2025. URL https://arxiv.org/abs/2506.17811.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791, 2023. URL https://arxiv.org/abs/2306.03310.

Chenyang Ma, Kai Lu, Guangyu Yang, Jiuming Liu, Shitong Xu, Bill Byrne, Ioannis Havoutis, Niki Trigoni, and Andrew Markham. CycleVLA: Proactive self-correcting vision-language-action models via subtask backtracking and minimum bayes risk decoding. *arXiv preprint arXiv:2601.02295v2*, 2026. URL https://arxiv.org/abs/2601.02295v2.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering your generalists: Improving robotic foundation models via value guidance. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4996–5013. PMLR, 2025. URL https://proceedings.mlr.press/v270/nakamoto25a.html.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*, 2024. URL https://arxiv.org/abs/2406.02523.

NVIDIA, Linxi Fan, Fanbo Xiang, et al. GR00T n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. URL https://arxiv.org/abs/2503.14734.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. URL https://arxiv.org/abs/2405.12213.

Amey P. Pasarkar and Adji Bousso Dieng. Cousins of the Vendi Score: A family of similarity-based diversity metrics for science and machine learning. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 3808–3816. PMLR, 2024. URL https://proceedings.mlr.press/v238/pasarkar24a.html.

Physical Intelligence, Kevin Black, Noah Brown, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Inference-time enhancement of generative robot policies via predictive world modeling. *arXiv preprint arXiv:2502.00622v4*, 2026. URL https://arxiv.org/abs/2502.00622v4. Revised version; accepted to IEEE Robotics and Automation Letters.

Yilin Wu, Thomas Tian, Gokul Swamy, and Andrea Bajcsy. From foresight to forethought: VLM-in-the-loop policy steering via latent alignment. In *Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.076. URL https://www.roboticsproceedings.org/rss21/p076.html.

Sizhe Zhao, Shengping Zhang, Shuo Yang, Weiyu Zhao, Shuigen Wang, and Xiangyang Ji. TapSampling: Inference-time sampling with a task-progress-understanding verifier for robotic manipulation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.25547.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024. URL https://arxiv.org/abs/2411.04983.

A Model Architectures and Training Objectives

A.1 Causal Window Anticipation

PTA anticipates upcoming PDWs and tracks their state using frozen robot-domain V-JEPA visual features, language, proprioception, gripper state, and executed actions. A projection layer and a unidirectional GRU predict window entry, active occupancy, and commit. We train on complete trajectories and local clips with

$$\mathcal{L}_{\text{window}} = \sum_{r \in \{\text{enter},\text{active},\text{commit}\}} \alpha_r \, \text{BCE}_{m_r}(\widehat{y}_r, y_r), \tag{12}$$

where m_r selects valid annotated times and α_r balances label frequencies. Repeated attempts are labeled as separate events.

The online decoder uses hysteresis and consecutive observations to stabilize window opening and closing, with boundary predictions refining entry and commit. The decoded window establishes an operation-specific correction context that is updated from subsequent observations. Table 19 lists the occupancy thresholds used in closed-loop evaluation.

A.2 Dual-View Spatial Prediction

A frozen ViT-B/14 backbone encodes external and wrist images into spatial patch tokens. Learnable queries aggregate the two views into a joint 768-dimensional latent. The dynamics model conditions on three observed times, aligned proprioception and actions, and a candidate action sequence to predict five future latent states. Proprioception has eleven dimensions; each action contains six end-effector displacement coordinates and a gripper command. Inputs use domain-specific coordinate transformations and normalization.

The world-model loss combines prediction at three horizons with representation regularization:

$$\mathcal{L}_{\text{WM}} = \frac{1}{3} \sum_{h \in \{1,3,5\}} \|\widehat{z}_{t+h} - z^{\text{obs}}_{t+h}\|_2^2 + \lambda_{\text{reg}} \mathcal{L}_{\text{SIGReg}}. \tag{13}$$

Predicted and observed states correspond to the same executed action sequence. SIGReg follows Balestriero & LeCun (2025). Multisource pretraining precedes domain adaptation. LIBERO sampling balances gripper states, small adjustments, and gripper switches; RoboCasa sampling balances data sources and tasks.

A.3 Completion-Time Prediction

The completion evaluator is a causal Transformer with two layers, four attention heads, and width 192. It receives up to sixteen observed latent states sampled every two control steps, followed by five consecutive predicted states and their actions. Tokens encode state, adjacent-state difference, relative time, observed or predicted status, object query, and operation type. Future tokens also include action features.

The current-state readout p_V attends only to observed history. The candidate-end readout p_Q attends to that history and the candidate's predicted sequence. Candidates share the history encoding and have separate future sequences. Each replan extends the history with new real observations.

For a known completion time, the compatible label set $\mathcal{B}_i$ contains the corresponding bin. An ended failed attempt is assigned to non-completion. A right-censored observation at time c_i is compatible with every bin extending beyond c_i, including the bin containing c_i, and with non-completion. Temporally uncertain labels use the bins intersecting their uncertainty interval.

The partial-label likelihood in Eq. (7) supervises current-state, candidate-end, and future-time readouts. The current-state and candidate-end distributions share the current decision as their time origin. An auxiliary object-control loss links the temporal representation to the queried interaction. Completion scores are the bin-weighted expectations in Table 5; they estimate discounted completion under the continuation behavior represented in training.

Table 5: Completion-time bins and discounted utilities ($\beta = 60$). Time is measured in control steps from the current decision.

Class	Remaining steps	Utility
0	Already complete	1
1	$(0, 5]$	$\exp(-5/60)$
2	$(5, 15]$	$\exp(-15/60)$
3	$(15, 30]$	$\exp(-30/60)$
4	$(30, 60]$	$\exp(-60/60)$
5	$(60, 120]$	$\exp(-120/60)$
6	$(120, 240]$	$\exp(-240/60)$
Non-completion	Beyond the modeled horizon	0

A.4 Task-Relative Object Queries

Let $e_i \in \mathbb{R}^d$ denote a frozen language embedding for an object role, and $\bar{e}$ the mean embedding of the task's query group. For multiple distinct roles, we combine task semantics with a normalized role residual:

$$\widetilde{e}_i = \sqrt{\frac{d}{2}} \left(\frac{e_i}{\|e_i\|_2} + \frac{e_i - \bar{e}}{\|e_i - \bar{e}\|_2} \right), \tag{14}$$

for nonzero denominators. The first term retains context; the second distinguishes roles such as object and destination or left and right door.

Object-control supervision contrasts roles under the same observation. During search, all candidates use the same query set $\mathcal{Q}_t$. The objective takes the minimum query-wise gain relative to the nominal prefix, so each candidate is evaluated against the same object relations.

A.5 Consequence Diversity

The RBF kernel in Eq. (4) compares trajectories with equal temporal weights. Its unit diagonal gives $\mathrm{tr}(G/K) = 1$, yielding $1 \le S_t \le K$. Diversity comparisons use the same candidate count, horizon, bandwidth, and sampling distribution.

An order-two measure from the generalized Vendi family (Pasarkar & Dieng, 2024) is

$$\mathrm{VS}_2 = \left(\sum_r \lambda_r^2 \right)^{-1} = \frac{K^2}{\|G\|_F^2}. \tag{15}$$

Applying the same measure to trajectory prefixes of lengths one through five characterizes how predicted consequences separate over the forecast horizon. The online schedule uses consequence diversity to choose the interval before the next search.

B Intent-Constrained Action Refinement

B.1 Search Procedure

At each scheduled search, PREEMPT optimizes a smooth correction around the policy's action sequence (Algorithm 1). PTA establishes the operation context, and consequence diversity sets the interval before the next search. Each search uses four CEM rounds with 64 candidates and eight elites per round.

Each search evaluates 256 sampled corrections. The nominal prefix provides the reference score. Selection considers the final round and the nominal prefix; earlier rounds update the sampling distribution. Candidate counts exclude nominal-reference evaluations, which are included in inference time.

Algorithm 1 PREEMPT: local action refinement

Require: Real history h_t, policy intent I_t, shared queries $\mathcal{Q}_t$, executed offset D_t
Require: World model F_θ, completion evaluator, calibrated search scales and costs
Require: An operation context and a scheduled search
Extract nominal prefix $a_t^0 = I_t[1:H]$ with $H = 5$
Build phase-local translation and rotation coordinates from all L intent steps
Initialize the 12-dimensional deformation distribution from domain calibration
Predict $\widehat{Z}_t^0$; cache $Q(h_t, q, a_t^0)$ for each $q \in \mathcal{Q}_t$
for $r = 1, \ldots, 4$ **do**
    Sample 64 deformation vectors; construct valid smooth prefixes $\mathcal{A}_t^{(r)}$
    Preserve nominal gripper commands and impose intent direction and path constraints
    Predict $\widehat{Z}_t(a)$ in a batch for $a \in \mathcal{A}_t^{(r)}$
    Evaluate $Q(h_t, q, a)$ for all candidates under the same $\mathcal{Q}_t$
    Compute $J_t(a)$ from Eqs. (8) and (10)
    Select the eight highest-scoring elites and update the CEM distribution
end for
Compute S_t from candidate trajectories and set the next search interval
Set $\mathcal{A}_t = \mathcal{A}_t^{(4)}$; compare it with $J_t(a_t^0) = 0$
Select the highest positive-objective correction, or a_t^0 if none is positive
Execute the selected full five-step prefix
Update real history, observed operation state, and the executed command offset D_t
Obtain a new policy intent at the next replan

B.2 REFERENCE GEOMETRY AND SMOOTH CORRECTIONS

We convert the complete intent segment into end-effector poses in the control frame. Gripper changes and motion-direction reversals define phases. Each prefix segment uses direction information from its corresponding phase, including later actions within that phase. Two six-dimensional control points specify translation and rotation offsets; cubic splines interpolate smooth corrections from a zero-offset starting point.

Corrections are expressed in local intent coordinates, transformed into the robot control frame, and converted back to action increments. Gripper commands remain fixed. Translation follows the intent's direction constraints, while domain-calibrated action statistics determine deformation ranges and covariance scales.

The cumulative offset D_t tracks executed pose-command corrections within an operation. Path, endpoint, and cumulative-offset costs discourage implausible deformations and drift across replans. We update D_t after executing the selected five-step prefix. It represents command-space corrections rather than an estimate of object pose.

Selection objective. Subtracting nominal costs gives $J_t(a_t^0) = 0$. The optimizer selects a correction with positive objective, retaining the nominal prefix otherwise. Because the objective combines completion gain and structural costs, reducing cumulative-offset cost can compensate for a small decrease in predicted completion utility.

B.3 POLICY ACTION HORIZONS

Table 6: Action horizons for each policy and benchmark. The intent tail supplies motion geometry beyond the five-step prediction horizon.

Policy / domain	Generated steps	Intent L	Executed H
$\pi_{0.5}$ / LIBERO	10	10	5
$\pi_{0.5}$ / RoboCasa	50	first 10	5
OpenVLA-OFT / LIBERO	8	8	5

The policies retain their pretrained or task-adapted weights throughout simulation and real-robot evaluation. Policy-specific action normalization and query encoders connect each policy to the auxiliary models. OpenVLA trajectories used for auxiliary training are described in Appendix C.

C Training Data and Calibration

We train PTA on annotated interaction intervals, the world model on executed action sequences and their observed futures, and the completion evaluator on local-operation outcomes.

C.1 Window Annotations

Table 7: PTA training data and episode splits. PDW counts refer to annotated events.

Component	Collected	Valid	PDW events
LIBERO local clips	4,800	3,390 clips	3,814
LIBERO complete trajectories	1,500	1,500 episodes	3,809
Total LIBERO events			7,623

Episode source	Train	Validation	Held-out
LIBERO complete trajectories	1,188	167	145
RoboCasa window annotations	1,012	210	196

LIBERO combines 3,390 valid local clips with 1,500 complete trajectories, yielding 7,623 valid physical decision windows. The complete trajectories contain 3,811 annotated events, of which 3,809 have valid boundaries; 1,397 trajectories contain multiple windows. They comprise 600 LIBERO-10, 350 Goal, 200 Object, and 350 Spatial episodes, collected from 600 $\pi_{0.5}$ and 900 OpenVLA rollouts.

The RoboCasa split contains 1,012 training, 210 validation, and 196 test episodes. Annotations include operation type, object identity, entry, active interval, commit, and validity. Video and control sequences are temporally aligned. Complete trajectories preserve intervals between windows and repeated attempts. Splits group clips by parent trajectory and repeated LIBERO rollouts by task and initial scene.

C.2 Operation-Completion Data

Completion labels use operation-specific physical evidence: stable object control and co-motion for acquisition, a sustained target relation after release for placement, and the required joint state for opening or closing. Failed attempts and censored observations receive distinct labels. Each training window pairs observed history, executed actions, predicted futures, object queries, and completion supervision.

Table 8: Completion-evaluator data. Temporal windows from the same episode share a partition.

Partition	Episodes	Temporal windows
LIBERO training	4,302	62,144
LIBERO validation	1,029	14,886
LIBERO total	**5,331**	**77,030**
RoboCasa training	1,272	35,491
RoboCasa validation	257	7,112
RoboCasa held-out	291	8,165
RoboCasa total	**1,820**	**50,768**

LIBERO provides 5,331 valid $\pi_{0.5}$ episodes covering all ten LIBERO-10 tasks and 12,840 local attempts. Training and validation contain 240 and 60 parent scenes, respectively. RoboCasa draws from 2,119 trajectories: 722 successful demonstrations, 697 $\pi_{0.5}$ rollouts, and 700 additional policy rollouts. Of these, 1,820 yield valid temporal windows spanning fourteen tasks and 602 task–role queries. Physical-state labels and video-derived labels retain their validity and temporal uncertainty. Sampling and loss weights are organized by episode and local attempt.

Table 9: Domain-adaptation data for the latent world model.

Domain	Episodes	Train	Validation	Held-out
LIBERO-10	979	774	105	100
RoboCasa	1,419	994	206	219

C.3 World-Model Adaptation

The LIBERO-10 data comprise 500 official demonstrations, 379 policy rollouts, and 100 OpenVLA rollouts. RoboCasa uses 722 demonstrations and 697 $\pi_{0.5}$ rollouts. Each sample aligns three historical observations and proprioceptive states with executed actions and their future observations. RoboCasa pairs a task-specific external view with a synchronized wrist view. State normalization maps both domains to the eleven-dimensional proprioceptive representation.

The completion evaluator shares its domain's visual representation with the world model. Predicted training states are conditioned on the executed action sequence and paired with the corresponding completion outcome. Evaluation tasks and initial scenes are described in Appendix D.2.

Search calibration. Policy samples from the same observed state determine path scales, deformation covariance, cumulative-offset scales, and completion-score normalization. Calibration uses the full intent segment and the local coordinates and operation conditions used during search. RoboCasa calibration covers fourteen tasks, 56 trajectories, 112 decision points, and 1,792 policy samples, with operation-specific statistics for grasping, opening, and closing.

D Evaluation Protocols and Related Results

D.1 Related Inference-Time Methods

Table 10 summarizes published $\pi_{0.5}$/LIBERO results. Each method is compared with the base policy used in its own study. DreamAvoid and its GPC-RANK comparison use 200 episodes per suite; TapSampling uses community $\pi_{0.5}$ weights and 50 trials per task on LIBERO-Long.

Table 10: Published $\pi_{0.5}$/LIBERO success rates under the original evaluation protocols.

Source	Evaluation scope	Own $\pi_{0.5}$ base (%)	Method (%)
DreamAvoid (DA-ABL) (Fan et al., 2026)	Four LIBERO suites (mean)	96.5	97.8
GPC-RANK in DreamAvoid (Fan et al., 2026)	Four LIBERO suites (mean)	96.5	96.6
TapSampling (Zhao et al., 2026)	LIBERO-Long	96.8	98.0

D.2 Evaluation Sets and Statistical Measures

Simulation comparisons use matched tasks and initial scenes. The principal aggregate is the success rate with equal task weights. Paired confidence intervals resample initial scenes within tasks while preserving method pairs. Unless otherwise stated, we use 20,000 bootstrap samples and report 95% intervals.

The main LIBERO evaluation uses 50 initial scenes per task, giving 2,000 episodes. RoboCasa uses 50 trials for each of fourteen tasks, giving 700 episodes. Real-robot evaluations use 50 trials for each of four tasks and each method.

The window-allocation comparison uses 500 LIBERO-10 episodes. Refinement is enabled for stove activation with pot placement, two-mug placement, and book placement (150 episodes); the other 350 episodes execute the nominal five-step prefix. Window-guided search and search at every replan share this task scope. The aggregate weights all ten tasks equally.

E Inference Cost and Model Size

E.1 Episode Inference Cost

Table 11 reports PREEMPT's cost over 2,000 LIBERO episodes. Inference time includes the policy, visual processing, PTA, world-model prediction, and completion evaluation. The matched comparison in Section E.3 includes observation processing for both methods.

Table 11: PREEMPT inference cost on LIBERO. Time is mean/P95 seconds per episode; search frequency is the fraction of control steps with candidate evaluation.

Method	Time (s)	Search (%)	Candidates/ep.
PREEMPT	68.23/126.02	8.25	3421.44

Across suites, PREEMPT's P95 latency is 1,174.8–1,966.8 ms per decision and 1,215.2–2,195.8 ms per search decision. Episode inference time includes all online processing, with shared encodings counted once.

E.2 Parameter and Memory Footprint

The auxiliary models add 0.335B parameters to the 3.617B policy, an increase of 9.26%. Their FP32 weights occupy 1.339 GB, compared with 7.234 GB for the BF16 policy. Tables 12 and 13 summarize total and component-wise weight storage.

Table 12: Parameter counts and weight storage. The policy uses BF16 and auxiliary models use FP32.

Model	Parameters (B)	Weights (GB)
$\pi_{0.5}$	3.617	7.234
Auxiliary models	0.335	1.339
Combined	3.952	8.573

Table 13: Auxiliary-model parameters and FP32 weight storage.

Component	Parameters (M)	Weights (MB)
V-JEPA target encoder	86.833	347.333
V-JEPA predictor	26.319	105.276
PTA trunk	19.864	79.456
PTA temporal readout	0.109	0.436
Action-conditioned world model	200.111	800.443
Completion evaluator	1.628	6.511
Total	334.864	1,339.454

The world model contributes 200.111M parameters and the online V-JEPA encoder and predictor contribute 113.152M. Weight storage is reported separately from measured GPU memory, which includes inference buffers and the simulator.

Figure 5(b) shows PREEMPT's full auxiliary parameter count alongside reported model components: the 25.6M Q network in V-GPS (Nakamoto et al., 2025), the 0.5B Qwen backbone in TapSampling (Zhao et al., 2026), and the 7B verifier backbone in RoboMonkey (Kwok et al., 2025). DreamAvoid uses a distilled student world model, while FOREWARN combines world-model features with a VLM.

Table 14 reports device-wide peak memory on an A800 for one matched initial scene in each of three tasks, with the methods run sequentially. PREEMPT increases peak memory by +3.55, +4.12, and +3.27 GiB for stove-and-pot, two-mug, and book placement, respectively. Device memory is sampled approximately once per second.

Table 14: Peak device memory on an A800 for three matched-scene examples.

Task	Base (GiB)	PREEMPT (GiB)	Increase (GiB)
Stove and pot	7.74	11.29	+3.55
Two-mug placement	7.74	11.86	+4.12
Book placement	8.18	11.45	+3.27

E.3 DECISION LATENCY

Tables 15 and 16 examine one task from each LIBERO suite, with 20 paired initial scenes per task. Decision latency excludes the first two warm-up replans; episode totals include them and the visual updates between replans. Both methods include observation processing in this comparison. All timings in this subsection measure inference, excluding simulation.

We profile PREEMPT with intent length $L = 10$, execution horizon $H = 5$, and four rounds of 64 candidates. In this profile, search intervals are independent of consequence diversity and corrections can continue after the initiating window. Across twenty initial scenes of two-moka-pot placement, 1,072 four-round searches evaluate 274,432 candidates. Including policy inference and observation processing, these decisions take 592.31 ms on average and 742.16 ms at P95. Giving equal weight to the twenty scenes yields 595.04 ms, with a scene-bootstrap 95% interval of [580.27, 610.75] ms.

Literature context. Figure 5(a) summarizes reported costs for predictive action refinement. DreamAvoid (Fan et al., 2026) reports 2.133 s per eight-proposal intervention on an RTX 4090 for proposal generation, video prediction, and value scoring, excluding its asynchronous trigger. Its GPC-RANK comparison takes 0.694 s for sampling, world-model prediction, and reward prediction over eight proposals. FOREWARN (Wu et al., 2025) reports 3.7 s for world prediction, narration, and evaluation of six action modes obtained from 100 plans. This steering-core measurement is hatched; the GPU and inclusion of initial sampling and clustering are unspecified.

Table 15: Decision latency on four LIBERO tasks (20 paired episodes each). WM/CEM gives the fraction of replans using prediction/search.

Task	Method	Replans	Mean (ms)	P95 (ms)	WM/CEM (%)
LIBERO-10	Base	1,829	84.11	115.60	–
	PREEMPT	1,829	457.00	713.87	81.9 / 58.6
Goal	Base	666	85.40	112.44	–
	PREEMPT	603	227.12	311.46	42.6 / 0.0
Object	Base	631	83.48	109.84	–
	PREEMPT	782	316.41	603.31	63.2 / 22.0
Spatial	Base	336	86.15	119.93	–
	PREEMPT	357	393.21	655.64	63.9 / 52.9

Table 16: Episode inference time on the same four tasks, reported as mean/P95 seconds.

Task	Base mean/P95 (s)	PREEMPT mean/P95 (s)
LIBERO-10	8.95/11.34	93.60/111.70
Goal	4.20/9.69	23.65/50.45
Object	3.93/5.72	34.04/87.05
Spatial	2.80/3.26	17.07/22.39

Using each initial scene as a paired unit, the mean latency increases are 373.71 ms [347.52, 398.44] for LIBERO-10, 139.43 ms [129.00, 149.62] for Goal, 229.55 ms [215.90, 243.26] for Object, and 302.83 ms [281.22, 325.34] for Spatial. In the Goal example, none of the 257 predictions leads to search because no eligible control query is active.

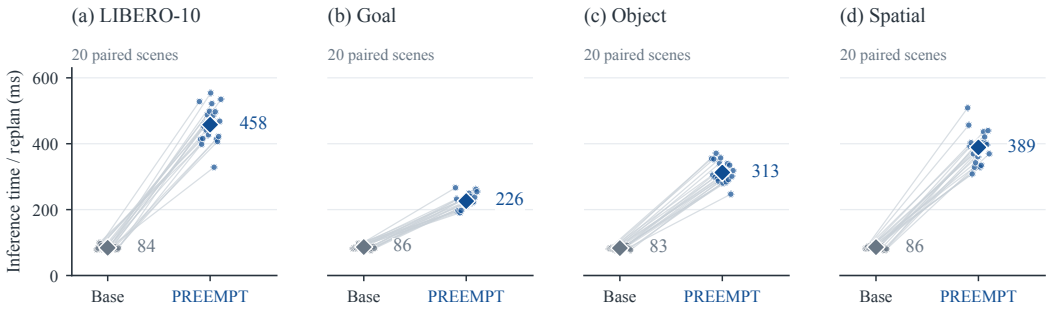

Figure 7: Decision latency for 20 paired initial scenes in each of four LIBERO tasks. Small points average replans within each episode; lines connect the base policy and PREEMPT from the same initial scene. Diamonds and labels give the mean across the 20 scenes. All panels share the same vertical scale.

F WINDOW LOCALIZATION AND PREDICTION QUALITY

F.1 WINDOW LOCALIZATION AND DYNAMICS

Tables 17 and 18 evaluate temporal window localization and latent prediction. Window metrics use the RoboCasa Core-14 three-view model with 210 validation and 196 test episodes and model-generated PDW reference labels; uncertain intervals are masked. Thresholds are selected on validation episodes only: occupancy 0.8/0.5, one opening and four closing confirmations, and edge threshold/lookback 0.8/2. The closed-loop thresholds are given separately in Table 19. World-model errors use 320 prediction points from 96 held-out LIBERO-10 episodes.

Table 17: PTA window-localization accuracy.

Module	Metric / sample	Value
Windows	Event recall / 319 labeled events	83.39% (266/319)
	Occupancy IoU / valid annotated time	50.27%
	Online opening signed delay / 133 nearby matches	$+0.453$ s mean ($n = 133$)
	Refined entry/commit MAE / 133 nearby matches	0.809/0.717 s MAE
	Fragmentation / 319 labeled events	3.45% (11/319)
	Zero-overlap openings / 372 predictions; inactive occupancy	11.56% (43/372); 25.76% time

Window precision, recall, and F1 are 71.51%, 83.39%, and 76.99%, respectively, from 266 true positives, 106 false positives, and 53 false negatives under one-to-one positive-overlap matching. Recall at occupancy-IoU thresholds 0.10, 0.25, and 0.50 is 82.45%, 75.24%, and 58.31%. Timing uses 133 complete matched windows whose refined boundaries are each within ten 5-Hz ticks of the reference. Opening-delay MAE/median is 0.853/0.800 s; refined entry and commit MAE/median are 0.809/0.800 s and 0.717/0.600 s. These errors are differences in sampled event time rather than model execution latency.

The example in Figure 4 is selected at the median IoU of the matched refined intervals (0.645). Its online bar instead shows the causal opening and closing confirmations used during execution.

The action-conditioned world model improves over copy-last and shuffled-action prediction at horizons three and five (Table 18); copy-last has the lowest one-step error. Episode-cluster bootstrap intervals for the model's H1/H3/H5 errors are [0.003099, 0.004170], [0.007424, 0.010646], and [0.012550, 0.018223]. The corresponding model-minus-copy-last intervals are $[-0.000590, +0.000734]$, $[-0.017125, -0.008343]$, and $[-0.050129, -0.029909]$.

Table 18: Latent prediction MSE at one-, three-, and five-step horizons on 96 held-out LIBERO-10 episodes.

Predictor	H1	H3	H5
Copy last observation	**0.003498**	0.021241	0.054379
Shuffled actions	0.004905	0.021667	0.055442
Action-conditioned world model	0.003596	**0.008894**	**0.015141**

G HYPERPARAMETERS AND METHOD SCOPE

Table 19 collects the model and search settings used throughout the method. Policy-dependent action horizons appear in Table 6. Auxiliary-model architecture and training objectives are described in Appendix A.

Table 19: Model and search hyperparameters.

Parameter	Value
Prediction and execution horizon	5 control steps
Intent length	10 for $\pi_{0.5}$; 8 for OpenVLA-OFT
World-model history	3 observations
Visual latent dimension	768
Proprioception / action dimensions	11 / 7
Completion Transformer	2 layers, 4 heads, width 192
Completion history	Up to 16 observations, stride 2
Completion-bin endpoints	0, 5, 15, 30, 60, 120, 240 steps
Discount scale β	60 steps
Window occupancy thresholds	0.5 / 0.4 (opening / closing)
Window confirmation	2 rising / 1 falling observations
Deformation parameters	Two 6-D control points
CEM population / elites / rounds	64 / 8 / 4
Structural cost weights	0.05 each
Adaptive search intervals	5, 10, or 15 control steps

Method scope. PREEMPT refines end-effector motion around an existing policy intent while preserving gripper commands. Its correction range is therefore determined by the policy's plan and the permitted deformation. The cumulative-offset penalty controls drift across replans, and the window schedule concentrates search around physical interactions.

Completion-time prediction extends beyond the five-step visual forecast using learned continuation outcomes. Its reliability depends on world-model fidelity and completion calibration in the target domain.