WHERE REFINEMENT PAYS: MEASURING AND ALLOCATING SAMPLING COMPUTE IN FLOW-MATCHING MANIPULATION POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-matching manipulation policies typically use the same number of function evaluations (NFE) at every call, assigning the same sampling budget to free-space motion and precision insertion. Does each decision benefit from the same amount of computation? Whole-episode comparisons cannot isolate the value of one budget choice because changing the budget also changes the states visited afterward. We introduce counterfactual state branching, a testbed that restores exact simulator states and policy histories, matches sampler noise, and varies the budget at selected calls. All branches then use the same continuation budget. We compare terminal success and cumulative NFE from the intervention to termination. Across four flow-matching policies on nine VLA-Arena suites, budget responses vary strongly with the policy and decision state. Using more computation throughout an episode often fails to improve success and can reduce it. We use these paired success and cost outcomes to train FlowBudgetAllocator (FBA), which operates across suites while keeping the action policy frozen. In simulation, it achieves higher observed success with fewer NFE than the default high budget. On basic configurations, it improves task success over the best fixed budget with only a modest increase in NFE. FBA exhibits distinct budget-selection patterns across tasks and execution phases. On the real robot, it uses substantially lower sampling budgets than the default high-budget policy without reducing observed task success. By measuring the consequences of individual budget choices, we turn sampling computation into a resource to allocate during control.

1 INTRODUCTION

In flow-matching image and video generation, larger sampling budgets are commonly used to improve sample quality by increasing the number of function evaluations (NFE) used to integrate the learned vector field (Esser et al., 2024; Liu et al., 2023b; Jin et al., 2025). Flow-matching robot policies (Black et al., 2024; 2025a; Shukor et al., 2025; Yuan et al., 2026) inherit this intuition. They generate action chunks with a budget chosen at deployment and reused at every call. Yet each sampled action in robot control changes the physical state and the conditions of the next policy call. A uniform budget gives a gripper moving through free space and one inserting a key the same amount of refinement.

Fixed-budget results challenge this inherited intuition. Across four widely used flow-matching manipulation policies on VLA-Arena (Zhang et al., 2026) and LIBERO (Liu et al., 2023a), Figure 1(a) shows that success does not increase monotonically with NFE. Observed success peaks at different budgets across policies and evaluation settings. This contrast also appears within an episode. Figure 1(b) shows the stakes of a single call in a task demanding precise alignment: from a shared state with matched noise, reducing that call's NFE from 10 to 1 changes timeout to success.

Yet knowing which fixed budget performs better does not reveal where its computation helps or hurts. Changing the budget changes the actions and, in turn, the states visited by later policy calls; whole-episode success rates compare these different executions. This raises two questions:

Where does sampling computation help or hurt task success?
Can measured budget effects guide sampling-budget allocation?

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.

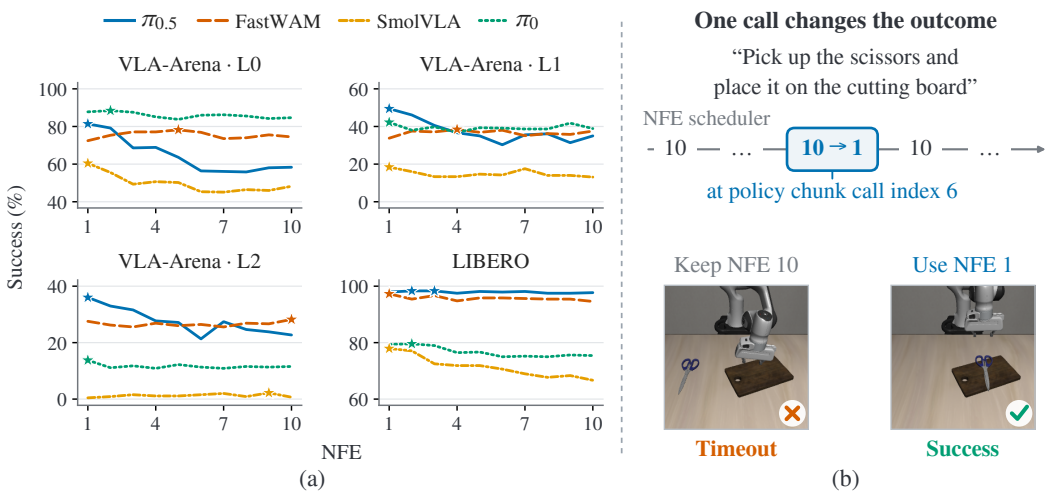


Figure 1: **Increasing NFE does not guarantee higher task success.** (a) Fixed-budget success for four policies on nine VLA-Arena suites at L0/L1/L2 and four LIBERO suites. Panels use different success scales; stars mark each curve's observed maxima over the tested budgets. (b) One matched example with $\pi_{0.5}$ on Cautious Grasp L0: using NFE 1 instead of NFE 10 only at call index 6 changes the recorded outcome from timeout to success. Both branches start from the same state with matched sampler noise; all other calls use NFE 10. Terminal frames show the timeout branch after call 29 and the successful branch after call 23. Call indices in this example are zero-based.

To answer the first question, we introduce *counterfactual state branching*, the controlled closed-loop testbed illustrated in Figure 2. It restores exact simulator states and policy histories and matches sampler noise across budgets. We vary the budget at calls selected by physical events, such as entering the approach region or establishing contact. After the intervention, all matched branches use the same budget until termination. These comparisons map budget responses across decision states and quantify their effects on terminal success and remaining computation.

The paired success and cost outcomes both measure budget value and supervise FlowBudgetAllocator (FBA), which learns budget choices with the action policy frozen. Once deployed, its choices reshape later states and decisions. We therefore test both the resulting closed-loop performance and how the value of a choice changes with its position and subsequent budget selections.

Our contributions are as follows: (i) **Counterfactual state branching.** We introduce a testbed for measuring budget effects and generating FBA training data. (ii) **Response maps and value measurements.** Budget-response maps and success-value differences ΔQ show that budget effects vary strongly with the policy and the decision state, and that uniformly higher budgets often fail to improve closed-loop success or degrade it. Expert-state probes characterize changes in mean-action agreement and sampling variance alongside numerical refinement. (iii) **Adaptive sampling-budget allocation.** On L0, FBA improves observed success over the best fixed budget using additional NFE. A choice's value depends on later budget selections. Real-robot adaptation lowers sampling budgets and call latency, with higher observed success than fixed 10 across all three scenes.

2 Counterfactual State Branching

2.1 Policies and Budgets

Given an observation and language instruction, a flow-matching policy integrates a learned vector field from noise to generate an action chunk, whose prefix is executed before replanning. A decision *state* contains simulator and controller states, policy history, call index and remaining episode time. Budget k counts vector-field evaluations per call, with finite candidates $\mathcal{K} = \{k_1, \ldots, k_M\} \subset \mathbb{N}_{>0}$. We use native task success and episode cost ΣNFE.

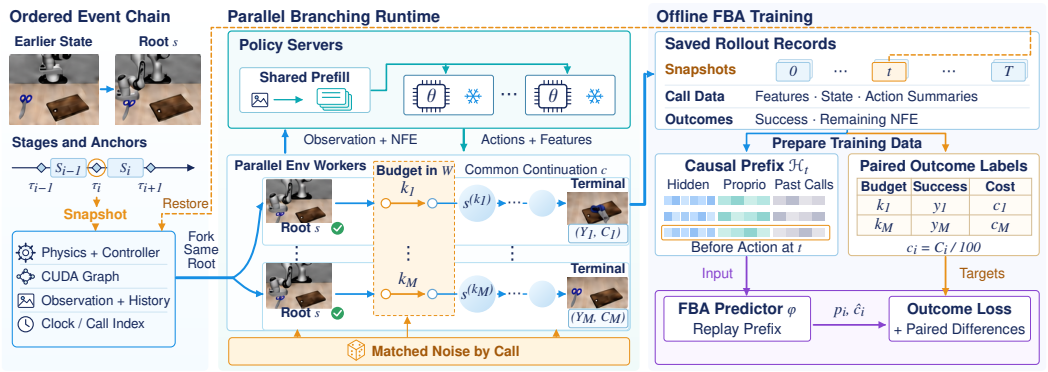


Figure 2: **Matched budget branches provide measurements and supervision for FBA.** Event onset times (anchors) τ_i delimit stages S_i and locate branch roots. Parallel workers restore the same decision state and policy history, apply candidate budgets within intervention window W, and continue at a common budget c with sampler noise matched by call index. Policy servers reuse the saved state's prefill encoding. Rollout records contain snapshots, call data and outcomes. Any saved snapshot can be restored with its policy history for matched branching via the dashed return path at left. Call data contain policy features, robot state and executed-action summaries, from which decision-time prefixes $\mathcal{H}_t$ are formed for offline training. Each prefix is paired with terminal-success labels $y_i = Y_i$ and normalized remaining costs $c_i = C_i/100$, where C_i counts NFE from intervention to termination. FBA replays each prefix and fits parameters ϕ at its final call to these outcomes and their within-root differences. Future outcomes supply targets only. Snowflakes mark the frozen policy; scene insets and feature matrices are schematic.

A *stage* is an execution phase between successive physical events. A *transition* is a local interval at initialization or around a stage boundary. We denote these by S_i and T_i, respectively.

2.2 Matched Budget Interventions

We branch from a saved decision state (the root) on a reference rollout without budget interventions. Each candidate $k \in \mathcal{K}$ starts from the same restored state and policy history. Each branch uses k at calls $u, \ldots, v-1$ in window $W = [u, v)$, then a common budget $c \in \mathcal{K}$ until termination. The reference fixes both endpoints; native termination may end a branch earlier. Sampler noise is matched by zero-based call index.

Subsequent actions respond to each branch's observations.

Parallel environment workers run the branches through policy servers, reusing the shared observation and instruction's cached prefill encoding. Before branching, we verify the restored state, observation, success flag, and policy history against the reference using the checks in Appendix B.2.

$Y_s(k, c; W) \in \{0, 1\}$ records terminal success; $C_s(k, c; W)$ records cumulative NFE from saved state s through termination, including the window and continuation. For reference budgets $c^- < c^+$ in $\mathcal{K}$, define

$$\Delta_s^{\text{down}}(k; W) = Y_s(k, c^+; W) - Y_s(c^+, c^+; W), \tag{1}$$

$$\Delta_s^{\text{up}}(k; W) = Y_s(k, c^-; W) - Y_s(c^-, c^-; W). \tag{2}$$

Downgrading applies $k < c^+$ to states visited by the high-budget reference policy; upgrading applies $k > c^-$ to states visited by the low-budget reference policy. Each comparison keeps its continuation budget fixed.

2.3 Stages, Transitions, and Branch Selection

We detect four ordered physical events on each reference trajectory: *free-space onset*, *approach-region entry*, *contact establishment*, and *goal attainment*. Their first qualifying onset times τ_i de-

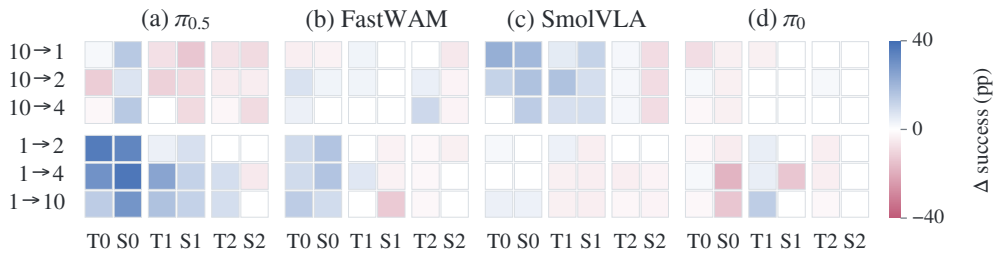


Figure 3: **Budget effects across stages and transitions in Cautious Grasp (VLA-Arena, L0).** Colors show paired terminal-success differences (pp) from the corresponding fixed-budget reference; positive values favor the intervention. Rows $b \to k$ use k within the indicated window and b thereafter (upper: $b = 10$; lower: $b = 1$). Columns alternate transition windows T_i and stages S_i along the reference trajectory. Windows can overlap and include different states.

limit S_0, *free-space approach*, S_1, *contact acquisition*, and S_2, *post-contact manipulation*. The transitions are T_0 at episode initialization, T_1 around approach-region entry, and T_2 around contact establishment. These event times align the saved states used for measurement and training by physical progress. Event rules are fixed before evaluating budget outcomes, with detector and window definitions in Appendix B.1. These labels are not FBA inputs.

3 Measurements of Budget Responses

3.1 Experimental Configuration

We simulate in MuJoCo (Todorov et al., 2012). LIBERO shows a partial ceiling effect for $\pi_{0.5}$ and FastWAM (roughly 95–98% success across budgets). Subsequent simulation experiments therefore use nine VLA-Arena suites with precision-contact scenarios and controlled task and scene variations. VLA-Arena uses L0 for fine-tuning and defines L1 and L2 through progressively larger task and scene variations (Zhang et al., 2026).

We study π_0, $\pi_{0.5}$, SmolVLA and FastWAM using Euler integration, $\mathcal{K} = \{1, 2, 4, 10\}$ and $(c^-, c^+) = (1, 10)$ for paired interventions. We also evaluate every integer budget from 1 to 10, using the same tasks and initializations within each policy and evaluation setting. We fine-tune all parameters of the four pretrained models on L0 for 30,000 steps.

3.2 Fixed Budgets

Figure 1(a) shows non-monotonic fixed-budget responses on both LIBERO and VLA-Arena. On VLA-Arena, NFE 1 attains the highest observed success in most policy–level combinations, while FastWAM peaks at intermediate budgets on L0/L1.

For $\pi_{0.5}$ on Unseen Objects L0, NFE 1/2 shows greater physical progress than NFE 4/10 by call 9, as measured in Appendix C.1.5.

3.3 Budget Responses across Stages and Transitions

We compare budget interventions across stages and transitions of Cautious Grasp. The policy must pick the target object without disturbing its surroundings, moving from a free-space approach to a precision-critical grasp. In Figure 3, raising NFE from 1 to 4 ($1 \to 4$) for $\pi_{0.5}$ gives $+37.5$ percentage points (pp) during free-space approach (S_0) and -5.3 pp during post-contact manipulation (S_2). The opposite signs motivate context-dependent budget choices.

3.4 Measuring the Value of an Intervention Window

A budget choice can change both sampled actions and the states visited afterward. From saved state s, using budget k within window W and continuation c gives expected terminal success and

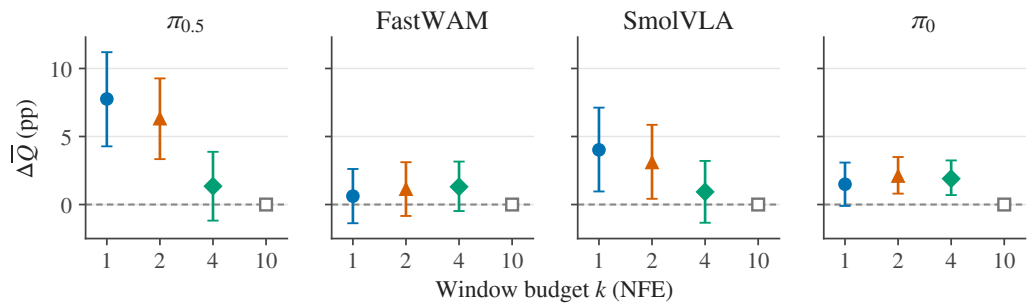


Figure 4: **Budget-dependent value across stage windows.** Mean success changes $\Delta\bar{Q}_{10,W}(k, 10)$ (pp) for four policies across nine L0 suites, using k throughout W and NFE 10 thereafter, relative to matched NFE 10 branches. Each policy uses common windows with outcomes for all budgets; suites are equally weighted. Bars show pointwise 95% reference-cluster bootstrap intervals. Windows may span multiple calls; $k = 10$ is zero by construction.

remaining cost

$$Q_{c,W}(s, k) = \mathbb{E}[Y_s(k, c; W)], \quad C_{c,W}(s, k) = \mathbb{E}[C_s(k, c; W)]. \tag{3}$$

Expectations cover sampler and rollout randomness. Relative to budget k_0, define $\Delta Q_{c,W}(s; k, k_0) = Q_{c,W}(s, k) - Q_{c,W}(s, k_0)$. For a distribution d over eligible state–window pairs, we measure

$$\Delta\bar{Q}_c(k, k_0) = \mathbb{E}_{(s,W)\sim d}[\Delta Q_{c,W}(s; k, k_0)], \tag{4}$$

with analogous remaining-cost differences, which also reflect changes in episode length. Each pair supplies one observed outcome per budget.

Figure 4 aggregates stage-window comparisons across nine L0 suites using fixed-10 references and continuation. For $\pi_{0.5}$, NFE 1 gains 7.76 pp in observed success and saves 56.85 remaining NFE relative to NFE 10. Appendix C.1.1 gives the state-transition formulation and population estimates.

Single-call results appear in Appendix C.1.2. Matched two-call extensions in Appendix C.1.3 show that extending an upgrade need not preserve its benefit.

3.5 Numerical Fidelity and Task Value

The numerical checks in Appendix C.2.1 show that NFE 10 means are closer than NFE 1 means to the finite NFE-32 reference, without necessarily improving agreement with demonstrations.

To examine changes in both mean and variability, we probe four checkpoints at 1,800 fixed expert observations each, using eight shared sampler seeds across budgets on six VLA-Arena suites. Comparisons use the same ten-step, seven-coordinate action prefix. At an expert state s, let $a_k(s, \xi)$ be the budget-k action block under sampler noise ξ, with mean $\mu_k = \mathbb{E}_\xi a_k$. We vectorize action blocks and use Euclidean norms.

Sampling variability. Sampling variance is $\mathcal{V}_k = \mathbb{E}_\xi\|a_k - \mu_k\|^2$. Figure 5(a) shows that variance responds differently across policies and action components. For $\pi_{0.5}$, the mean within-frame NFE–position-variance correlation is 0.730; the weighted fraction of frames with nondecreasing variance is 80.5%. SmolVLA also shows a positive position trend, whereas FastWAM's is negative. Rotation variance decreases with budget on average in all four checkpoints. Lower budgets therefore yield more concentrated position actions in many tested $\pi_{0.5}$ states, while other components can respond in the opposite direction. Appendix C.2.2 defines these fixed-state sampling statistics and compares them with full-block averages.

Mean-action alignment and task success. For demonstrated block e, the mean-action alignment gain from replacing budget k_2 with k_1 is

$$G_{\text{align}}(s; k_1, k_2) = \|\mu_{k_2} - e\|^2 - \|\mu_{k_1} - e\|^2. \tag{5}$$

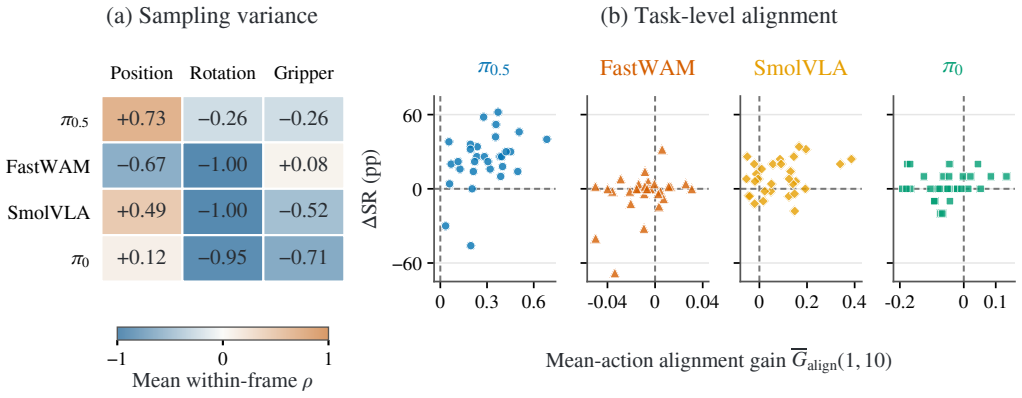


Figure 5: **Sampling variability and mean-action alignment across budgets.** Expert probes use 1,800 frames across six VLA-Arena suites and eight shared sampler seeds per checkpoint. (a) Within-frame Spearman correlations between NFE and action-component variance over $\{1, 2, 4, 10\}$, averaged within episodes, tasks and suites. (b) Mean-action alignment gain $\bar{G}_{\text{align}}(1, 10)$ versus fixed-budget success change $\Delta\text{SR} = \text{SR}(1) - \text{SR}(10)$ (pp), with 30 task means per checkpoint. Positive values on either axis favor fixed 1. Panels share a vertical scale but use model-specific horizontal ranges. Probes and rollouts are sampled separately.

Positive gain favors k_1 in mean-action agreement. Within $\pi_{0.5}$ and SmolVLA, tasks with larger mean-action alignment gains $\bar{G}_{\text{align}}(1, 10)$ also tend to show larger success gains from budget reduction in Figure 5(b). The pattern is less ordered for π_0, while FastWAM's task-level success changes cluster near zero with several large losses. Appendix C.2.3 specifies the finite-seed estimates.

4 Learning Budget Allocation

We study FBA on $\pi_{0.5}$, whose task-level alignment gains tend to accompany budget-reduction gains in Figure 5(b), while upgrades have opposite effects during approach and after contact in Figure 3.

4.1 Learning from Matched Budget Outcomes

We train FBA by *counterfactual distillation* using the single-call case of Section 3.4. Fixed NFE 1 gives $\pi_{0.5}$ the highest observed success among tested fixed budgets on VLA-Arena L0 in Figure 1(a). We therefore set $c_{\text{train}} = 1$. At each event root at call t, matched branches with $W = [t, t + 1)$ supply labels $y_i = Y_s(k_i, c_{\text{train}}; W)$ and normalized remaining cost $c_i = C_s(k_i, c_{\text{train}}; W)/100$ for each $k_i \in \mathcal{K}$. These comparisons supervise when to retain NFE 1 and which higher budget to select. All candidates share a causal prefix $\mathcal{H}_t$: frozen encoder features, proprioception and preceding executed-call summaries, ending before the root's action. Event identities are excluded.

From 2,250 fixed-1 L0 reference episodes of $\pi_{0.5}$, the *ordinary* corpus contains 6,748 eligible event roots. To include states reached after changing the budget, *mixed* adds event roots and their executed successors, yielding 17,990 examples.

A recurrent predictor with parameters ϕ maps $\mathcal{H}_t$ to success value predictions $p_i = \widehat{Q}_\phi(\mathcal{H}_t, k_i)$ and normalized remaining costs $\widehat{c}_i$. Each training root has labels for all $M = |\mathcal{K}| \geq 2$ candidates. We supervise the final call of its prefix with

$$L_{\text{sim}} = \frac{1}{M} \sum_{i=1}^{M} \left[\ell_{\text{BCE}}(p_i, y_i) + (\widehat{c}_i - c_i)^2 \right] + 0.5 L_{\text{pair}}, \tag{6}$$

$$L_{\text{pair}} = \frac{2}{M(M-1)} \sum_{i<j} \left[(\Delta\widehat{Q}_{\phi,ij} - \Delta y_{ij})^2 + (\Delta\widehat{c}_{ij} - \Delta c_{ij})^2 \right]. \tag{7}$$

Here $\Delta\widehat{Q}_{\phi,ij} = p_i - p_j$ and $\Delta a_{ij} = a_i - a_j$ for labels and costs. Since $\mathbb{E}[\Delta y_{ij} \mid s] = \Delta Q_{1,W}(s; k_i, k_j)$, paired outcomes supervise value differences. Separate outputs support deployment cost weights λ without retraining. From ensemble means $\bar{p}_i, \bar{c}_i$ of the predictions $p_i, \widehat{c}_i$, let r index $k_{\min} = \min \mathcal{K}$. We select $k^* = k_{i^*}$ with $i^* = \arg\max_i[(\bar{p}_i - \bar{p}_r) - \lambda(\bar{c}_i - \bar{c}_r)]$, using $k_{\min}$ when $\max_i \bar{p}_i - \min_i \bar{p}_i < 0.01$. The frozen policy supplies encoder features for $\mathcal{H}_t$ and cached sampler prefill; k^* sets NFE for chunk A_t. Appendix D.1 specifies data construction, architecture and optimization.

At deployment, FBA applies these fixed-1 outcome predictions at each call using the updated causal history. We evaluate the resulting allocation in full episodes and examine call-position and continuation effects in Appendices D.4.1 and D.4.2.

4.2 Offline Adaptation on the Robot

Simulation supplies matched outcomes for all candidate budgets at a restored root; real-robot rollouts supply outcomes for the budgets actually executed. We collect fixed-budget rollouts at NFE 1/2/4/10 and human-in-the-loop (HIL) rollouts that increase NFE from 1 to 2/4/10 or decrease it from 10 to 1/2/4. An operator selects a grasping moment in scenes A/C or a placement moment in B, changes the budget for one call, and then restores the background budget. These conditions yield 500/300/300 collection rollouts in A/B/C.

For a T-call rollout, each causal prefix $\mathcal{H}_t$ is paired with the executed budget k_t, terminal success y, and normalized remaining cost $c_t^{\mathrm{H}} = \sum_{j=t}^{T-1} k_j/100$. For the executed candidate i, with $k_i = k_t$, the per-call outcome loss is

$$\ell_{\mathrm{H},t} = \ell_{\mathrm{BCE}}(p_{t,i}, y) + (\widehat{c}_{t,i} - c_t^{\mathrm{H}})^2. \tag{8}$$

We average first over calls within each episode, then over episodes. We omit L_{pair} because physical rollouts provide no matched within-state budget pairs. Labels describe outcomes under the actual budget sequence; future information enters only the labels.

Starting from the simulation-trained three-member FBA ensemble, we fine-tune all three members offline for each scene under the protocol in Appendix D.5.3, keeping the action policy frozen. During evaluation, each scene's adapted ensemble selects budgets autonomously with $\lambda = 2$ and its parameters fixed.

4.3 Simulation Evaluation

Each FBA ensemble is shared across nine VLA-Arena suites, with weights fixed across L0/L1/L2. L0 closed-loop evaluation uses all 2,250 reference task–initialization configurations with fresh execution seeds. FBA selects from $\mathcal{K} = \{1, 2, 4, 10\}$. We compare it with fixed budgets, ProbeFlow (Fang et al., 2026), and AdaFlow (Hu et al., 2024) under the evaluation protocol in Appendix D.1.4. ProbeFlow is training-free and uses velocity similarity, while AdaFlow adapts step sizes using a variance predictor trained on the same L0 data as FBA with the action policy frozen. FBA and AdaFlow follow $\pi_{0.5}$'s L0 training and in-distribution evaluation (Zhang et al., 2026).

Table 1 reports ordinary-trained FBA with $\lambda = 2$ at L0 and L2 and $\lambda = 1$ at L1; Appendix D.3.1 compares training data and cost weights. FBA improves observed success over both fixed and adaptive baselines across L0–L2. Relative to fixed 1, the strongest fixed-budget baseline, it raises L0 success from 81.6% to 86.6% while increasing mean cumulative NFE from 15.1 to 18.7. The gains over fixed 1 are smaller on L1 and L2, at 3.02 and 2.27 pp, respectively. FBA also uses less cumulative NFE than fixed 10 and ProbeFlow at all three levels. Compared with AdaFlow, its success advantage comes with lower cumulative NFE on L0 and slightly higher NFE on L1/L2.

Intervention timing and budgets. Both intervention timing and budget shares vary across suites. In a separate A100 evaluation on the same L0 configurations with fresh execution streams, intervention rates in Figure 6(a) are high early in Cautious Grasp and rise again around calls 10–13 in Dynamic Obstacles and Hazard Avoidance. In Figure 6(b), NFE 4 accounts for 0.9% of calls in Static Obstacles and 8.3% in Preposition Combinations. Allowing FBA to select budgets after the initial call improves observed success by 2.98 pp at 3.80 additional mean cumulative NFE relative to initial-call-only selection (Appendix D.4.1). At states from these trajectories, paired continuation

Table 1: **Success, computation and latency for $\pi_{0.5}$ on nine VLA-Arena suites.** Results pool $9 \times 250 = 2{,}250$ episodes per method and level. Success is reported as estimate $\pm$ the half-width of a 95% interval; cumulative NFE is averaged over complete episodes. Bold marks the highest observed success per level.

Method	Success (%) ↑ L0 / L1 / L2	Mean ΣNFE ↓ L0 / L1 / L2	Median latency ↓ ms/call
Fixed NFE 10	56.7±1.9 / 32.7±1.7 / 23.3±1.4	201.8 / 245.6 / 262.9	75.3
Fixed NFE 4	63.8±1.8 / 35.6±1.6 / 27.6±1.4	74.4 / 95.6 / 102.0	61.7
Fixed NFE 2	81.3±1.4 / 47.0±1.6 / 33.9±1.3	31.0 / 44.1 / 49.2	57.1
Fixed NFE 1	81.6±1.2 / 48.0±1.4 / 34.6±1.1	15.1 / 21.8 / 24.3	55.3
ProbeFlow	72.4±1.8 / 37.5±2.0 / 26.4±1.4	56.8 / 72.1 / 78.9	60.1
AdaFlow	61.8±1.3 / 31.2±1.6 / 24.1±1.5	21.4 / 32.1 / 34.3	61.5
FBA (ours)	**86.6±1.2 / 51.0±1.5 / 36.9±1.2**	18.7 / 34.8 / 34.9	58.2

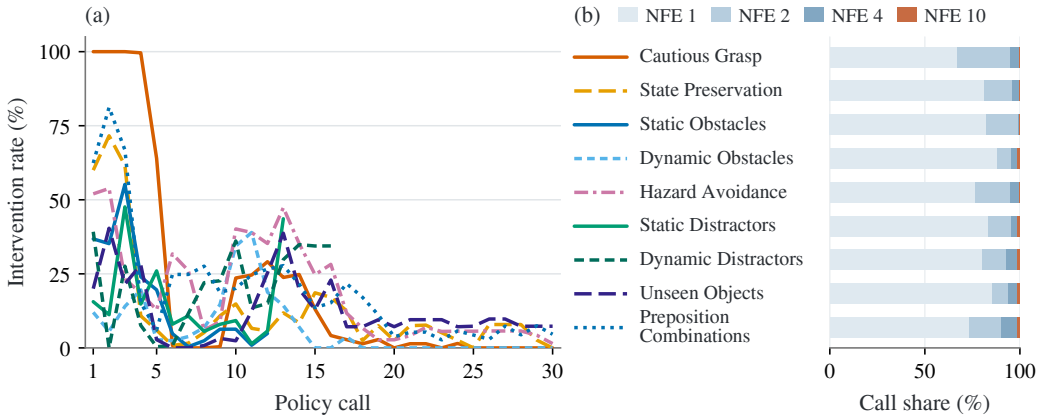


Figure 6: **FBA intervention profiles across suites.** Ordinary three-seed ensemble, $\lambda = 2$, selecting at every call; 250 L0 episodes per suite. (a) Intervention rates (NFE>1) among episodes reaching each call. (b) NFE 1/2/4/10 shares of all calls.

tests in Appendix D.4.2 show that the incremental success benefit of a current budget choice depends on later allocation.

4.4 Real-Robot Evaluation

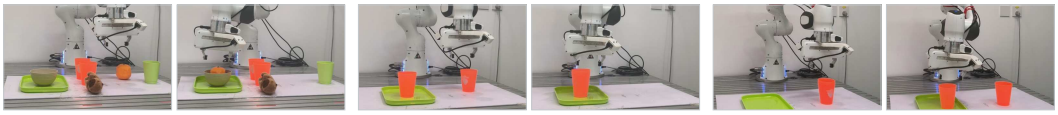
The three tasks in Figure 7 place different demands on allocation. Orange placement (A) mainly tests perception, with few precision-alignment phases. Cup stacking (B) requires precise alignment late in the episode, whereas unstacking (C) requires it during the early and middle portions of the episode. Each method runs 50 trials in A and 30 each in B and C, under the task criteria and evaluation protocol in Appendix D.5.

Budget responses and latency. Table 2 also shows non-monotonic fixed-budget success on the robot. Against each scene's best fixed budget, FBA ties fixed 1 in A (45/50) and exceeds fixed 1 in B (19/30 versus 17/30) and fixed 10 in C (25/30 versus 23/30). FBA mostly selects NFE 1 in A/B. Strong fixed 1 performance leaves less room for gains in perception-oriented A; gains in B/C are consistent with precision-alignment phases providing allocation signals.

Compared with default fixed 10, FBA records more successes in all three scenes with 19.5% lower pooled median call latency (66.9 versus 83.1 ms). Its latency remains above fixed 1's 62.5 ms.

Table 2: **Observed successes and call latency on the robot.** Success is reported as successes/trials. For each method, latency is the median over evaluation calls pooled across scenes.

	Successes/trials ↑			Median latency ↓
Method	A: Orange placement	B: Cup stacking	C: Cup unstacking	ms/call
Fixed NFE 10	43/50	11/30	23/30	83.1
Fixed NFE 4	41/50	16/30	21/30	69.2
Fixed NFE 2	42/50	12/30	19/30	64.8
Fixed NFE 1	45/50	17/30	18/30	62.5
FBA (ours)	45/50	19/30	25/30	66.9



(a) Orange placement (b) Cup stacking (c) Cup unstacking

Figure 7: **Real-robot scenes: (a) orange placement, (b) cup stacking, and (c) cup unstacking.**

FBA has 0.714M parameters per member and 2.142M across its three-member ensemble. Reusing policy features and sampler KV caches avoids an additional encoder forward pass.

5 Related Work

Sampling acceleration. Consistency Policy (Prasad et al., 2024) and SnapFlow (Luan et al., 2026) reduce robot-policy sampling steps through consistency learning or distillation. These approaches modify the action generator for few-step inference. Our budget interventions instead keep the action policy frozen, measuring the consequences of changing its sampling computation.

Adaptive budget allocation. ProbeFlow (Fang et al., 2026) schedules integration using velocity similarity without training; AdaFlow (Hu et al., 2024) adapts step sizes using learned variance. D3P (Yu et al., 2025) jointly trains a step allocator and diffusion policy. ELASTIC (Li et al., 2026) learns sequential and parallel compute allocation around a frozen policy. FBA learns from matched success and cost outcomes, with one ensemble shared across all nine simulation suites. Appendix A reviews these methods and flow-matching robot policies.

6 Discussion and Conclusion

Limitations. Our measurements rely on event-anchor sampling and cannot characterize budget effects at every state along a trajectory. The action probes show correlations with closed-loop performance, while the causal roles of mean-action alignment and sampling variability remain to be established. In our real-robot setup, physical states cannot be restored precisely for matched counterfactual branching, so FBA learns from outcomes under executed budgets. HIL data collection also depends on operator intervention, which limits collection efficiency.

Conclusion. The value of additional sampling computation depends on the policy, the decision state, and subsequent budget choices. Counterfactual state branching makes its local contribution measurable under matched states, histories, and noise. We use the paired success and cost outcomes to train FBA to choose budgets during control, and evaluate these choices in closed-loop simulation and on the real robot. Future work could extend these matched success–cost measurements and their use as supervision to other flow-policy inference choices, including integration time grids and initial sampling noise scales.

AI USE STATEMENT

We used generative AI tools for language polishing, discussion of research ideas, assistance with experimental design, and assistance with mathematical proofs. The authors take responsibility for the final content of this work, including its claims, proofs, and reported results.

REPRODUCIBILITY STATEMENT

Appendix B specifies event and intervention definitions, branch continuation rules, and the state-restoration and noise-matching procedures. Appendix C reports sampling populations, measurement definitions, and uncertainty estimation for the budget-response and numerical analyses. Appendix D documents FBA training-data construction from reference trajectories and paired branches, architecture, objectives, hyperparameters, baselines, and simulation evaluation, including matched call-position ablations and two-continuation branching with execution-repeatability checks. It also details robot hardware, data collection, per-scene adaptation, evaluation counts and success criteria, and end-to-end latency measurement.

REFERENCES

Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Stefan Bauer, and Manuel Wüthrich. Causalworld: A robotic manipulation benchmark for causal structure and transfer learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder. arXiv preprint arXiv:2107.05407, 2021. ICML 2021 AutoML Workshop.

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.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Proceedings of The 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14 (101):3207–3260, 2013.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023.

Lars Buesing, Théophane Weber, Yori Zwols, Sébastien Racanière, Arthur Guez, Jean-Baptiste Lespiau, and Nicolas Heess. Woulda, coulda, shoulda: Counterfactually-guided policy search. In *International Conference on Learning Representations (ICLR)*, 2019.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

Inkook Chun, Seungjae Lee, Michael S. Albergo, Saining Xie, and Eric Vanden-Eijnden. Dasip: Dynamic test-time compute scaling for robot control with stochastic interpolant policies. arXiv preprint arXiv:2511.20906, 2025.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. First return, then explore. *Nature*, 590:580–586, 2021. doi: 10.1038/s41586-020-03157-9.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 2024.

Zhou Fang, Jiaqi Wang, Yi Zhou, and Qiongfeng Shi. Probeflow: Training-free adaptive flow matching for vision-language-action models. arXiv preprint arXiv:2603.17850, 2026.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Zifeng Gao, Guanxing Lu, Tianxing Chen, Wenxun Dai, Ziwei Wang, Chao Shang, Wenbo Ding, and Yansong Tang. Manicm: Real-time 3d diffusion policy via consistency model for robotic manipulation. arXiv preprint arXiv:2406.01586, 2024.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems*, 2025.

Alex Graves. Adaptive computation time for recurrent neural networks. arXiv preprint arXiv:1603.08983, 2016.

Sigmund H. Høeg, Yilun Du, and Olav Egeland. Streaming diffusion policy: Fast policy synthesis with variable noise diffusion models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

Eric J. Horvitz, Gregory F. Cooper, and David E. Heckerman. Reflection and action under scarce resources: Theoretical principles and empirical study. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1121–1127. Morgan Kaufmann, 1989.

Xixi Hu, Bo Liu, Xingchao Liu, and Qiang Liu. AdaFlow: Imitation learning with variance-adaptive flow-based policies. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 138836–138858, 2024. doi: 10.52202/079017-4405.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 652–661. PMLR, 2016.

Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713, 2025.

Andrew Zou Li, Gokul Swamy, Yonatan Bisk, and Andrea Bajcsy. Elastic: Efficiently learning to adaptively scale test-time compute for generative control policies. arXiv preprint arXiv:2606.31132, 2026.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. arXiv preprint arXiv:2411.19650, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

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), Datasets and Benchmarks Track*, 2023a.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023b.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5775–5787, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 2025. doi: 10.1007/s11633-025-1562-4.

Wuyang Luan, Junhui Li, Weiguang Zhao, Wenjian Zhang, Tieru Wu, and Rui Ma. Snapflow: One-step action generation for flow-matching vlas via progressive self-distillation. arXiv preprint arXiv:2604.05656, 2026.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, 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. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.090.

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

Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.071.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1-3):361–395, 1991. doi: 10.1016/0004-3702(91)90015-C.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics. arXiv preprint arXiv:2506.01844, 2025.

Andreas Sochopoulos, Nikolay Malkin, Nikolaos Tsagkas, João Moura, Michael Gienger, and Sethu Vijayakumar. Fast flow-based visuomotor policies via conditional optimal transport couplings. In *Conference on Robot Learning (CoRL)*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *PMLR*, pp. 32211–32252, 2023.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

Zhendong Wang, Max Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. In *International Conference on Machine Learning (ICML)*, 2025.

Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. arXiv preprint arXiv:2502.05855, 2025.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-Cache: Efficient vision-language-action manipulation via adaptive token caching. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 164448–164473, 2025. doi: 10.52202/085713-5484.

Shu-Ang Yu, Feng Gao, Yi Wu, Chao Yu, and Yu Wang. D3p: Dynamic denoising diffusion policy via reinforcement learning. arXiv preprint arXiv:2508.06804, 2025.

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

Yang Yue, Yulin Wang, Bingyi Kang, Yizeng Han, Shenzhi Wang, Shiji Song, Jiashi Feng, and Gao Huang. Deer-vla: Dynamic inference of multimodal large language models for efficient robot execution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. VLA-Arena: An open-source framework for benchmarking vision-language-action models. In *Forty-third International Conference on Machine Learning (ICML)*, 2026.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 49842–49869, 2023.

A Extended Related Work

Budget allocation connects efficient action generation with the decision-dependent value of computation. We review the policy, sampling, and evaluation methods that inform this connection.

Flow-matching-based robot policies. Flow matching learns a time-dependent vector field that transports noise into data (Lipman et al., 2023); rectified flow emphasizes paths that facilitate numerical sampling (Liu et al., 2023b). In robotics, π_0 couples vision-language representations to a flow-matching action expert (Black et al., 2024), and $\pi_{0.5}$ extends this policy family toward open-world generalization (Black et al., 2025a). SmolVLA develops a compact vision-language-action model with a flow-matching action expert (Shukor et al., 2025). Fast-WAM jointly models actions and future video during training while studying action generation without explicit test-time future imagination (Yuan et al., 2026). Conditional optimal-transport couplings further improve few-step flow-based visuomotor sampling (Sochopoulos et al., 2025).

Action representations and generative control. RT-2 and OpenVLA use tokenized actions (Brohan et al., 2023; Kim et al., 2025), while FAST compresses action sequences for autoregressive generation (Pertsch et al., 2025). Continuous denoising underlies Diffusion Policy (Chi et al., 2023), Octo and RDT-1B (Octo Model Team et al., 2024; Liu et al., 2025); CogACT and DexVLA couple diffusion action modules to vision-language representations (Li et al., 2024; Wen et al., 2025).

Few-step sampling and policy distillation. DDIM, DPM-Solver, DPM-Solver++, and UniPC reduce generative sampling cost through alternative discretizations and solvers (Song et al., 2021; Lu et al., 2022; 2025; Zhao et al., 2023). Progressive distillation and consistency models learn fewer-step transformations (Salimans & Ho, 2022; Song et al., 2023); Shortcut Models and MeanFlow train generators for large integration steps or one-step generation (Frans et al., 2025; Geng et al., 2025). Robot-policy methods include Consistency Policy, ManiCM, and One-Step Diffusion Policy (Prasad et al., 2024; Gao et al., 2024; Wang et al., 2025), with SnapFlow targeting flow-matching VLAs (Luan et al., 2026). These methods accelerate generation; our interventions measure the success–cost response to sampling budgets with the generator fixed.

Deployment efficiency beyond sampling steps. DeeR-VLA adapts model execution depth (Yue et al., 2024), and VLA-Cache reuses visual-token computation (Xu et al., 2025). Streaming Diffusion Policy and real-time action chunking address the temporal interaction between inference and execution (Høeg et al., 2025; Black et al., 2025b). These methods change different sources of delay, motivating separate reporting of network evaluations and measured deployment latency.

Adaptive computation and budget allocation. The value of computation and metareasoning frame computation as a resource whose usefulness depends on the current decision (Horvitz et al., 1989; Russell & Wefald, 1991). Adaptive Computation Time and PonderNet learn how much internal computation to perform (Graves, 2016; Banino et al., 2021). For robot control, AdaFlow adapts integration using learned variance (Hu et al., 2024); DASIP selects budgets and solver configurations from observation-based difficulty (Chun et al., 2025); ProbeFlow uses velocity similarity without training (Fang et al., 2026). D3P jointly learns a step allocator and diffusion policy (Yu et al., 2025). ELASTIC learns sequential and parallel compute allocation around a frozen policy (Li et al., 2026). FBA learns budget selection from matched terminal success and remaining cumulative NFE under a common continuation rule.

Counterfactual evaluation and state restoration. Offline counterfactual evaluation relies on logging and modeling assumptions (Bottou et al., 2013; Jiang & Li, 2016); model-based policy search evaluates alternative outcomes of recorded trajectories (Buesing et al., 2019). Go-Explore uses restoration to revisit states (Ecoffet et al., 2021), while CausalWorld supports controlled task interventions (Ahmed et al., 2021). We use state restoration to compare sampling budgets, matching policy history and sampler noise across branches.

B COUNTERFACTUAL STATE BRANCHING

To measure the effect of a budget choice, we need to compare actions from the same decision state and specify what follows them. Physical events locate comparable moments in a task; state restoration and matched sampler noise provide the paired executions. This chapter defines these comparisons and examines how faithfully they can be reproduced.

B.1 LOCATING AND COMPARING BUDGET INTERVENTIONS

A budget intervention has three defining choices: where it starts, how long it lasts, and which budget follows it. We specify these choices for each reference population, so that stage-wide changes, single-call changes and complete allocation policies answer distinct questions.

B.1.1 EVENT DETECTION

Physical events align interventions with task progress: approaching an object, establishing contact, and reaching the goal. For each task and subgoal occurrence, we detect these events from the unmodified reference trajectory. Geometric criteria are calibrated using its physical measurements and fixed before budget outcomes are observed.

Free-space onset marks the start of three consecutive physical steps with sufficient end-effector clearance, using hysteresis, without object, background or obstacle contact or recovery. *Approach-region entry* is an outside-to-inside crossing of the task's approach region before the first valid primary contact, confirmed in one step. *Contact establishment* marks the start of three consecutive steps of gripper contact with the target object, excluding background and obstacle contact. *Goal attainment* requires official success and, where enabled, the task's object–goal proximity condition, confirmed in one step. Gripper release is not an additional gate on this primary goal event.

For an event starting at τ_i with dwell d_i, confirmation occurs at $\widehat{\tau}_i = \tau_i + d_i - 1$ when the required predicates remain valid throughout the run. Events are accepted in canonical order; missing or out-of-order events cannot supply a stage endpoint. Recorded call boundaries map physical steps to the policy calls that executed them, including calls shortened by termination. A window is ineligible if its endpoints fall in the same call or its required successor call is absent.

A supplementary release window runs from the release-containing call through the first official-success call. Release requires prior gripper contact, subsequent contact loss, and measured gripper reopening relative to the running minimum, sustained for two steps; a one-step terminal run is flagged as truncated. This diagnostic selects successful references with confirmed release and is kept separate from the primary population.

B.1.2 EVENT ANCHORS, STAGES, TRANSITIONS, AND ROOTS

An *event anchor* is the first qualifying onset τ_i of a registered physical event on the reference trajectory. Confirmation at $\widehat{\tau}_i$ validates this onset without shifting it. Let c_i be the call containing τ_i and $b_i = c_i + 1$ the next call boundary. An *execution stage* spans consecutive boundaries: $S_i = [b_i, b_{i+1})$. The primary stages are S_0, *free-space approach*, from free-space onset to approach-region entry; S_1, *contact acquisition*, from approach-region entry to contact establishment; and S_2, *post-contact manipulation*, from contact establishment to goal attainment, using each event's following call boundary. A *root* is the saved decision state at the start of a selected call. Event anchors locate physical moments, stages group calls, and roots supply states for budget comparisons.

The *transition windows* T_1 and T_2 nominally use $[c_i, c_i + 2)$ at approach-region entry and contact establishment: the anchor-containing call and its successor, with the same treatment budget in both. Such windows straddle the corresponding stage boundary and can overlap stage windows or each other. The *initial transition* $T_0 = [0, 2)$ covers the first two calls. A goal-attainment window T_3 is defined only when a successor call exists; it has no eligible observations in the L0 primary map and is omitted from Figure 8. Same-call events can produce zero-length stage windows and are recorded as ineligible; execution stops at native termination.

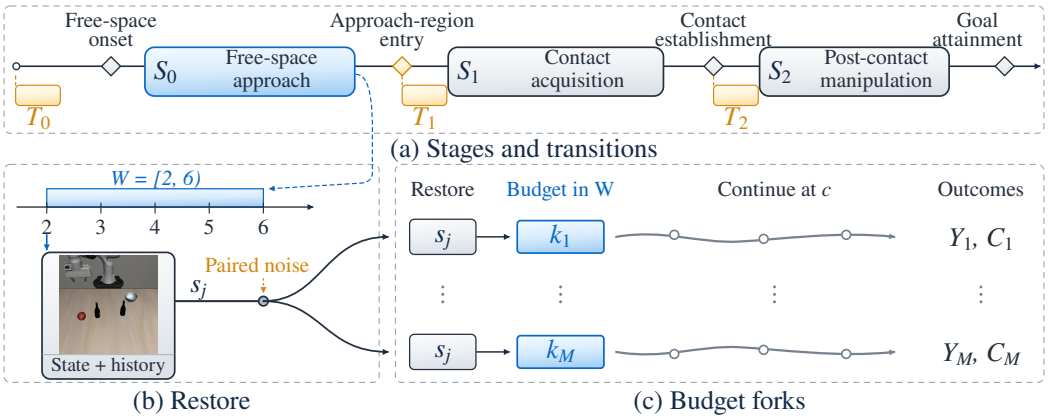


Figure 8: **Counterfactual state branching over $\mathcal{K} = \{k_1, \ldots, k_M\}$.** (a) Stages S_i and transitions T_i align comparisons with physical progress. (b) Each branch restores the same saved state and policy history. (c) It uses budget k_i within W, then continues with common budget c; sampler noise is matched by call index. Y_i denotes terminal success and C_i the remaining cumulative NFE. The example uses $\pi_{0.5}$ on Static Obstacles L2 with $W = [2, 6)$ in zero-based call indices.

Figure 8(a) illustrates nominal windows. Each comparison uses its recorded one- or two-call offsets; the schematic shows the ordering of windows. Nested roots branch at an event-containing call and again at its successor. Each window family uses its own eligibility rules and denominator.

B.1.3 Matched Branches and Continuation Rules

The experimental unit is a reference root and its matched candidate branches. For a reference policy budget b, the reference supplies the root state and an intervention window. A candidate $k \in \mathcal{K}$ is used within that window, after which the branch follows a fixed continuation budget c until success, native termination or the official time limit. Window endpoints remain fixed at the reference call indices, even if a branch reaches a physical milestone earlier or later. Native termination ends execution before a nominal window endpoint when applicable. All candidates share the root state, prompt, history before the root and noise convention. We evaluate changes in b, c or intervention duration as separate conditions, since each changes the question the paired outcomes answer.

The bidirectional maps use each evaluated policy's own low- and high-budget references. The low-budget background tests upgrades with a low-budget continuation; the high-budget background tests downgrades with a high-budget continuation. The two directions therefore measure responses in the states visited by their respective reference policies. The experimental candidate set is $\{1, 2, 4, 10\}$, while the branching interface accepts a configured finite set $\mathcal{K}$.

The map study covers nine suites, five tasks per suite and three difficulty levels, with initializations and eligible windows recorded for each comparison. Its reference trajectories are sampled separately from the 2,250-episode-per-level FBA evaluation. Paired effects use valid outcomes for both budgets, and incomplete groups are recorded. A failed execution is retried under the same condition, keeping its initialization regardless of prior outcomes. Population summaries and uncertainty account for shared references: several roots and candidate branches can come from one trajectory.

B.2 Restoring Decisions and Matching Executions

Restoring the physical state alone is insufficient: task predicates, the observation and policy history also determine the next decision. At call c_0, we therefore save the simulator and controller states, task status, environment random-number-generator state, observation and causal policy history. Each candidate starts from an independent restoration of this snapshot. Before execution, we check the restored state, observation, official success predicate, reference and history, and current call index. A failed check excludes the branch from comparison without labeling it a task failure.

Table 3: Fork-runtime restoration and replay checks. The Cautious Grasp replay covers both complete branches. The rejected root shows why checks include task predicates and numeric state.

Check	Evaluated support	Recorded result
Root qualification	VLA-Arena event windows; each of $\pi_{0.5}$ and FastWAM	$19{,}912/19{,}912$ leaves qualified per model; $4{,}978/4{,}978$ complete four-arm groups.
Complete branch replay	Selected $\pi_{0.5}$ Cautious Grasp case; NFE 1/10 arms, 18/24 calls	All 42 calls match noise/action digests, NFE, physical starts, executed steps and terminal flags.
Rejected restore	VLA-Arena release windows; one shared root in both models	State/image $\delta = 0$ but success predicate differs; 4 leaves/model remain missing out of 1,164 planned.

For candidates k, k' in one group, we match sampler noise according to

$$\xi_j^{(k)} = \xi_j^{(k')} = g_\xi(r, q, j), \tag{9}$$

where r, q, and j identify the reference, repeat, and logical call, respectively. The deterministic noise generator g_ξ is independent of which worker executes the branch. At each call, we log noise/action digests, requested and actual NFE, executed steps, and termination flags. We rerun the control arm from the restored state. Restored inputs and call-indexed sampler noise match; subsequent states follow each branch's budgets.

B.2.1 How Faithfully Are Branches Reproduced?

We check decision-state restoration and replay selected complete branches. These checks test each comparison's starting point and subsequent execution. Table 3 reports coverage and results. For compared numeric arrays, $\delta(x, \widetilde{x}) = \max_j |x_j - \widetilde{x}_j|$; image differences use RGB values scaled to $[0, 1]$.

The accepted-branch counts include successful retries. The rejected release root shows why the checks extend beyond numeric state and images: both match, yet the restored success predicate differs. We exclude every candidate at that root to preserve the paired comparison.

B.2.2 Executing Many Comparisons with a Shared Policy

Matched branching repeatedly evaluates the same frozen policy from different saved states. Resident GPU servers therefore share model weights across independent simulator workers. Workers divide a fixed list of comparisons and save each group's outcomes and execution failures; interrupted collection resumes with the same conditions. Each request specifies sampler noise and NFE, and JAX inference checks the number of evaluations actually performed.

For $\pi_0/\pi_{0.5}$, prefill and integration are compiled separately. Denoising steps reuse the vision-language features of the current observation, while a distinct fork observation is encoded independently. Device work is synchronized before host timing. Parallel workers and queueing affect collection throughput; deployment latency is measured separately at the policy call.

C MEASUREMENTS OF BUDGET RESPONSES

The fixed-budget reversals raise two further questions: how does the response depend on where and for how long the budget changes, and what does numerical refinement change in the generated actions? We first examine success and cost under specified interventions, then follow the states reached in closed-loop execution. Expert-action probes separate numerical convergence, mean-action alignment and sampling variance.

C.1 WHERE AND FOR HOW LONG DOES BUDGET MATTER?

A response map locates budget-sensitive stages, while a single-call comparison measures the effect of one decision. Extending the intervention to two calls measures how its effect changes with duration. Each comparison uses its own reference population, pairing success and cumulative NFE from the same configuration.

C.1.1 BUDGET RESPONSES ACROSS STAGE WINDOWS

State transitions and window values. A budget-k call induces a state-transition kernel $P_k(ds' \mid s)$ through action generation, execution and history updates, with expectation operator $(\mathcal{T}_k f)(s) = \int f(s')P_k(ds' \mid s)$. Let $V_c(s)$ be terminal-success probability when all calls from s use budget c. Terminal states are absorbing, with V_c equal to the success indicator and no further cost, including when a branch ends before the window does. For a reference-fixed window $W = [u, v)$, composing $L = v - u$ transitions gives $Q_{c,W}(s, k) = (\mathcal{T}_k^L V_c)(s)$, the operator form of Equation (3), with $Q_{c,W}(s, c) = V_c(s)$. For $W = [t, t + 1)$, this reduces to $Q_{c,W}(s, k) = (\mathcal{T}_k V_c)(s)$: one budget-$k$ call followed by budget c.

Population estimates. The population value is $\bar{Q}_c(k) = \mathbb{E}_{(s,W)\sim d}[Q_{c,W}(s, k)]$, using the window values in Equation (3). Linearity gives $\Delta\bar{Q}_c(k, k_0) = \bar{Q}_c(k) - \bar{Q}_c(k_0)$, as measured in Equation (4); costs are defined analogously. For each sampling distribution d, we average roots within reference, references within suite, and then suites equally. Reference-cluster bootstrap intervals are stratified by suite and task; the displayed pointwise intervals are descriptive. Reference policy, root sampling and continuation budget define the population being compared. With one binary outcome per state–window pair and budget, these measurements estimate population effects rather than individual-state success probabilities.

Suite-level response maps. Budget responses vary across stages, suites and policies. Figure 9 extends the L0 map in Figure 3 to all nine suites with the same budget contrasts, nominal window order and color scale. Panels show paired success differences on each policy's reference population.

Map sampling. Each map cell starts from a pool of 50 reference trajectories (five tasks with ten initializations each) for the corresponding suite, policy, and background. We retain references with an eligible event window and valid outcomes for both budgets in the comparison.

Reducing the budget during a stage. Figure 4 uses the L0 budget-downgrade stage population, with fixed-10 references and NFE 10 after the registered window W. In this figure, the subscript W on $\Delta\bar{Q}_{10,W}$ identifies the stage-window population used for $\Delta\bar{Q}_{10}$; it does not imply identical window endpoints across references. Eligible windows are averaged within each reference. Reference identifiers corresponding to the same suite, task and initialization are then merged before averaging equally across references within each suite and across suites. All four budgets use the same set of windows with complete outcomes. The numbers of groups/references are 693/352 for $\pi_{0.5}$, 829/390 for FastWAM, 615/327 for SmolVLA, and 797/406 for π_0, each covering nine suites and 45 task strata. The paired reference-cluster bootstrap is stratified by suite-task (20,000 draws); intervals are pointwise and descriptive, without hypothesis tests. The control contrast is zero by construction.

Reducing the budget throughout a stage can improve success while saving computation. For $\pi_{0.5}$, $k = 1$ gives $+7.76$ pp $[4.28, 11.20]$ versus 10 and saves 56.85 remaining NFE. The $k = 2$ and $k = 4$ success differences are $+6.31$ pp $[3.34, 9.27]$ and $+1.35$ pp $[-1.18, 3.87]$. These intervals summarize stage-window effects on fixed-10 references, rather than individual map cells.

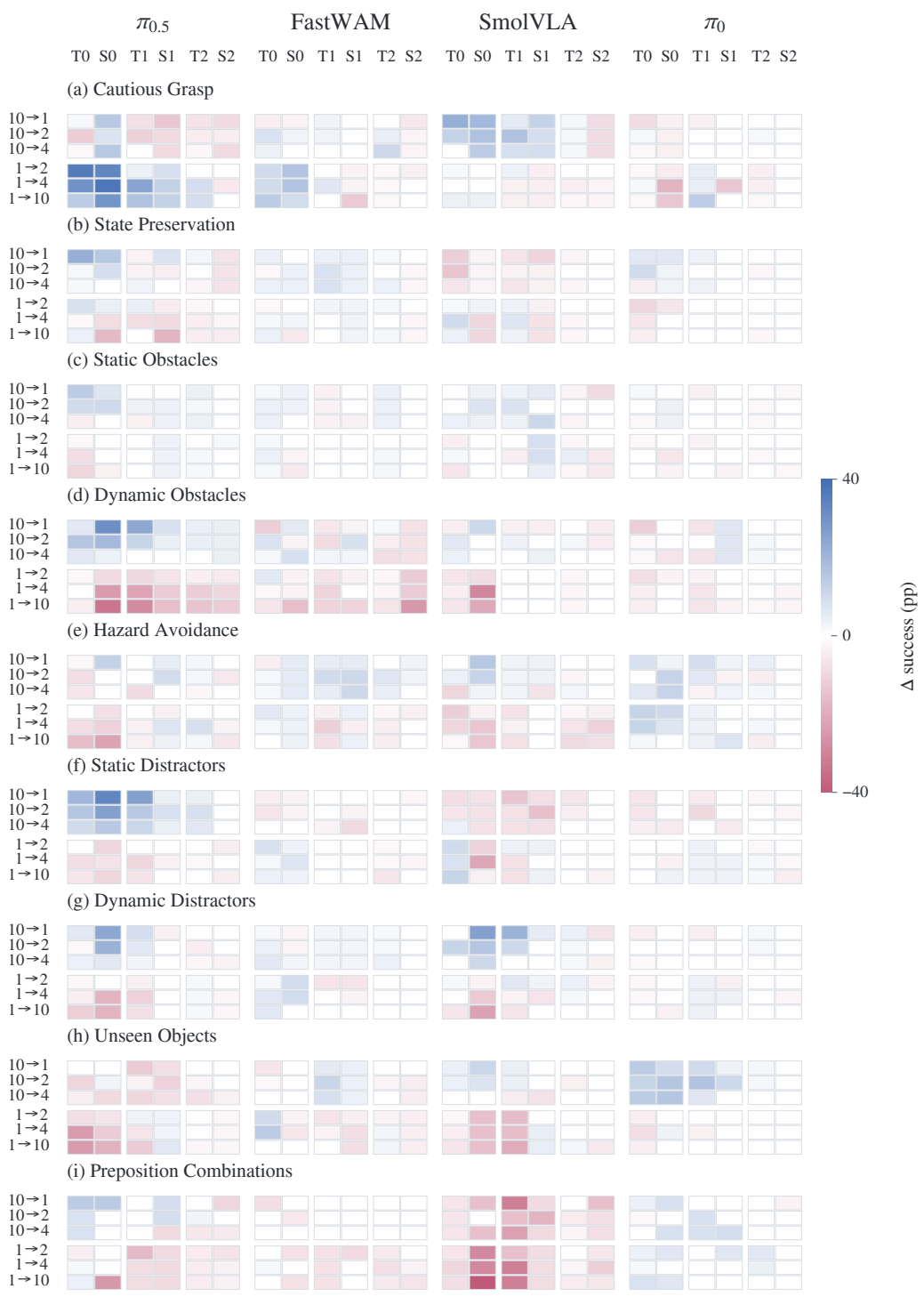


Figure 9: **Budget-response maps across all nine VLA-Arena suites (L0).** Four model panels per suite follow Figure 3. Rows $10 \rightarrow k$ and $1 \rightarrow k$ use fixed-10 and fixed-1 references and continuation, respectively. Columns follow T0, S0, T1, S1, T2, S2; windows can overlap and have different eligible populations. Blue/rose indicate positive/negative paired success differences; white denotes zero. The scale saturates at ± 40 percentage points. No cell-wise significance is implied.

Table 4: **States and references in the L0 single-call comparison.** Each event-sampled root has four matched budget outcomes.

Policy	Event-sampled roots	Reference trajectories
$\pi_{0.5}$	2,250	2,250
FastWAM	1,125	1,125
SmolVLA	1,125	1,125
π_0	450	450

C.1.2 Single-Call Value at Event-Sampled Roots

To measure the effect of spending more at a single decision, we compare candidate budgets at event-sampled roots from fixed-1 references. The current call uses $k \in \{1, 2, 4, 10\}$, with NFE 1 thereafter. Each sampled state has one matched outcome per budget in each source population; root and reference counts appear in Table 4.

For event roots stored in two-call trees, we take the outcome of the branch that returns to NFE 1 at the second call. The stored parent rollout does not constitute an additional noise repeat. Records of the same saved state from different sampling groups are retained separately; their sampler noise and continuation executions are not assumed to be independent or identical. The 20,000-draw bootstrap resamples reference clusters within suite-task strata, keeping all budgets and roots from a reference together.

Single calls on an NFE-1 background. Starting from fixed 1 changes the question: can spending more at one call improve the remaining episode? Figure 10(b) reports paired success changes for $\pi_{0.5}$ at event-sampled roots of $+0.67$pp $[-0.44, 1.78]$ at $k = 2$ and -1.82pp $[-3.29, -0.36]$ at $k = 10$. These upgrades add 0.93 and 9.36 cumulative NFE to the remaining rollout, respectively. Figure 10(a) gives the corresponding absolute success rates and remaining costs for all four policies.

Initial-call effects. At the initial call, a separate L0 comparison for $\pi_{0.5}$ gives $+4.18$pp $[2.80, 5.56]$ for $k = 2$ versus 1, followed by NFE 1.

C.1.3 Extending a Budget Choice to a Second Call

We next compare applying budget k for one call with applying it for two consecutive calls, using NFE 1 afterward in both cases. Figure 10(c) shows that extending the $\pi_{0.5}$ intervention to a second call changes success by -2.27pp $[-3.33, -1.20]$ at $k = 4$ and -2.58pp $[-3.73, -1.42]$ at $k = 10$. Longer intervention need not improve the outcome; budget and duration enter the decision together.

The duration comparisons reuse the 4,950 event-root trees. In seven trees, 28 first-call branches terminate; their nominal second-budget alternatives retain the absorbing outcome with no additional call or NFE. These reused outcomes are not additional independent observations.

Outcome disagreement at mixed diagnostic nodes. In the four policies' L0 mixed-node diagnostic populations, 4.1–9.2% of nodes have different binary outcomes at NFE 1 and 10 when the current call changes and subsequent calls use NFE 1. These populations include sampled roots and executed successor states. The statistic counts disagreement for this budget pair, with each policy evaluated on its own diagnostic population.

C.1.4 Reading the Fixed-Budget Curves and Branch Examples

The curves summarize whole-episode budget choices; the illustrated branches show how changing the budget from a shared state can reverse an outcome.

Fixed-budget curves. VLA-Arena defines L0 as in-distribution skills, L1 as near-distribution variations, and L2 as larger distribution shifts relative to L0 fine-tuning (Zhang et al., 2026). These

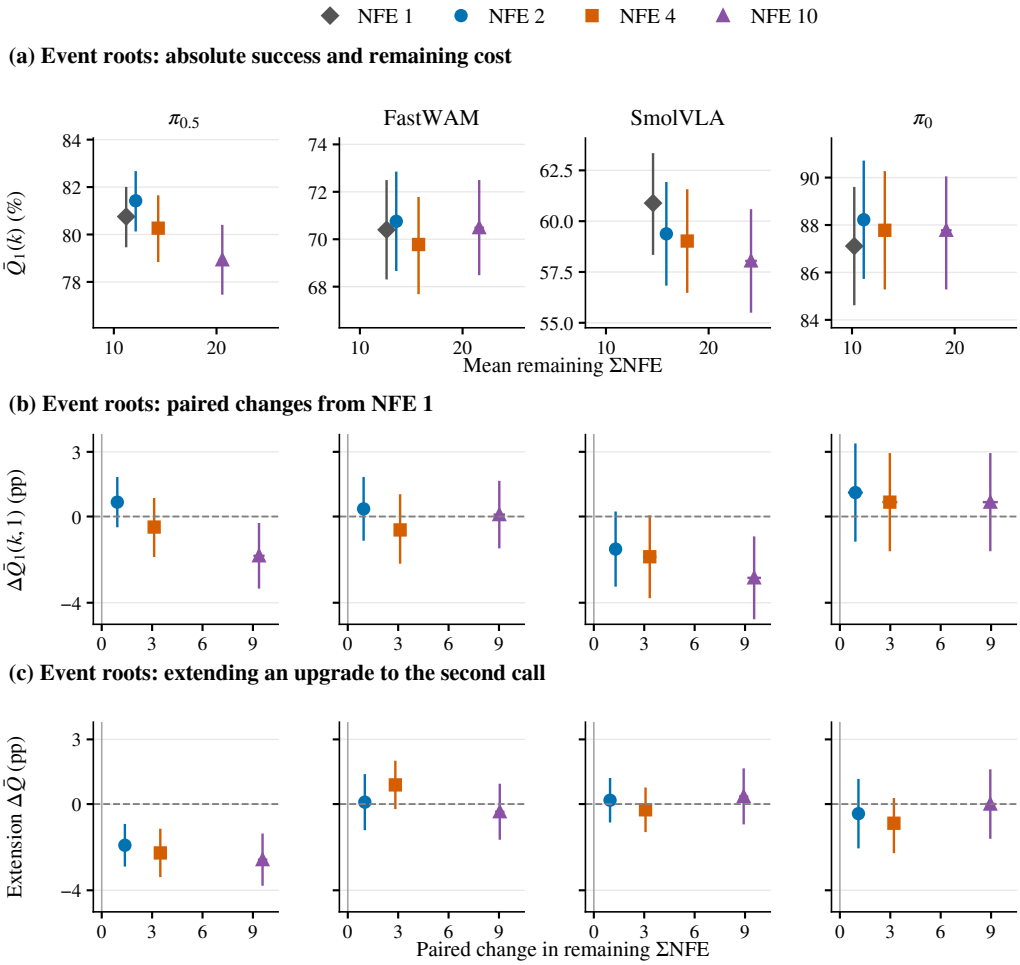


Figure 10: **Success and cost of single-call choices and their extension.** Columns show four policies at L0 event-sampled roots on fixed-1 references, with NFE 1 continuation. (a) One current call at the indicated budget: absolute success and remaining NFE, with marginal 95% reference-cluster intervals and model-specific success scales. (b) Paired changes in success and remaining NFE relative to NFE 1, using the outcomes in (a). (c) The added effect of keeping budget k for a second call instead of returning to NFE 1: (k, k) minus $(k, 1)$, with NFE 1 thereafter. First-call terminal branches remain absorbing with no second-call cost. Panels in (b,c) use common horizontal ranges and common vertical ranges across policies. Bars give paired 95% reference-cluster intervals, stratified by suite-task with equal suite weights.

changes are suite-specific: Static Obstacles has zero, one and two obstacles across L0/L1/L2; Unseen Objects progresses from specified training objects to new instances and then new related categories. The Arena curves in Figure 1(a) compare all ten integer budgets on the same suite–task–initialization configurations within each policy–level pair, pooled over nine suites. Each budget has 450 episodes, except $\pi_{0.5}$ with 360/360/361 at L0/L1/L2. The sampled configurations can differ across policies and levels. Lines connect observed success rates; stars mark the highest observed rate within each curve. Uncertainty intervals are not shown. This dense sweep uses a separate sample from the 2,250-episode FBA grid.

Cautious Grasp. A one-call budget reduction reverses failure to success in the selected case in Figure 1(b). It is a $\pi_{0.5}$ L0 development case: *pick up the scissors and place it on the cutting board* (task 03, initialization 38). The registered one-call window is $[6, 7)$, on a fixed-10 refer-

ence with NFE 10 continuation. Outcomes at budgets 1/2/4/10 are 1/0/0/0, with remaining NFE 171/232/234/240. The NFE-1 branch succeeds at physical step 249; the NFE-10 branch times out at step 310. To export the terminal camera observations, we replayed these two branches from the stored state and history. All 18 and 24 calls, respectively, match the original noise and action digests, budgets, executed-step counts and terminal flags. The two endpoint images show each branch's own termination, not a shared physical time.

Static Obstacles. The example in Figure 8 changes the budget across a stage. Its image and window come from a separate $\pi_{0.5}$ L2 case, *pick the onion and place it on the bowl 2* (task 03, initialization 19). Its S_0 window is $[2, 6)$, again on a fixed-10 reference with NFE 10 continuation. Outcomes are 1/0/0/0 at budgets 1/2/4/10, with remaining NFE 84/248/256/280 at episode termination. The image is the stored pre-intervention observation. The figure illustrates the general interface with symbolic candidates and outcomes, instantiated here by four budgets. Call indices start at zero.

Both cases record all four arms and pass restore checks. Images retain the full 256×256 field of view, flipped vertically for display. Both were selected post hoc from recorded outcome reversals.

Local measurements and terminal outcomes. A separate development analysis examines five selected pairs of early or middle and later interventions across the four policies. In all four middle–later pairs, the later root is closer to the goal, yet its single-call upgrades fail. Contact and release predicates also differ across these roots. Full first-call action traces are available at two supplementary roots. In the π_0 Hazard Avoidance case, NFE 2 ends the first call slightly farther from the goal than NFE 1. With NFE 1 thereafter, the NFE-2 branch succeeds and the NFE-1 branch fails. First-call root mean squared action error uses the NFE-10 block as its reference, while control-sequence smoothness also describes the locally executed block. The executed continuation supplies the terminal outcome that these local measurements alone leave unresolved.

C.1.5 Closed-Loop State Evolution under Fixed Budgets

Terminal success summarizes the episode but leaves open how the executions separate. We therefore track task geometry, contact and object lift under four fixed budgets. The comparison follows each budget's evolving states, complementing the interventions that start from a matched decision state.

Paired executions and demonstrations. This development study records $\pi_{0.5}$ on Unseen Objects L0: five tasks (cake, donut, kiwi, onion and tomato), ten native initializations per task and eight sampler-noise repeats, each at fixed NFE $\{1, 2, 4, 10\}$. The resulting 1,600 episodes share 50 initial simulator states and form 400 four-budget groups. Noise is matched across budgets; subsequent states follow each budget's own closed-loop execution. Control runs at 20 Hz with ten executed actions per policy call, shortened when native termination occurs. A passive-recording check matches action sequences and terminal states with recording on and off. Success counts at NFE 1/2/4/10 are 344/288/193/187 out of 400, respectively. These runs are separate from Figure 1(a)'s dense sweep.

The reference bank contains 64 distinct physical demonstration trajectories from the official dataset: 15/12/12/11/14 for the five tasks, respectively. Native evaluation initializations are not paired to demonstration starts. Stored demonstration states are restored using native scene assets checked against the source imagery; the original scene XML is unavailable. Demonstrations end at their last stored pre-action state, so their final successful post-action state need not be represented.

State and interaction measurements. Let p_E, p_O, p_G be end-effector, object and target-box positions in meters, and R_G the box-frame orientation. For task u's recorded demonstration states $\mathcal{D}(u)$, define

$$\begin{aligned} g_{\mathrm{pos}}(x) &= \left[R_G^\top(p_E - p_O);\ R_G^\top(p_O - p_G)\right], \\ D_{\mathrm{pos}}(x) &= \frac{100}{\sqrt{2}} \min_{e \in \mathcal{D}(u)} \|g_{\mathrm{pos}}(x) - g_{\mathrm{pos}}(e)\|_2 \quad \text{(cm)}. \end{aligned} \tag{10}$$

The semicolon denotes vector concatenation. The two relative-position components use the same nearest recorded state. This distance compares task geometry without requiring the rollout and demonstration to reach it at the same time. Object–target horizontal distance is $100\|(p_O - p_G)_{xy}\|_2$ cm, using object and target-box body origins to describe transport toward the target in the world horizontal plane. Contact is the fraction of executed steps in a call with gripper–target

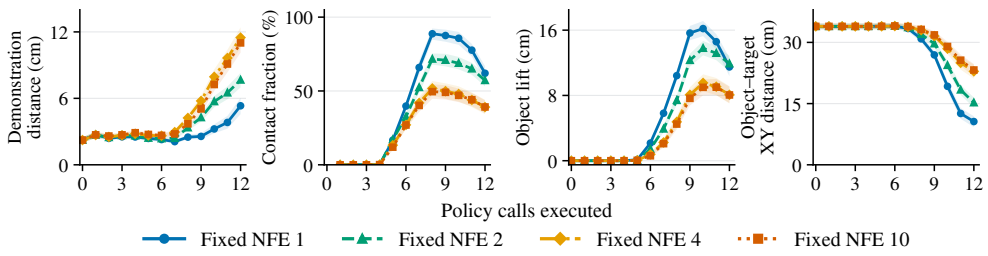


Figure 11: **Physical progress under fixed budgets on Unseen Objects L0.** For $\pi_{0.5}$, all 400 groups (five tasks $\times$ ten initializations $\times$ eight matched noise repeats) are observed at every budget through call 9. Later means use each budget's available states, without extending trajectories after termination. Contact is the fraction of executed steps within a call with gripper–object contact. Bands show pointwise 95% initialization-cluster bootstrap intervals for task-balanced means.

object contact, not a grasp-success probability. Object lift is vertical displacement from the initial object height. Initial call 0 has no contact execution window.

Comparing states as episodes end. Figure 11 shows calls 0–12. All 400 four-budget groups are observed through call 9, which may include a just-successful terminal state. Later calls have partial observation coverage. At call 12, NFE 1/2/4/10 retain 345/376/381/387 observed trajectories; 327 groups have all four budgets observed. Each curve uses its own budget's observed states, without restricting earlier calls to trajectories that remain observed later. States are not padded after termination, so post-call-9 means describe changing, budget-specific observation sets.

Means equally weight the five within-task observed-trajectory means. Pointwise 95% intervals use 10,000 bootstrap draws of initialization clusters within each task, retaining all available noise repeats and matched budgets. Within each resampled task, the sum of observed values is divided by the observed count; the five resulting task means are then averaged. Resampling weights are shared across calls, budgets and metrics. The display is descriptive, without selecting calls by significance. For NFE 1 versus 10, the jointly observed pair count is 339 at call 12, 157 at call 14 and 35 at call 20.

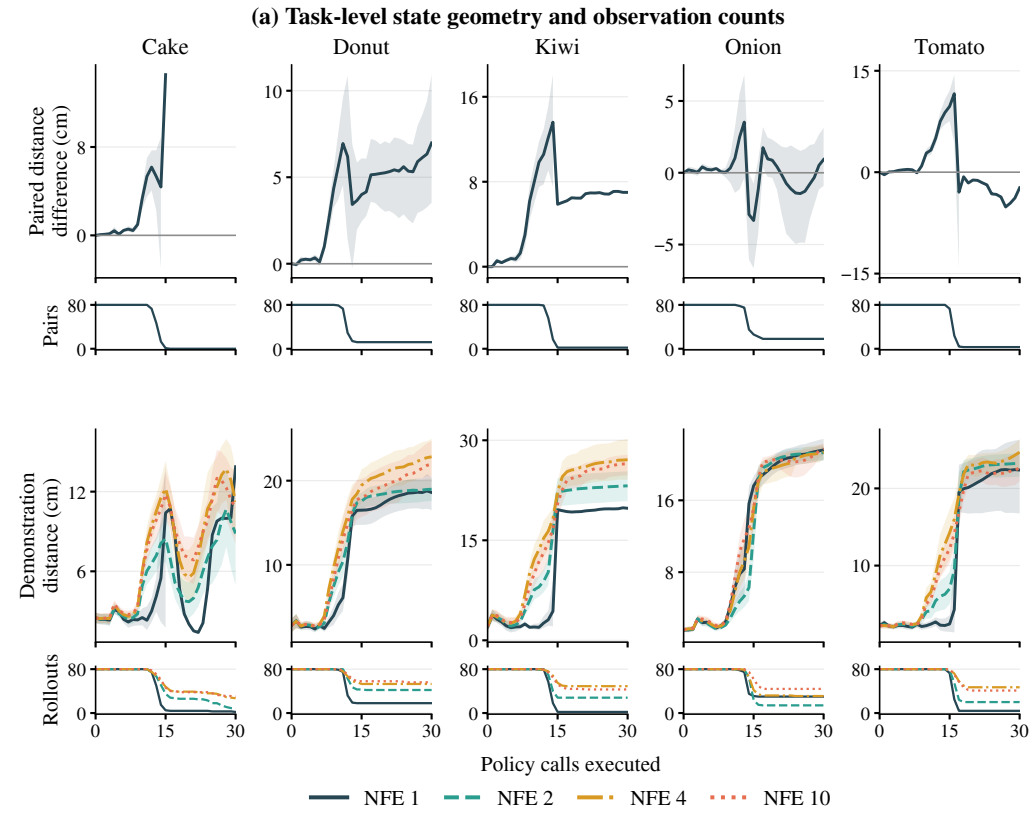
Early differences in geometry, contact and lift. At call 9, all four budgets are still observed in every matched group. NFE 1 is closer to recorded demonstration geometry on average, has a higher fraction of gripper–object contact, and lifts objects farther than the larger budgets. Together, these observations indicate better early physical progress under NFE 1 in this task set. The distance separation is most pronounced on donut and kiwi and small on onion. The response is nonmonotonic in budget: NFE 4 is farther from demonstration geometry than NFE 10 at this boundary.

Figure 12(a) separates these task-level patterns: the upper panels compare jointly observed pairs, while the lower panels retain all four budgets. The count panels show how later termination changes the states contributing to each comparison.

A matched example of contact and lift. Figure 12(b) shows one tomato pair from the same initial state and matched noise. Among all 400 pairs, it minimizes the mean standardized squared distance to the pooled call-9 means of six measurements: both budgets' demonstration distance, contact fraction and lift. Selection does not use terminal outcomes. NFE 1 subsequently succeeds and NFE 10 times out. The example illustrates contact and lift; its call-9 demonstration distances, 2.865 versus 3.088 cm, are closer than the pooled means. Frames are original recorded camera observations with only a vertical flip for display.

C.2 Numerical Probes

How do sampling budgets change mean actions and their variability? We hold each expert-trajectory observation fixed and vary budget and sampler noise, separating numerical refinement, mean-action agreement with demonstrations and sampling variance. The four-model probe uses 600 expert episodes on six suites, taking early, middle and late observations for 1,800 frames per checkpoint. All budgets $\{1, 2, 4, 10, 32\}$ share $J = 8$ sampler seeds. Comparisons use the same ten-step, seven-



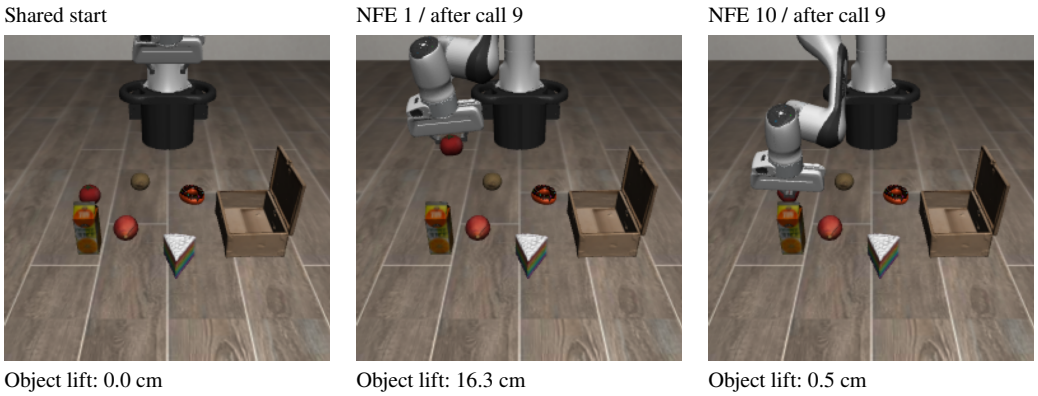


Figure 12: **Closed-loop state geometry and a matched execution example.** Results use $\pi_{0.5}$ on Unseen Objects L0. (a) Upper panels: NFE 10 minus NFE 1 demonstration distance and jointly observed pair counts. Lower panels: absolute distances for all four budgets and their separate rollout counts. Bands show pointwise 95% intervals from 2,000 within-task initialization-cluster bootstrap draws, retaining noise repeats and matched budgets. Intervals are omitted with fewer than three observed initializations or fewer than 95% valid draws. States are not extended after termination. (b) One shared tomato initialization and both budgets after call 9: object lift is 16.3 cm under NFE 1 and 0.5 cm under NFE 10. The example was selected for its proximity to pooled call-9 distance, contact and lift measurements, without using terminal outcomes.

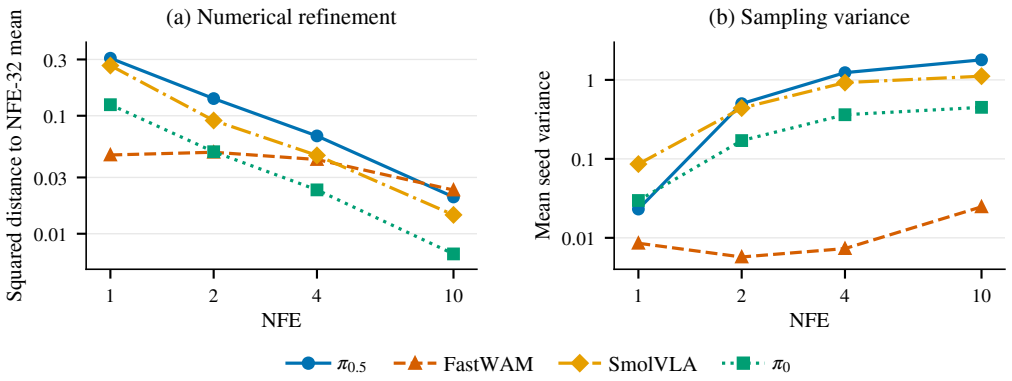


Figure 13: **Numerical refinement and sampling variance across budgets.** Each checkpoint is evaluated on 1,800 expert frames with eight shared sampler seeds; the six VLA-Arena suites receive equal weight. (a) Squared distance of the eight-seed mean action to its NFE-32 mean; the reference zero is omitted. (b) Sampling variance of the 10×7 action block, computed across seeds at each expert state and averaged within episodes, tasks and suites. Both panels use logarithmic vertical scales with separate ranges.

coordinate action prefix after each model's documented output transformation. Phases are averaged within episodes, episodes within tasks, and tasks within equally weighted suites. Bars on B_{32} and G_{align} denote these hierarchical averages of the frame-level estimates; variance summaries use the same weighting. The NFE-32 sample mean supplies the finite numerical reference. These probes hold expert-trajectory states fixed; closed-loop forks measure terminal outcomes under budget interventions. Cross-checkpoint comparisons also mix training recipes and sampled states.

For each frame, let $\widehat{\mu}_k(s) = J^{-1} \sum_{j=1}^{J} a_k(s, \xi_j)$. The empirical sampling variance is

$$\widehat{\mathcal{V}}_k(s) = \frac{1}{J} \sum_{j=1}^{J} \|a_k(s, \xi_j) - \widehat{\mu}_k(s)\|^2. \tag{11}$$

C.2.1 Numerical Refinement and Finite-Reference Bias

Even if integration converges, its limiting mean can differ from the demonstrated action. For the limiting mean $\mu_\infty(s) = \lim_{k \to \infty} \mu_k(s)$, when it exists, let $b_k(s) = \mu_k(s) - \mu_\infty(s)$ and $B_\infty(s) = \|\mu_\infty(s) - e(s)\|^2$. Mean-action expert error decomposes as

$$\|\mu_k - e\|^2 = B_\infty + \|b_k\|^2 + 2\langle \mu_\infty - e, b_k \rangle. \tag{12}$$

A negative cross-term can outweigh $\|b_k\|^2$, placing the finite-budget mean closer to e than the limiting mean. We estimate B_∞ with the finite-reference proxy

$$\widehat{B}_{32}(s) = \|\widehat{\mu}_{32}(s) - e(s)\|^2. \tag{13}$$

Distances sum over the 10×7 executed-prefix block. The finite reference retains integration and Monte Carlo error. Figure 13(a) plots $\|\widehat{\mu}_k - \widehat{\mu}_{32}\|^2$; the zero at 32 is omitted. At NFE 10, mean actions are closer on average to the finite NFE-32 reference than at NFE 1 for all four checkpoints. Their mean full-block sampling variance in Figure 13(b) is higher.

Expert mismatch across checkpoints. Figure 14(a) compares the four checkpoints on six VLA-Arena suites. Mean squared-bias proxies are 0.270 for FastWAM, 0.521 for π_0, 1.551 for SmolVLA, and 2.649 for $\pi_{0.5}$; corresponding fixed-1 minus fixed-10 success differences are -3.60, $+2.67$, $+8.87$, and $+23.53$ percentage points. The rank association is $\rho = 1$ with exact two-sided $p = 0.083$. These exploratory comparisons mix training recipes and sampled state distributions. Within models, SmolVLA has the largest positive task-level correlation estimate in Figure 14(b), while the other three estimates are closer to zero.

Table 5: **Measured variance trends depend on policy and action component.** Spearman correlations between NFE and seed variance are computed within each expert frame over $\{1, 2, 4, 10\}$, then averaged within episodes, tasks, and suites with equal weights at each level. The last column gives the corresponding weighted fraction with nondecreasing variance.

Policy	Component	Mean within-frame ρ	Nondecreasing frames (%)
FastWAM	Position	−0.667	1.8
	Rotation	−1.000	0.0
	Gripper	0.077	11.3
π_0	Position	0.117	43.5
	Rotation	−0.954	0.0
	Gripper	−0.707	11.7
SmolVLA	Position	0.492	65.5
	Rotation	−1.000	0.0
	Gripper	−0.521	20.5
$\pi_{0.5}$	Position	0.730	80.5
	Rotation	−0.261	0.1
	Gripper	−0.260	24.7

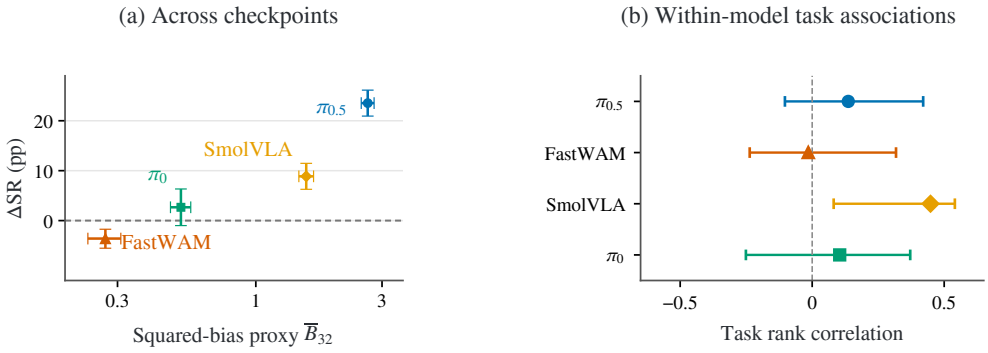


Figure 14: **Squared bias and the success benefit of budget reduction.** (a) Four checkpoints averaged over six VLA-Arena suites: NFE-32 squared-bias proxy $\bar{B}_{32}$ versus fixed-1 minus fixed-10 success difference ΔSR (pp), with a logarithmic horizontal axis and marginal 95% intervals on both axes. (b) Within-model task rank correlations between the same squared-bias proxy and success difference, after adjusting for suite. Pointwise 95% intervals are obtained by resampling expert episodes and paired rollout outcomes within tasks, holding the task set fixed. Both panels concern the squared-bias proxy rather than alignment gain.

Expert mismatch over task progress. The $\pi_{0.5}$ squared-bias proxy rises from 1.549 early to 4.273 late, with the same direction in all six suites. SmolVLA shows a similar gradient, π_0 is nearly flat, and FastWAM has a heavy mid-phase tail rather than a consistent gradient.

C.2.2 Measured Sampling Variance Across Budgets

Sampling variance responds differently to budget across policies and action components. Position variance rises in many $\pi_{0.5}$ and SmolVLA frames, while rotation variance decreases across all four policies on average. For action component g (position, rotation or gripper), the empirical sampling variance is

$$\widehat{v}_{k,g}(s) = \frac{1}{J} \sum_{j=1}^{J} \|a_{k,g}(s, \xi_j) - \widehat{\mu}_{k,g}(s)\|^2. \tag{14}$$

Full-block $\widehat{\mathcal{V}}_k$ is the sum over these disjoint components. Figure 13(b) shows its across-state average. Large-variance frames and components can dominate this mean, so its budget trend need not match that of a typical frame or component.

For Figure 5(a), we compute the Spearman correlation between NFE and $\widehat{v}_{k,g}(s)$ within each frame over $k \in \{1, 2, 4, 10\}$, then apply the episode–task–suite averaging above. Table 5 reports the same correlations and the weighted fraction of frames whose variance is nondecreasing across these four budgets. NFE 32 supplies the numerical reference but is excluded from these trend statistics. These are descriptive summaries of variation over sampler seeds at fixed expert states, rather than epistemic uncertainty or variability along a rollout. We use $1/J$ normalization for empirical sampling variance; this is not an unbiased population-variance estimator.

C.2.3 Mean-Action Alignment and Task-Level Comparisons

Equation (5) measures the reduction in mean-action expert distance from replacing k_2 with k_1. Subtracting Equation (12) across budgets gives

$$G_{\text{align}}(s; k_1, k_2) = \|b_{k_2}\|^2 - \|b_{k_1}\|^2 + 2\langle \mu_\infty - e, b_{k_2} - b_{k_1} \rangle,$$

where the shared B_∞ cancels and state arguments are suppressed. Using the sample means, the finite-seed estimate is

$$\begin{aligned} \widehat{G}_{\text{align}}(s; k_1, k_2) &= \|\widehat{\mu}_{k_2} - e\|^2 - \|\widehat{\mu}_{k_1} - e\|^2 \\ &= (N_{k_2} - N_{k_1}) + (X_{k_2} - X_{k_1}), \end{aligned} \tag{15}$$

where $N_k = \|\widehat{\mu}_k - \widehat{\mu}_{32}\|^2$ and $X_k = 2\langle \widehat{\mu}_{32} - e, \widehat{\mu}_k - \widehat{\mu}_{32} \rangle$ are the finite-reference numerical-mean-bias and cross-terms in Equation (12). The shared proxy $\widehat{B}_{32}$ in Equation (13) cancels.

Figure 5(b) shows all 30 task estimates per checkpoint for $(k_1, k_2) = (1, 10)$; positive gain favors NFE 1. Panels share the success axis but use separate alignment-axis ranges. Tasks in the lower-right quadrant have better expert alignment but lower success. Independently sampled probes and rollouts are compared through task-level means under the aggregation defined above. The association tests concern the two components separately; the summed alignment gain is shown descriptively.

D Learning Budget Allocation

We use paired success and cost outcomes to train FBA, then test how budget choices translate into closed-loop performance through call-position, continuation and real-robot comparisons.

D.1 Learning from Paired Budget Outcomes

For each $k_i \in \mathcal{K}$, $i = 1, \ldots, M$, where $M = |\mathcal{K}|$, the predictor in Figure 15 estimates success and remaining NFE under training continuation $c_{\text{train}} = k_{\min}$, with $k_{\min} = 1$ in our experiments. The action policy is frozen; only FBA is trained.

D.1.1 Constructing Training Data from Budget Measurements

Reference trajectories and event-anchor roots. Each task specifies an instruction, scene and goal; multiple initial configurations instantiate that task. The data pool contains 2,250 $\pi_{0.5}$ fixed-1 reference episodes: nine L0 suites, five tasks per suite, and 50 initial configurations per task. We apply the event detection and anchor-to-call mapping defined in Section 2.3 to select eligible root calls along each reference. The ordinary pool contains 6,748 such nodes. Each root retains its event identity, reference identity, simulator state, observation, and policy history. The event identifies where to collect supervision; it is not supplied as an input to FBA.

After restoration, matched-noise branches execute the candidate budgets at the current call and continue at NFE 1 to native termination or the episode limit. One matched future stream supplies an outcome for each budget. This single-call intervention produces the success–cost labels for choosing at the selected root. Roots are retained regardless of success or outcome differences across budgets.

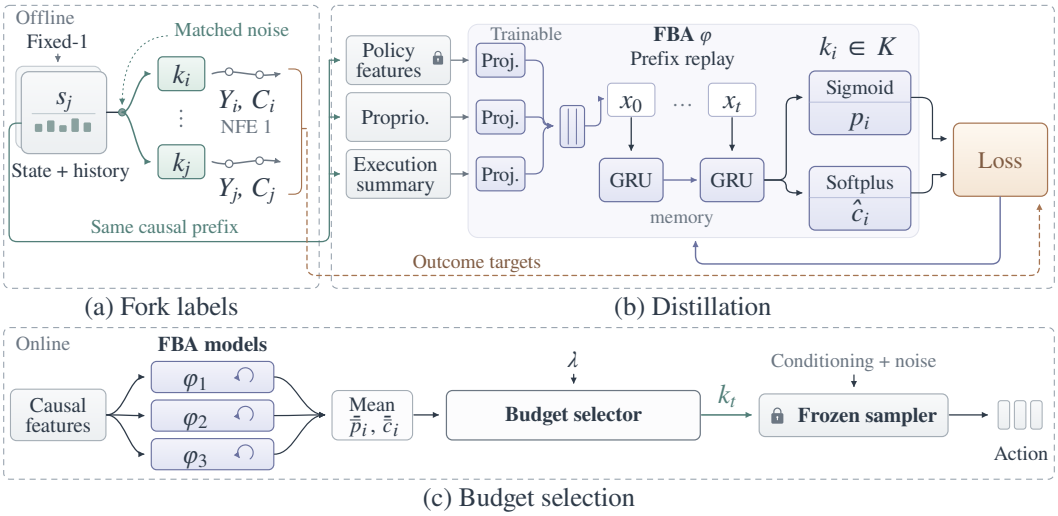


Figure 15: **Counterfactual outcome distillation in simulation.** (a) Event anchors on fixed-1 references locate matched single-call budget forks, producing (Y_i, C_i) for each $k_i \in \mathcal{K}$ with NFE 1 continuation. (b) A gated recurrent unit (GRU) takes decision-time feature histories as input and predicts success values p_i and normalized remaining cost $\widehat{c}_i$ for each candidate budget. Paired branch outcomes provide supervision; Proj. denotes projection. (c) An ensemble selects the frozen sampler's budget.

Causal inputs and paired outcome targets. At root call t, let $\mathcal{H}_t$ contain the feature rows from calls 0 through t. A row contains the policy's hidden representation and proprioception before the current action, together with a summary of the preceding executed call. All budgets share this prefix. For each k_i, we record terminal success y_i and actual remaining cumulative NFE, including the current budget-k_i call and all continuation calls but excluding the cost already spent before the root. The latter is divided by 100 to obtain c_i. The resulting record $(\mathcal{H}_t, \{(y_i, c_i, m_i)\}_{i=1}^{M})$ contains $y_i = Y_s(k_i, 1; W)$ and $c_i = C_s(k_i, 1; W)/100$ for $W = [t, t+1)$, with m_i marking an observed label. Future states and outcomes supply labels only; all candidate outcomes remain paired within the example.

Training on states reached after a budget choice. A budget choice also changes the next decision state. The mixed condition extends supervision to these successor states through nested forks at selected event anchors. Each candidate budget at a nested root produces its own actual successor state and executed history. Each nonterminal successor is branched over the candidate budgets once more, followed by NFE 1 to termination. Its training prefix appends the executed parent call and the successor's current features to the reference history.

To retain the single-call target at a nested parent, its label uses the child branch with budget 1 and adds the parent's paid NFE to that child's remaining cost. Successor labels measure cost from the successor onward. A parent call that terminates contributes its observed outcome and paid cost without a successor node. Both parent and successor labels therefore follow the specified continuation, using executed branches rather than a best-leaf replacement. This expansion adds 2,250 roots and 8,992 successor nodes to the ordinary corpus. Table 6 counts root and successor examples.

Both data pools contain complete matched groups, with every budget label observed and no groups skipped. Unavailable executions are recorded separately from task failures.

D.1.2 Representation and Recurrent Architecture

The lightweight FBA contains 714,088 parameters per member, totaling 2,142,264 for the three-member ensemble. Its policy features and the action sampler's KV cache come from the same prefix computation, requiring no additional encoder forward pass.

Table 6: **Data pools constructed from matched measurements.** Both conditions derive from the same 2,250 L0 references. Each retained root or successor state counts as one example with four observed budget labels.

Training condition	Root nodes	Successor nodes	Total nodes
Ordinary	6,748	0	6,748
Mixed	8,998	8,992	17,990

At decision t, the input consists of the policy's hidden feature x_t, proprioception u_t, and preceding-call summary v_t. For $\pi_{0.5}$, the 2,048-dimensional policy feature passes through a linear projection, layer normalization and SiLU to 256 dimensions. Proprioception contains 31 continuous coordinates and five validity indicators. Continuous coordinates are standardized using training data and masked by their group validity before a 64-dimensional projection. The 16-dimensional preceding-call summary contains the mean and last executed actions, the budget, and duration; its projection has 32 dimensions. A 128-dimensional GRU processes their concatenation:

$$\begin{aligned} h_t &= \mathrm{GRU}([\phi_x(x_t); \phi_u(u_t); \phi_v(v_t)], h_{t-1}), \\ p_{t,i} &= \sigma(w_i^\top h_t + b_i), \qquad \widehat{c}_{t,i} = \mathrm{softplus}(a_i^\top h_t + d_i). \end{aligned} \tag{16}$$

The projections, GRU, and output heads together have parameters ϕ. Initial-call history is zero. Suite, task and stage identities are excluded from the inputs. At deployment, each ensemble member maintains its own recurrent state.

D.1.3 Masked Outcome and Paired-Difference Losses

Selecting a budget requires both its expected outcome and its advantage over the alternatives. We therefore fit absolute success and cost alongside the paired differences between budgets. At a simulated node at call t, let $W = [t, t+1)$, $y_i = Y_s(k_i, c_{\mathrm{train}}; W)$, $c_i = C_s(k_i, c_{\mathrm{train}}; W)/100$, and $m_i \in \{0, 1\}$ indicate an available label. The success head is $p_i = \widehat{Q}_\phi(\mathcal{H}_t, k_i)$, with $\Delta\widehat{Q}_{\phi,ij} = p_i - p_j$. Matched labels satisfy $\mathbb{E}[y_i - y_j \mid s] = \Delta Q_{1,W}(s; k_i, k_j)$. Their realized difference is a noisy target, not an observed success probability difference at an individual state. FBA predicts from $\mathcal{H}_t$ rather than the full simulator state; its population target averages these values conditional on the available history. We suppress the node index and write $D = \max(1, \sum_i m_i)$ and $P = \max(1, \sum_{i<j} m_i m_j)$. With $\Delta a_{ij} = a_i - a_j$ and $\ell_{\mathrm{BCE}}(p, y) = -y \log p - (1-y)\log(1-p)$, the node loss is

$$L_{\mathrm{arm}} = \frac{1}{D} \sum_i m_i \left[\ell_{\mathrm{BCE}}(p_i, y_i) + (\widehat{c}_i - c_i)^2 \right], \tag{17}$$

$$L_{\mathrm{pair}} = \frac{1}{P} \sum_{i<j} m_i m_j \left[(\Delta\widehat{Q}_{\phi,ij} - \Delta y_{ij})^2 + (\Delta\widehat{c}_{ij} - \Delta c_{ij})^2 \right], \tag{18}$$

$$L_{\mathrm{sim}} = L_{\mathrm{arm}} + 0.5 L_{\mathrm{pair}}. \tag{19}$$

Binary cross-entropy uses logits for numerical stability. Pairwise terms use probabilities and normalized costs. Missing labels contribute to neither numerator nor denominator; a node with fewer than two valid arms has zero pairwise loss. Minibatches average these node-normalized losses.

Optimization and deployment. For each training condition, FBA uses the full L0 corpus described in Section D.1.1. We use AdamW with learning rate 3×10^{-4}, weight decay 10^{-4}, batch size 16 and gradient-norm clipping at 1.0 for 60 epochs. Each epoch makes N draws from the training corpus, where N is its node count, sampling uniformly through suite, task, reference, decision node and acquisition repeat. Each sequence resets the GRU state and replays its causal prefix; loss is applied only at the last valid, unpadded call.

Three independent training seeds produce final-epoch checkpoints; deployment averages their probabilities and costs after output activation. Final mean training losses range from 0.177 to 0.180 for ordinary data and from 0.163 to 0.167 for mixed data. A separate memorization probe checks the

ability to fit a small label set: training on 256 nodes for 300 epochs reduces success BCE from 0.738 to 0.041 on that subset. These are training-fit diagnostics.

Training uses NVIDIA A100 GPUs. The primary simulation comparison in Table 1 and real-robot evaluation use NVIDIA RTX 4090 GPUs. The continuation experiment and the fixed 1, After-first, and Full call-position conditions use A100 80GB GPUs. These evaluations use one policy server per GPU and inference batch size one; the A100 experiments add no deployment-latency measurements.

Budget selection. For ensemble averages $\bar{p}_i, \bar{c}_i$ of $p_i, \widehat{c}_i$, let r index $k_{\min}$. Select k_{i^*} with $i^* = \arg\max_i[(\bar{p}_i - \bar{p}_r) - \lambda(\bar{c}_i - \bar{c}_r)]$. Subtracting the common baseline leaves the maximizer of $\bar{p}_i - \lambda\bar{c}_i$ unchanged. The first difference is the ensemble's predicted success-value gain over $k_{\min}$; the second is its predicted additional normalized cost. If $\max_i \bar{p}_i - \min_i \bar{p}_i < 0.01$, use $k_{\min}$. Equal utility scores are resolved by lower predicted cost, then lower budget. The score uses normalized remaining NFE, not measured call latency.

D.1.4 Comparing Complete Allocation Strategies

The allocation comparison evaluates $\pi_{0.5}$ and includes fixed budgets, ProbeFlow, AdaFlow, and FBA. All methods start from the same applicable checkpoint and use matched evaluation tasks and initial conditions. Following the $\pi_{0.5}$ policy's L0 training and evaluation setting in VLA-Arena (Zhang et al., 2026), we train FBA and AdaFlow on L0 data and evaluate them in closed loop within that distribution. L0 evaluation spans all five tasks per suite and all 2,250 task–initialization configurations in the reference pool, with fresh execution seeds and newly executed complete episodes. L1/L2 evaluation uses the same L0-trained checkpoints without updating their weights. Table 1 reports the aggregate development results.

Success, cumulative computation and call latency. Table 1 pools nine suites, each with five tasks and 50 initializations per task: 2,250 episodes per method and level, with one evaluation execution per configuration. Fixed-budget and FBA runs are paired by suite, task and initialization, using matched execution seeds within each level. For all methods, the $\pm$ values give task-stratified 95% interval half-widths, with $H = 45$ suite-task cells weighted equally. For a cell with $n_h = 50$ episodes and sample variance s_h^2, the variance of the aggregate mean is estimated as $\widehat{V} = H^{-2}\sum_{h=1}^{H} s_h^2/n_h$; intervals use the mean $\pm 1.96\sqrt{\widehat{V}}$. Paired differences and single-model training-seed variation are reported separately below. In both simulation and real-robot evaluation, latency is the median (p50) of per-call elapsed wall-clock times recorded during the actual evaluation runs, from call start to completion after JAX compilation warmup. The measurement covers the complete policy invocation and includes FBA computation for the adaptive method.

Baseline definitions. Fixed budgets hold NFE constant. On frozen $\pi_{0.5}$, we faithfully reproduce ProbeFlow's published algorithm (Fang et al., 2026), including its velocity-similarity criterion and probe-reuse rule. NFE counts all velocity evaluations, including probes. For AdaFlow (Hu et al., 2024), we train a variance predictor while keeping $\pi_{0.5}$ frozen and use its predictions to adapt integration step sizes. The comparison evaluates each complete inference strategy using closed-loop success and cumulative NFE.

ProbeFlow configuration. We follow the published main configuration (Fang et al., 2026): $N_{\min} = 2$, $N_{\max} = 10$, $\Delta N = 2$, $\epsilon = 0.008$, and lookahead horizon 0.5. The integration schedules are $N \in \{2, 4, 6, 8, 10\}$. With the published reuse rule, two probe evaluations are reused at the minimum; larger schedules discard the lookahead evaluation and reuse the initial velocity. Counting these evaluations gives actual NFE $N + \mathbf{1}\{N > 2\}$, or $\{2, 5, 7, 9, 11\}$ for the five schedules.

AdaFlow supervision and configuration. The variance predictor uses the same full L0 training corpus as FBA. Following AdaFlow (Hu et al., 2024), for a training condition o and action block a, draw $x_0 \sim \mathcal{N}(0, I)$ and flow time $\tau \sim \text{Uniform}[0, 1]$, and form $x_\tau = (1 - \tau)x_0 + \tau a$. The predictor $\sigma_\psi^2(x_\tau, \tau \mid o)$ is trained with AdaFlow's Gaussian negative-log-likelihood objective using the squared velocity residual

$$R_\tau = \|a - x_0 - v_\theta(x_\tau, \tau \mid o)\|_2^2,$$

where v_θ is the frozen policy's velocity field. Only the variance predictor is updated. Its optimization hyperparameters match FBA: AdamW, learning rate 3×10^{-4}, weight decay 10^{-4}, batch size 16, gradient-norm clipping at 1.0 and 60 epochs, using the final-epoch checkpoint.

Inference follows AdaFlow's variance-adaptive Euler rule. In noise-to-action flow time, initialize $z_0 \sim \mathcal{N}(0, I)$ and use

$$h_\tau = \min\left\{1 - \tau, \max\left\{\epsilon_{\min}, \frac{\eta}{\sigma_\psi(z_\tau, \tau \mid o)}\right\}\right\},$$
$$z_{\tau+h_\tau} = z_\tau + h_\tau v_\theta(z_\tau, \tau \mid o).$$

We use the original robot-manipulation setting $\eta = 1.0$ and $N_{\max} = 10$, with $\epsilon_{\min} = 1/N_{\max} = 0.1$. The final step is clipped to the remaining flow time; zero predicted variance takes the entire remaining step. Each Euler update uses one velocity evaluation, giving 1–10 actual NFE per policy call. The variance-head prediction is not an additional velocity evaluation.

Section D.5 details the physical-data training objective, cost target and HIL collection.

D.2 Five-Fold Diagnostics of Budget Prediction

Predicting whether an episode will succeed is only part of budget allocation: the predictor must also distinguish outcomes under alternative budgets at the same state. We examine these two abilities and the information provided by each input channel using five-fold cross-validation grouped by reference trajectory. The 2,250 L0 references are partitioned into five folds of 450 references, with all nodes from each reference kept in the same fold. Each diagnostic model trains on four folds and predicts outcomes on the remaining fold. These diagnostics evaluate prediction on held-out references within the covered suites. The five-fold split applies only to these separate diagnostic models. Each diagnostic is a development experiment on its stated population.

D.2.1 Which Inputs Help Predict Outcomes?

On the same mixed-data diagnostic split, macro-averaged success AUC is 0.927 with policy hidden features alone, 0.929 after adding proprioception, and 0.932 with the full preceding-call summary. Thus frozen policy features carry most of the observed predictive signal, with modest gains from the additional channels. These differences do not isolate the benefit of GRU memory, since all three variants retain the recurrent model; a parameter-matched memoryless model remains untested.

D.2.2 From Outcome Prediction to Budget Discrimination

We test whether accurate outcome prediction also distinguishes budgets at matched states. On held-out references from the mixed-data diagnostic, success prediction reaches macro-AUC 0.932 with 95% interval $[0.925, 0.937]$. Outcome AUC evaluates success prediction; sign agreement evaluates the ordering of budgets at matched nodes where two candidate outcomes differ. For a fixed pair (k_i, k_j), let $\mathcal{D} = \{n : y_{n,i} \neq y_{n,j}\}$. For a nonempty set, define

$$A_{ij} = \frac{1}{|\mathcal{D}|} \sum_{n \in \mathcal{D}} \mathbf{1}\big\{(p_{n,i} - p_{n,j})(y_{n,i} - y_{n,j}) > 0\big\}. \tag{20}$$

Prediction ties count as disagreement. For the lowest/highest budget pair, 1,647 of 17,990 diagnostic nodes have different outcomes, and agreement is 0.721 with reference-cluster 95% interval $[0.695, 0.748]$. A within-node arm-label permutation probe returns 0.500 with interval $[0.473, 0.527]$. The predictor distinguishes observed relative outcomes above chance on held-out references; label permutation removes this separation.

The distinction between prediction and discrimination also appears across the two diagnostic populations. Ordinary data yield macro-AUC 0.948 and sign agreement 0.695, whereas mixed data yield 0.932 and 0.721, with 573 versus 1,647 discriminating nodes. Thus the ordering by outcome AUC differs from the ordering by sign agreement on these respective populations.

Table 7: **Full-benchmark FBA configurations on $\pi_{0.5}$.** Each level pools nine suites with 250 episodes each. Ordinary uses event-root examples; mixed adds nested roots and their executed successor states. λ weights predicted cost.

Configuration	Success (%): L0 / L1 / L2	Mean ΣNFE: L0 / L1 / L2
Ordinary $\lambda = 0.5$	85.5 / 49.2 / 34.4	23.4 / 40.4 / 45.7
Ordinary $\lambda = 1$	86.1 / 51.0 / 35.2	20.8 / 34.8 / 40.1
Ordinary $\lambda = 2$	86.6 / 49.2 / 36.9	18.7 / 30.8 / 34.9
Mixed $\lambda = 0.5$	85.7 / 48.9 / 33.7	24.5 / 49.2 / 53.8
Mixed $\lambda = 1$	85.4 / 48.8 / 33.8	22.0 / 39.8 / 43.0
Mixed $\lambda = 2$	86.3 / 48.8 / 34.3	19.6 / 32.9 / 35.8

D.3 Success and Cost under Learned Allocation

D.3.1 Training Data and Cost Preference

Training data determine which responses FBA sees, while λ determines how it trades predicted success against cost. We examine both choices across all nine suites.

The evaluated conditions combine ordinary or mixed training with $\lambda \in \{0.5, 1, 2\}$, giving six ensemble configurations per level. Table 7 displays all six configurations over the nine suites, pooling 2,250 episodes per configuration and level. Table 1 reports ordinary training with $\lambda = 2$ at L0 and L2 and $\lambda = 1$ at L1. Success and cost use the same configuration at each level.

Paired success and cost differences. Table 8 reports all six configurations against fixed 1. Each comparison pairs 2,250 episodes by suite, task and initialization. We compute each episode's paired success and cost differences and apply the same task-stratified variance estimate, using within-cell sample variances of those differences and weighting the 45 cells equally. Success differences are in percentage points; cost differences are in cumulative NFE per episode. Intervals are pointwise 95% development intervals for each configuration. They summarize evaluation-episode uncertainty conditional on the trained models and the evaluated suite-task set.

Increasing λ from 0.5 to 2 lowers observed cumulative NFE for both training conditions at every level, but success need not fall with it. With ordinary training at L0, mean NFE decreases from 23.4 to 18.7 while observed success rises from 85.5% to 86.6%. The mixed-data diagnostic has higher sign agreement on its own population, yet this ordering does not carry over to the reported closed-loop results. The diagnostic and deployment evaluations therefore answer different questions: one tests relative-outcome prediction on its sampled nodes, the other measures success and cost along FBA's executed trajectories.

All six configurations improve observed L0 success over fixed 1 and use more cumulative NFE. At L1, ordinary $\lambda = 1$ improves observed success over fixed 1 by 3.02 pp; at L2, ordinary $\lambda = 2$ improves it by 2.27 pp.

D.3.2 L0 Single-Model Training-Seed Variation

Table 9 evaluates the three independently trained models from each data condition separately at $\lambda = 1$, using the same 2,250 matched L0 evaluation configurations. Run numbers identify the three training seeds. These are single-model deployments; the main comparison averages their predictions in an ensemble and also varies λ. The intervals use the same paired task-stratified variance estimate as Table 8.

All six individually deployed models improve observed L0 success over fixed 1, with success and cost varying across seeds. These runs characterize single-model variation at L0; L1/L2 have ensemble evaluations only, without independently retrained ensemble replicates.

Table 8: **Paired FBA differences from fixed NFE 1.** Positive success differences favor allocation; positive NFE differences mean added computation. Bold marks the configurations reported in Table 1.

Level	Configuration	Δsuccess (pp) [95% CI]	ΔΣNFE [95% CI]
L0	Ordinary $\lambda = 0.5$	$+3.91\ [2.45, 5.37]$	$+8.28\ [7.85, 8.71]$
	Ordinary $\lambda = 1$	$+4.53\ [3.10, 5.96]$	$+5.69\ [5.35, 6.03]$
	Ordinary $\lambda = 2$	$+5.02\ [3.65, 6.39]$	$+3.56\ [3.27, 3.85]$
	Mixed $\lambda = 0.5$	$+4.09\ [2.63, 5.55]$	$+9.40\ [8.80, 10.01]$
	Mixed $\lambda = 1$	$+3.78\ [2.36, 5.20]$	$+6.83\ [6.34, 7.33]$
	Mixed $\lambda = 2$	$+4.67\ [3.27, 6.07]$	$+4.42\ [4.05, 4.79]$
L1	Ordinary $\lambda = 0.5$	$+1.20\ [-0.41, 2.81]$	$+18.56\ [17.85, 19.26]$
	Ordinary $\lambda = 1$	$+3.02\ [1.45, 4.59]$	$+13.01\ [12.49, 13.54]$
	Ordinary $\lambda = 2$	$+1.29\ [-0.18, 2.76]$	$+8.97\ [8.56, 9.38]$
	Mixed $\lambda = 0.5$	$+0.93\ [-0.80, 2.67]$	$+27.38\ [26.29, 28.47]$
	Mixed $\lambda = 1$	$+0.84\ [-0.78, 2.47]$	$+18.01\ [17.24, 18.77]$
	Mixed $\lambda = 2$	$+0.80\ [-0.80, 2.40]$	$+11.05\ [10.54, 11.57]$
L2	Ordinary $\lambda = 0.5$	$-0.18\ [-1.55, 1.19]$	$+21.36\ [20.65, 22.07]$
	Ordinary $\lambda = 1$	$+0.53\ [-0.80, 1.87]$	$+15.72\ [15.15, 16.29]$
	Ordinary $\lambda = 2$	$+2.27\ [0.90, 3.63]$	$+10.55\ [10.14, 10.97]$
	Mixed $\lambda = 0.5$	$-0.93\ [-2.35, 0.49]$	$+29.42\ [28.34, 30.50]$
	Mixed $\lambda = 1$	$-0.84\ [-2.22, 0.53]$	$+18.69\ [17.95, 19.43]$
	Mixed $\lambda = 2$	$-0.36\ [-1.69, 0.98]$	$+11.44\ [10.96, 11.92]$

Table 9: **Single-model FBA results across training seeds at L0.** All models use $\lambda = 1$; brackets give pointwise paired 95% development intervals for success differences from fixed 1.

Training data	Run	Success (%)	Mean ΣNFE	Δsuccess vs. fixed 1 (pp)
Ordinary	1	86.9	19.7	$+5.29\ [3.79, 6.79]$
Ordinary	2	85.7	20.0	$+4.09\ [2.72, 5.46]$
Ordinary	3	84.8	25.0	$+3.24\ [1.71, 4.77]$
Mixed	1	86.7	22.5	$+5.07\ [3.62, 6.52]$
Mixed	2	85.8	22.8	$+4.18\ [2.67, 5.68]$
Mixed	3	86.5	21.3	$+4.89\ [3.50, 6.28]$

D.4 Call-Position and Continuation Comparisons

FBA learns from single-call effects and makes a sequence of choices at deployment. We examine this step in two ways: first by controlling which calls may use allocation, then by testing the same current choice under fixed and adaptive continuation. Both experiments freeze the ordinary three-seed $\pi_{0.5}$ FBA ensemble, use $\lambda = 2$, cost scale 100, and $\mathcal{K} = \{1, 2, 4, 10\}$, with the same success-spread fallback as the primary evaluation. No weights are updated. The fixed 1, After-first, and Full episode evaluations and the continuation experiment use A100 80GB GPUs, one policy server per GPU, and batch-one inference with concurrent simulator workers. First-only success outcomes come from an additional evaluation. Episode NFE counts actual solver evaluations; collection throughput is not a latency measure.

D.4.1 Call-Position Ablation

To separate the initial choice from later choices, we evaluate four policies that differ only in where allocation is allowed. Each of the nine L0 suites contributes five tasks and 50 initial configurations per task. Four conditions run all 2,250 configurations: fixed 1; First-only, which lets FBA choose at the initial call; After-first, which lets it choose at every subsequent call; and Full, which lets it choose throughout. Restricted calls execute NFE 1. Every condition advances its recurrent state

Table 10: **Closed-loop outcomes with allocation at different call positions.** Each condition uses 2,250 L0 episodes and the frozen FBA ensemble. First-only selects the initial budget, After-first selects at every later call, and Full selects at every call; calls without allocation use NFE 1.

Budget selection	Success (%)	Mean ΣNFE
Fixed NFE 1	80.67	15.29
First-only	83.11	15.07
After-first	83.78	18.74
Full	86.09	18.87

Table 11: **First-only outcomes across suites.** The frozen FBA ensemble selects the initial budget; all subsequent calls use NFE 1. Each suite contributes 250 L0 episodes.

Suite	Successful episodes	Success (%)	95% interval (%)
Dynamic distractor	239/250	95.60	$[92.80, 98.00]$
Static distractor	243/250	97.20	$[95.20, 99.20]$
Preposition	165/250	66.00	$[60.40, 71.20]$
Unseen objects	201/250	80.40	$[75.60, 85.20]$
Cautious grasp	172/250	68.80	$[63.20, 74.00]$
Dynamic obstacles	220/250	88.00	$[84.40, 91.20]$
Hazard avoidance	174/250	69.60	$[65.60, 73.20]$
State preservation	213/250	85.20	$[80.80, 89.20]$
Static obstacles	243/250	97.20	$[95.20, 99.20]$
All suites	**1,870/2,250**	**83.11**	**[81.78,84.40]**

using its own executed history, including calls with a forced budget. First-only therefore still makes a learned, state-conditioned choice at the initial call.

The fixed 1, After-first, and Full conditions start from the same saved simulator and controller state and observation, with no prior execution history. Their fresh execution streams are matched by configuration and logical call on the L0 task–initialization grid. Together with the separate First-only evaluation, the reported comparison contains 2,250 episodes per condition, or 9,000 in total.

Our primary comparison, Full versus First-only, examines the additional benefit and cost of allowing allocation after the initial call. The three secondary comparisons assess initial-only allocation through First-only versus fixed 1, later-only allocation through After-first versus fixed 1, and the addition of an initial choice through Full versus After-first.

Initial and later choices both contribute. First-only and After-first each improve observed success over fixed 1, and Full attains the highest observed success in Table 10. Full produces 67 more successful episodes than First-only, corresponding to $+2.98$ pp $[1.56, 4.44]$ at $+3.80$ mean NFE $[3.51, 4.09]$. First-only improves success over fixed 1 by 2.44 pp, or 55 additional successful episodes, while changing mean episode NFE by -0.21. After-first improves on fixed 1 by 3.11 pp; adding the initial decision to it gives Full a further 2.31 pp.

The gains arise at different positions across suites. First-only's outcomes vary across suites in Table 11. Its gain over fixed 1 on Cautious Grasp is 32.80 pp, or 82 additional successful episodes. Losses in other suites partly offset this gain, yielding 55 additional successes overall. Full's largest observed gain over First-only is on Preposition Combinations, 8.80 pp. The complete suite comparisons show both gains and losses from adding allocation at particular positions.

The 95% intervals for comparisons involving First-only appear in Table 12. For After-first versus fixed 1 and Full versus After-first in Table 13, we use 10,000 paired configuration bootstrap draws within the 45 suite–task strata, with equal task and suite weights. These pointwise 95% intervals condition on the frozen ensemble, without variation from retraining.

Table 12: **Call-position effects: primary comparison and initial allocation.** Entries show estimates with 95% intervals for comparisons involving First-only. The notation $x^{+u}_{-\ell}$ gives bounds $[x-\ell, x+u]$; $\pm$ denotes equal lower and upper errors. Each suite contains 250 episodes; all-suite estimates pool 2,250. ΔNFE is the difference in mean cumulative NFE per episode.

	Full − First-only		First-only − fixed 1	
Population	Δsuccess (pp)	ΔNFE	Δsuccess (pp)	ΔNFE
All suites	$+2.98^{+1.46}_{-1.42}$	$+3.80 \pm 0.29$	$+2.44^{+1.29}_{-1.28}$	$-0.21^{+0.19}_{-0.20}$
Dynamic distractor	$+3.20 \pm 2.40$	$+3.74 \pm 0.49$	-3.20 ± 2.40	$+0.48^{+0.31}_{-0.30}$
Static distractor	$+2.00 \pm 2.00$	$+2.07^{+0.63}_{-0.56}$	$-1.20^{+1.20}_{-1.60}$	$+0.16^{+0.05}_{-0.04}$
Preposition	$+8.80^{+5.20}_{-4.80}$	$+5.61^{+1.14}_{-1.06}$	-0.40 ± 5.20	$+0.83 \pm 0.73$
Unseen objects	$+3.20^{+5.20}_{-4.80}$	$+3.92^{+1.05}_{-0.98}$	-0.80 ± 3.60	-0.21 ± 0.48
Cautious grasp	$+4.40 \pm 7.20$	$+5.96^{+1.29}_{-1.28}$	$+32.80 \pm 7.20$	-4.65 ± 1.22
Dynamic obstacles	$+1.20 \pm 2.40$	$+2.30^{+0.44}_{-0.36}$	$-3.60^{+2.00}_{-2.40}$	$+0.41^{+0.32}_{-0.26}$
Hazard avoidance	$+2.80 \pm 4.40$	$+4.90^{+0.75}_{-0.72}$	$-0.40^{+2.80}_{-3.20}$	$+0.60^{+0.48}_{-0.46}$
State preservation	$-0.40^{+5.20}_{-4.80}$	$+3.73^{+1.27}_{-1.20}$	$+0.80 \pm 4.40$	$+0.18^{+0.71}_{-0.70}$
Static obstacles	$+1.60^{+2.40}_{-2.00}$	$+1.94^{+0.33}_{-0.27}$	$-2.00^{+2.00}_{-2.40}$	$+0.29^{+0.11}_{-0.19}$

Table 13: **Call-position effects: subsequent allocation and the added initial choice.** Each suite contributes 250 episodes; all-suite estimates pool 2,250 with equal task and suite weights. Entries show estimates with pointwise paired 95% intervals: $x^{+u}_{-\ell}$ gives bounds $[x-\ell, x+u]$, with $\pm$ for equal lower and upper errors. ΔNFE is the difference in mean cumulative NFE per episode.

	After-first − fixed 1		Full − After-first	
Population	Δsuccess (pp)	ΔNFE	Δsuccess (pp)	ΔNFE
All suites	$+3.11 \pm 1.42$	$+3.46^{+0.30}_{-0.29}$	$+2.31^{+1.07}_{-1.11}$	$+0.13^{+0.24}_{-0.25}$
Dynamic distractor	-0.40 ± 1.60	$+4.15^{+0.59}_{-0.53}$	$+0.40^{+1.60}_{-1.20}$	$+0.07^{+0.52}_{-0.57}$
Static distractor	-0.40 ± 2.00	$+2.35^{+0.72}_{-0.63}$	$+1.20^{+1.60}_{-1.20}$	$-0.12^{+0.26}_{-0.36}$
Preposition	$+2.80 \pm 4.40$	$+5.66^{+0.91}_{-0.94}$	$+5.60 \pm 4.40$	$+0.78^{+1.10}_{-1.08}$
Unseen objects	$+2.80 \pm 5.20$	$+3.88^{+1.10}_{-1.03}$	-0.40 ± 2.00	$-0.17^{+0.49}_{-0.43}$
Cautious grasp	$+23.60^{+7.60}_{-7.20}$	$+1.74^{+1.27}_{-1.30}$	$+13.60 \pm 6.40$	$-0.43^{+1.15}_{-1.11}$
Dynamic obstacles	-0.80 ± 1.60	$+2.37^{+0.42}_{-0.38}$	$-1.60^{+1.20}_{-1.60}$	$+0.34^{+0.28}_{-0.24}$
Hazard avoidance	$+0.80 \pm 4.40$	$+5.28^{+0.81}_{-0.77}$	$+1.60 \pm 2.80$	$+0.22^{+0.52}_{-0.57}$
State preservation	$+0.40 \pm 5.20$	$+3.75^{+1.42}_{-1.36}$	0.00 ± 3.60	$+0.16^{+1.08}_{-1.12}$
Static obstacles	$-0.80^{+0.80}_{-1.20}$	$+1.94^{+0.32}_{-0.26}$	$+0.40^{+0.80}_{-0.40}$	$+0.28^{+0.17}_{-0.22}$

D.4.2 Does a Choice Retain Its Value under Adaptive Continuation?

The episode comparison changes an entire decision sequence. We now fix the current state and selected budget, varying later budget choices to test how the single-call effect depends on continuation.

We select five references per suite–task from the Full condition before examining outcomes, for 225 references. Each contributes its initial call and one noninitial call chosen uniformly by reservoir sampling, yielding 225 roots of each type. At each root, we compare the frozen FBA model's preselected budget $k^*(s)$ with NFE 1. Selection precedes branch outcomes; branches coincide and are deduplicated when $k^*(s) = 1$. Each current choice is followed either by fixed 1 or by the frozen FBA model for all remaining calls. Eight fresh future streams pair current-budget and continuation conditions at each root and logical call. All 9,520 branch rollouts run to termination.

Before the current decision, branches restore simulator and controller state, the recorded observation and physical counters, executed history, and all three ensemble members' recurrent states. Root

features and predictions must match the reference. The forced current budget executes once, after which the specified continuation controls the remaining calls. In initial roots, the chosen budgets 1/2/4/10 occur 129/78/18/0 times; in noninitial roots they occur 176/39/10/0 times. Population averages include these unchanged-budget roots.

For continuation $\rho \in \{\mathrm{fixed1}, \mathrm{FBA}\}$, we measure

$$d_Y(s,\rho) = \frac{1}{8}\sum_{r=1}^{8}\big[Y_{s,r}(k^*(s),\rho) - Y_{s,r}(1,\rho)\big], \tag{21}$$

and analogously $d_C(s,\rho)$ for remaining cumulative NFE. Here $Y_{s,r}(k,\rho)$ denotes the terminal outcome under future stream r and continuation rule ρ, extending the fixed-budget notation. We first average paired differences over streams within a root, then bootstrap paired references within suite–task strata for 10,000 draws, with equal task and suite weights. Initial and noninitial populations remain separate. The continuation contrast is $d_Y(s,\mathrm{FBA}) - d_Y(s,\mathrm{fixed1})$, with the analogous cost contrast; both use the same roots and paired streams.

Continuation changes the incremental value of the initial choice. At initial roots, choosing the preselected budget instead of NFE 1 increases success by 5.94 pp when both branches subsequently use fixed 1, and by 2.44 pp when both subsequently use FBA. The paired difference between these single-call effects is -3.50 pp $[-5.33, -1.61]$, with a $+0.63$ NFE change in the corresponding cost effect. At noninitial roots, the success effects are $+0.11$ pp under fixed 1 continuation and 0.00 pp under FBA continuation; both choices add remaining NFE on average. Tables 14, 15, and 16 give all success and cost estimates, intervals and suite-level results.

These single-call effects differ from the episode-level effect of After-first: the noninitial roots intervene at one call in states visited by Full, whereas After-first changes choices throughout an episode and generates its own subsequent states.

What varies across repeated executions? We verify the restored root, decision inputs and recurrent states, the first action, paired noise, actual solver counts and native termination. Repeated executions also reveal residual variation beyond the sampled policy noise. In 18 repeated A/A suffix pairs, 16 have identical future action paths and 17 have matching terminal-success labels and summaries. An independent four-process trace found one-intensity-level differences in one or two RGB values despite identical recorded physical/model states, preceding actions, recurrent states, and environment RNG. All original repeats are retained. Accordingly, the future-stream estimates describe the executed pipeline, including policy sampling and residual rendering variation. Intervals condition on the frozen models and summarize root populations. Degenerate bootstrap intervals are omitted.

D.5 Adapting Allocation to Real-Robot Outcomes

Real-robot collection provides outcomes for the budget actually executed at each call, whereas simulation supplies paired outcomes from restored states. We adapt the same success–cost predictor to these observed-budget labels and then evaluate autonomous allocation. The three scenes examine how the resulting choices depend on the manipulation task.

D.5.1 Hardware and Tasks

All scenes use a Franka Emika Panda arm, two wrist-mounted RealSense 405 cameras for close-up views, and a third, external Intel RealSense D435 camera covering the full workspace.

Task success and episode limit. Scene A succeeds when the orange is accurately placed in the bowl without contacting obstacles. Scene B requires accurate cup stacking without cup-wall contact during stacking. Scene C requires accurately removing a cup from the stack. Each evaluation episode is limited to 600 execution steps. Figure 16 shows the five scene-A layouts and representative stacking and unstacking operations. Each B/C example shows one of three evaluated layouts.

Table 14: **Single-call effects with fixed 1 continuation.** At each root, the current call uses either the preselected budget $k^*(s)$ or NFE 1; both branches use NFE 1 afterward. Initial and noninitial populations each contain 225 roots (25 per suite), with eight paired future streams per root. Pointwise 95% reference-bootstrap intervals are shown as $x^{+u}_{-\ell}$, with bounds $[x-\ell, x+u]$; $\pm$ denotes equal lower and upper errors. A dash marks an omitted degenerate interval. Cost differences are in remaining NFE.

	Initial roots		Noninitial roots	
Population	Δsuccess (pp)	ΔNFE	Δsuccess (pp)	ΔNFE
All suites	$+5.94^{+1.67}_{-1.83}$	$-0.44^{+0.33}_{-0.30}$	$+0.11^{+0.72}_{-0.61}$	$+0.31^{+0.14}_{-0.13}$
Dynamic distractor	$-1.00^{+1.00}_{-1.50}$	$+0.55^{+0.25}_{-0.22}$	$+0.50^{+1.00}_{-0.50}$	$+0.04^{+0.11}_{-0.07}$
Static distractor	0.00 (–)	$+0.04^{+0.08}_{-0.04}$	$-0.50^{+0.50}_{-1.00}$	$+0.27^{+0.28}_{-0.21}$
Preposition	$+9.50^{+8.00}_{-11.50}$	$-0.15^{+2.18}_{-1.50}$	$-1.00^{+1.00}_{-1.50}$	$+0.61^{+0.44}_{-0.37}$
Unseen objects	$+5.00^{+6.50}_{-6.00}$	$-0.61^{+1.03}_{-1.10}$	$-0.50^{+0.50}_{-1.00}$	$+0.56^{+0.46}_{-0.40}$
Cautious grasp	$+39.50^{+7.50}_{-8.00}$	$-5.21^{+1.37}_{-1.33}$	-0.50 ± 3.50	$+0.74^{+0.69}_{-0.72}$
Dynamic obstacles	0.00 (–)	$+0.08^{+0.12}_{-0.08}$	$-0.50^{+0.50}_{-1.00}$	$+0.51^{+0.42}_{-0.35}$
Hazard avoidance	-1.50 ± 3.00	$+0.75^{+0.55}_{-0.51}$	$+3.00^{+5.00}_{-3.00}$	$-0.10^{+0.52}_{-0.69}$
State preservation	$+1.00^{+5.50}_{-4.00}$	$+0.42^{+0.69}_{-1.00}$	$+0.50^{+1.00}_{-0.50}$	$-0.00^{+0.08}_{-0.10}$
Static obstacles	$+1.00^{+2.00}_{-1.00}$	$+0.21^{+0.22}_{-0.37}$	0.00 (–)	$+0.19^{+0.17}_{-0.15}$

Table 15: **Single-call effects with FBA continuation.** At each root, the current call uses either the preselected budget $k^*(s)$ or NFE 1; both branches use FBA afterward. The same preselected budget is used in both continuation conditions. Initial and noninitial populations each contain 225 roots (25 per suite), with eight paired future streams per root. Pointwise 95% reference-bootstrap intervals are shown as $x^{+u}_{-\ell}$, with bounds $[x-\ell, x+u]$; $\pm$ denotes equal lower and upper errors. A dash marks an omitted degenerate interval. Cost differences are in remaining NFE.

	Initial roots		Noninitial roots	
Population	Δsuccess (pp)	ΔNFE	Δsuccess (pp)	ΔNFE
All suites	$+2.44^{+1.06}_{-1.00}$	$+0.19 \pm 0.22$	0.00 ± 0.50	$+0.23^{+0.15}_{-0.16}$
Dynamic distractor	$-0.50^{+0.50}_{-1.00}$	$+0.59^{+0.50}_{-0.56}$	0.00 (–)	-0.10 ± 0.10
Static distractor	0.00 (–)	$+0.03^{+0.06}_{-0.03}$	$+1.00^{+2.00}_{-1.00}$	$-0.46^{+0.61}_{-1.01}$
Preposition	$+5.50^{+4.00}_{-4.50}$	$+0.68^{+1.15}_{-1.19}$	-0.50 ± 1.50	$+0.25 \pm 0.42$
Unseen objects	$+2.00^{+3.00}_{-2.00}$	$-0.30^{+0.38}_{-0.50}$	$-0.50^{+0.50}_{-1.00}$	$+0.62^{+0.50}_{-0.45}$
Cautious grasp	$+18.00^{+7.00}_{-6.00}$	$-1.02^{+0.91}_{-1.06}$	$-0.50^{+2.50}_{-2.00}$	$+0.63 \pm 0.57$
Dynamic obstacles	$-0.50^{+0.50}_{-1.00}$	$+0.12^{+0.18}_{-0.12}$	$-0.50^{+0.50}_{-1.00}$	$+0.46^{+0.39}_{-0.34}$
Hazard avoidance	-3.00 ± 3.00	$+1.17^{+0.38}_{-0.36}$	$+0.50 \pm 3.00$	$+0.34^{+0.49}_{-0.47}$
State preservation	$+1.00 \pm 2.50$	$+0.05^{+0.93}_{-0.90}$	$+0.50^{+1.00}_{-0.50}$	$+0.11^{+0.34}_{-0.21}$
Static obstacles	$-0.50^{+0.50}_{-1.00}$	$+0.44^{+0.15}_{-0.14}$	0.00 (–)	$+0.19^{+0.20}_{-0.16}$

D.5.2 Training the Scene-Specific Action Policies

The action policies for all three scenes are initialized from pretrained $\pi_{0.5}$ base and fine-tuned separately. For scenes A and B, we use successful human-in-the-loop trajectories containing both policy-generated controls and human corrective actions; intervention steps account for approximately 62% in both datasets. After excluding seven episodes with incomplete gripper release, A contains 146 episodes with 27,288 frames and B contains 72 episodes with 14,431 frames. The scene-A policy is trained for 10,000 updates, and the scene-B and scene-C policies for 8,000 updates each. All three use batch size 32 and peak learning rate 5×10^{-5}. States and actions use mean-standard-deviation normalization because sparse gripper commands make quantile normalization unstable. The fine-

Table 16: **Change in the single-call effect under adaptive continuation.** Entries give the current-budget effect under FBA continuation minus that under fixed-1 continuation. Each effect compares the same preselected budget $k^*(s)$ with NFE 1 at the current call. Initial and noninitial populations each contain 225 roots (25 per suite), with eight future streams per root. Roots and streams are paired across continuation conditions. Pointwise 95% reference-bootstrap intervals are computed directly for this difference in effects. The notation $x^{+u}_{-\ell}$ gives bounds $[x-\ell, x+u]$; $\pm$ denotes equal lower and upper errors, and a dash marks an omitted degenerate interval. Cost entries give the corresponding difference in remaining-NFE effects.

	Initial roots		Noninitial roots	
Population	Δsuccess (pp)	ΔNFE	Δsuccess (pp)	ΔNFE
All suites	$-3.50^{+1.89}_{-1.83}$	$+0.63 \pm 0.37$	$-0.11^{+0.55}_{-0.56}$	$-0.08^{+0.13}_{-0.15}$
Dynamic distractor	$+0.50 \pm 1.50$	$+0.03^{+0.59}_{-0.69}$	$-0.50^{+0.50}_{-1.00}$	$-0.14^{+0.14}_{-0.19}$
Static distractor	0.00 (–)	$-0.01^{+0.01}_{-0.02}$	$+1.50^{+2.00}_{-1.50}$	$-0.72^{+0.74}_{-1.04}$
Preposition	$-4.00^{+12.00}_{-9.00}$	$+0.83^{+2.01}_{-2.30}$	$+0.50^{+2.00}_{-1.50}$	$-0.36^{+0.52}_{-0.65}$
Unseen objects	$-3.00^{+4.00}_{-5.00}$	$+0.31^{+1.10}_{-0.89}$	0.00 (–)	$+0.07^{+0.09}_{-0.08}$
Cautious grasp	$-21.50^{+10.50}_{-9.50}$	$+4.19^{+1.58}_{-1.78}$	$0.00^{+2.50}_{-3.50}$	$-0.10^{+0.59}_{-0.47}$
Dynamic obstacles	$-0.50^{+0.50}_{-1.00}$	$+0.04^{+0.06}_{-0.04}$	0.00 (–)	$-0.05^{+0.10}_{-0.13}$
Hazard avoidance	$-1.50^{+2.50}_{-3.00}$	$+0.42^{+0.43}_{-0.42}$	-2.50 ± 3.00	$+0.43^{+0.35}_{-0.33}$
State preservation	$0.00^{+5.00}_{-6.00}$	$-0.36^{+1.44}_{-1.33}$	0.00 (–)	$+0.12^{+0.25}_{-0.13}$
Static obstacles	$-1.50^{+1.50}_{-2.00}$	$+0.22^{+0.27}_{-0.21}$	0.00 (–)	$0.00^{+0.06}_{-0.05}$

tuned action policies remain fixed during budget experiments. These preparation datasets are distinct from the subsequent evaluation trials and from the budget-intervention collection described next.

D.5.3 Human-Guided Budget Collection and Offline Training

An operator selects moments to increase or decrease NFE while the robot executes the frozen action policy. These interventions collect outcomes under different sampling budgets at task-relevant moments. We initialize from the simulation-trained FBA ensemble and fine-tune all three members on each scene's collected real-robot rollouts, yielding one adapted ensemble per scene. A trajectory can contain several budgets, so the training unit is an individual policy call with its actually executed budget, rather than an episode assigned a single budget.

Collection conditions and intervention points. At each layout–placement setting in each scene, we collect two rollouts for each fixed budget in $\{1, 2, 4, 10\}$. We then collect budget-intervention rollouts using two background budgets: from NFE 1, we increase to each of $\{2, 4, 10\}$; from NFE 10, we decrease to each of $\{1, 2, 4\}$. Each destination budget has two rollout repetitions. The operator visually selects the intervention point during execution. In scenes A and C, intervention points are restricted to grasping; in scene B, they are restricted to placement. The destination budget applies for exactly one policy call, then execution resumes the background budget.

Collection uses the same layout–placement configurations as evaluation: five layouts with five placements each in A, and three layouts with five placements each in B and C. Each configuration contributes $4 \times 2 = 8$ fixed-budget and $2 \times 3 \times 2 = 12$ intervention rollouts, for 20 in total: 500 rollouts for A and 300 each for B and C, or 1,100 overall. Each scene is adapted on its own data.

Learning from each executed budget. Recorded policy features x_t and executed budgets are kept in episode order. For an episode with T calls, let $\mathcal{H}_t$ contain the features and history available at the budget decision for call t, let $k_t \in \mathcal{K}$ be its executed NFE, and let $y \in \{0, 1\}$ denote terminal success. The remaining-cost target includes the current call and every later executed call:

$$C_t^{\mathrm{H}} = \sum_{j=t}^{T-1} k_j. \tag{22}$$

For each scene's adaptation run, $\mathcal{D}_{\mathrm{H}}$ pools the tuples $(\mathcal{H}_t, k_t, y, C_t^{\mathrm{H}})$ across its training episodes; call indices are local to each episode. Success is shared within an episode, while remaining NFE decreases as execution proceeds. Future outcomes and budget choices enter only the labels, not the input history. Budget labels refer to executed calls; a requested switch must align with its first effective call.

Adapting the outcome predictor. At each call, we fit terminal success and remaining NFE only for the executed budget, averaging losses first within episodes and then across episodes. At each call, candidate i satisfies $k_i = k_t$. Let $s_{\mathrm{H}} > 0$ be the physical cost-normalization scale. With $\widehat{c}_{t,i} = \widehat{C}_{t,i}/s_{\mathrm{H}}$ and $c_t^{\mathrm{H}} = C_t^{\mathrm{H}}/s_{\mathrm{H}}$, define

$$\ell_{\mathrm{H},t}^{(e)} = \ell_{\mathrm{BCE}}(p_{t,i}, y_e) + \alpha_{\mathrm{H}}(\widehat{c}_{t,i} - c_t^{\mathrm{H}})^2,$$
$$L_{\mathrm{H}} = \frac{1}{|\mathcal{B}|} \sum_{e \in \mathcal{B}} \frac{1}{T(e)} \sum_{t=0}^{T(e)-1} \ell_{\mathrm{H},t}^{(e)}. \tag{23}$$

Here $\mathcal{B}$ is an episode minibatch, $T(e)$ is the length of episode e, and $\alpha_{\mathrm{H}} > 0$ weights the cost loss. The selected index i is local to each call; episode indices are suppressed on predictions and cost targets. There is no within-state counterfactual pairwise term or imitation loss on the operator's chosen budget. In each scene's adaptation run, we fine-tune all FBA parameters for 10 epochs using Adam with learning rate 3×10^{-4}, a batch size of eight episodes and gradient-norm clipping at 1.0, while keeping the action policy frozen. We set the cost-normalization scale to $s_{\mathrm{H}} = 100$ NFE and the cost-loss weight to $\alpha_{\mathrm{H}} = 1$. Episodes are replayed in order with recurrent states reset between them. For N_{H} episodes, retaining the last incomplete minibatch gives $10\lceil N_{\mathrm{H}}/8 \rceil$ updates: 630 for A's 500 episodes and 380 each for B and C's 300 episodes.

What the physical labels predict. Each label describes the terminal result and remaining NFE under the budget sequence that the robot actually executed. In intervention rollouts, labels before the intervention point include the later one-call budget change; labels at the intervention point include the subsequent fixed-background continuation. Simulation targets $Y_s(k, c; W)$ and $C_s(k, c; W)$ instead use matched restored roots, a single-call window $W = [t, t+1)$ and a specified continuation budget c. For binary cross-entropy and squared cost loss, the unrestricted population optima are the conditional success probability and mean remaining cost in the collection distribution. The visually selected intervention point determines where the intervention occurs and can use information beyond the recorded history $\mathcal{H}_t$.

D.5.4 Autonomous Evaluation and Call Latency

Comparing methods within each scene. All five methods share the scene's frozen action policy. Fixed budgets use NFE 1, 2, 4 or 10; allocation uses that scene's adapted three-member ensemble with $\lambda = 2$, proprioception and preceding-call history, and padding for missing inputs. Models remain frozen across the scene's layouts and placements, with no human budget changes during evaluation.

Each method runs twice at five placements per layout: $5 \times 5 \times 2 = 50$ trials in A and $3 \times 5 \times 2 = 30$ each in B and C. Across five methods: 250/150/150 trials by scene, or 550 in total. Collection and evaluation share layouts and placements but use disjoint, separately executed rollouts.

Measured call latency. Table 2 reports median end-to-end latency of actual evaluation calls after JAX warmup, including the full policy and FBA computation where applicable. Each method's median pools A/B/C calls; latency measures one invocation rather than episode duration.

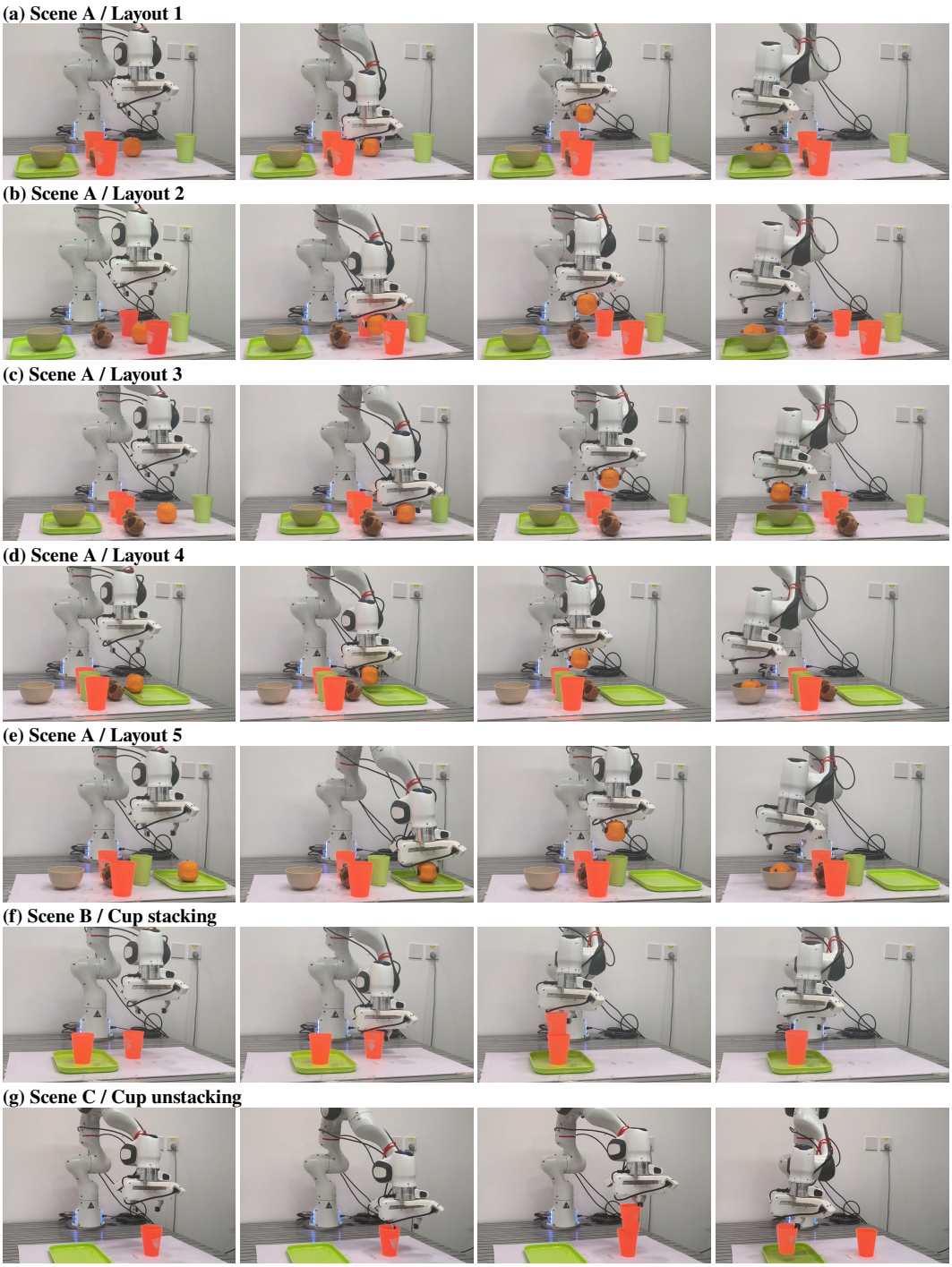


Figure 16: **Real-robot layouts and representative executions.** Rows (a–e) show the five orange-placement layouts in scene A; rows (f–g) show representative cup-stacking and cup-unstacking layouts in B/C. Each row contains four selected frames in temporal order. Success rates use the full evaluation grid.