PROPRIOCEPTIVE SKETCHES AS LONG-HORIZON INTENT FOR GENERATIVE ACTION POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative robot policies predict short action chunks but lack explicit long-horizon intent. Recent methods expose longer-horizon structure through language plans, subgoal images, or video forecasts, which are costly to generate and still need to be translated into robot motion. Predicting future robot motions avoids this translation, but a dense, time-indexed trajectory requires numerous parameters to cover the full remaining task, and over a short horizon it largely repeats the action chunk and adds little guidance for action generation. We propose Proprioceptive Action Models (PAM), which jointly generate a compact, timing-free sketch of the robot's remaining joint-space path and a dense executable action chunk within a single transformer denoiser. The sketch parameterizes the path by arc length rather than time, capturing geometric intent invariant to execution timing. Block-causal attention and a staggered denoising schedule maintain directed sketch-to-action dependence, ensuring the action tokens condition on a progressively cleaner sketch throughout sampling. In simulation, PAM improves over its action-only counterparts on Push-T and LIBERO-Long; on four real-world bimanual tasks, it raises success from 47.5% to 75.0%.

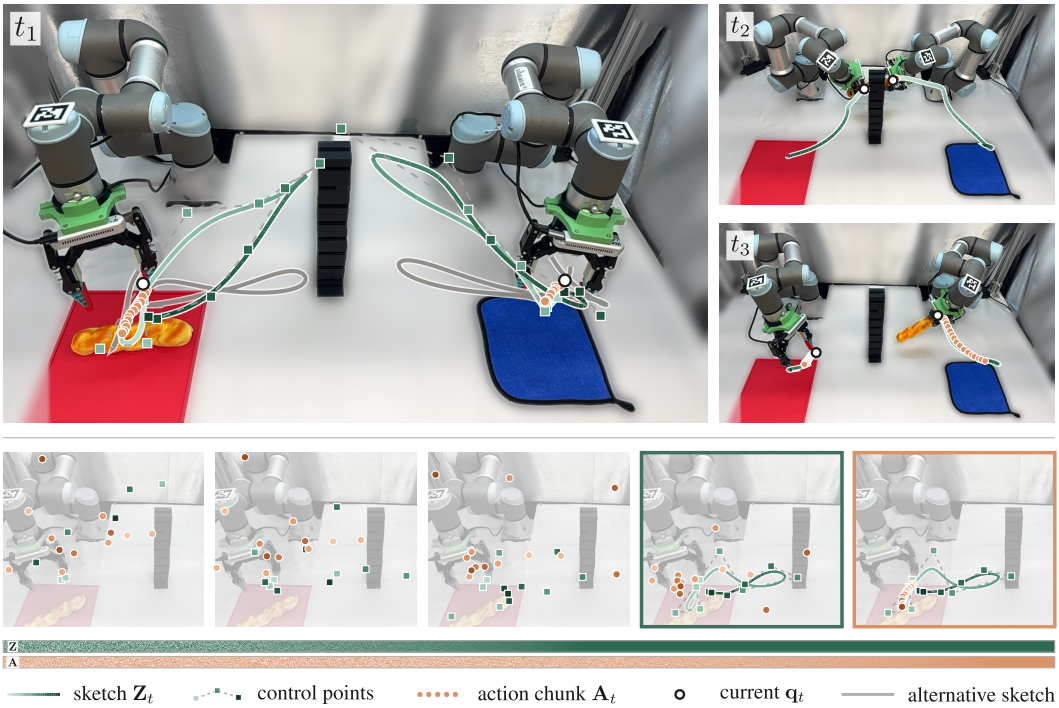


Figure 1: **PAM replans during a real-world dual-arm long-horizon task.** At each step, PAM jointly generates a proprioceptive sketch $\mathbf{Z}_t$ of the future configuration-space path and a short action chunk $\mathbf{A}_t$ conditioned on the sketch, while only $\mathbf{A}_t$ is executed. For visualization, the sketch and the action chunk are mapped through forward kinematics and projected into the camera view.

1 INTRODUCTION

Generative action policies have become a dominant paradigm for visuomotor robot control, producing temporally coherent commands through action chunking (Zhao et al., 2023; Chi et al., 2023; Black et al., 2025b; Reuss et al., 2025). Despite strong performance on short-horizon tasks, these policies remain less effective on long-horizon manipulation (Fan et al., 2025). One approach to this challenge is to guide action prediction with visual forecasts (Black et al., 2024; Hu et al., 2025; Tian et al., 2025), predictive scene latents (Zhang et al., 2026; Liu et al., 2026; Zhu et al., 2025), or language-based plans and reasoning (Wen et al., 2025; Zawalski et al., 2025). These representations capture anticipated scene evolution or semantic task structure, but can be costly to generate or require additional learned mappings to continuous control. This motivates a compact representation of long-horizon intent closely tied to the robot's own motion.

Future robot configurations provide a natural representation of the robot's intended motion. Demonstrations contain synchronized proprioceptive measurements, providing supervision without additional annotation in a space directly related to robot control. Prior work has used robot-motion representations to guide action prediction, including image-plane trajectories (Gu et al., 2024; Niu et al., 2025) and predicted end-effector paths (Chen et al., 2026). However, a dense, time-indexed configuration trajectory over the full remaining task is long and varies in length across demonstrations, making it difficult to represent with a limited number of parameters. Restricting the prediction to a short horizon instead makes it largely redundant with the executable action chunk, so it provides little additional guidance for action generation. An effective intent representation should therefore cover the full remaining motion with a fixed number of parameters while providing information beyond the action chunk.

We introduce the *proprioceptive sketch*, a compact representation of the robot's remaining configuration-space path to task completion. The sketch uses arc-length parameterization to remove execution timing and a small number of B-spline coefficients to limit geometric details. In the handoff task in Figure 1, the sketch describes the arm paths toward the transfer region and around the barrier without specifying motion speed or the duration of the exchange. It provides long-horizon geometric context for predicting a separate, short-horizon action chunk. Only the action chunk is executed, determining the immediate motion and gripper commands.

Proprioceptive Action Models (PAM) jointly generate the sketch and an action chunk within a single transformer denoiser. As illustrated in Figure 1, both outputs are initialized with Gaussian noise and refined through iterative denoising. A block-causal attention mask allows action tokens to attend to the sketch while preventing sketch tokens from reading the action stream. Following Diffusion Forcing (Chen et al., 2024), we train the denoiser with varying corruption levels across the two streams and use a staggered denoising schedule that denoises the sketch ahead of the actions, providing cleaner geometric context for action generation.

Our contributions are threefold. First, we formulate the proprioceptive sketch, a compact, timing-free representation of the remaining configuration-space path that serves as an explicit intermediate variable for action generation and is, by construction, invariant to temporal reparameterization of the demonstrated path. Second, we develop a joint generator with directed attention and staggered denoising to guide action prediction with the sketch. Third, we apply PAM to a diffusion transformer trained from scratch (PAM-DP), and a pretrained vision-language-action model (PAM-VLA). We evaluate PAM-DP on Push-T and four real-world bimanual tasks, and PAM-VLA on LIBERO-Long.

2 RELATED WORK

Predictive intermediate representations. Intermediate predictions provide action policies with information about intended behavior. Language-based approaches express task structure through subtask descriptions or reasoning traces (Wen et al., 2025; Black et al., 2025a; Zawalski et al., 2025). Visual approaches use future images or video to represent goal configurations and scene evolution (Black et al., 2024; Zhao et al., 2025; Hu et al., 2025; Tian et al., 2025). Other methods learn predictive representations of scene dynamics or visuomotor behavior in latent space (Zhang et al., 2026; Liu et al., 2026; Liang et al., 2024; Xu et al., 2026b). These representations describe the task rather than the robot's own motion, and they are either expensive to generate or require a learned

decoder to reach continuous control. PAM instead predicts intended motion as a configuration-space path, which requires no annotation beyond recorded proprioception.

Robot-motion representations. Robot motion provides an intermediate representation for planning and control. Hierarchical reinforcement learning uses state-space subgoals to guide lower-level policies (Nachum et al., 2018). RT-Trajectory conditions action prediction on image-plane trajectory sketches (Gu et al., 2024), while TraceVLA supplies visual traces of past motion (Zheng et al., 2025). Predictive approaches use future motion in different ways. GeoPredict supervises future keypoint prediction during training (Qian et al., 2026), whereas OASIS predicts end-effector trajectories that condition a separate action decoder (Chen et al., 2026). Recent spline policies similarly parameterize executable action trajectories with spline coefficients (Tian et al., 2026; Han et al., 2026; Yang et al., 2026). PAM likewise predicts future robot motion but differs in three respects: the sketch spans the remaining path to task completion, it removes execution timing through arc-length parameterization, and it is generated jointly with actions in a single denoiser.

Joint generation and denoising schedules. Joint world-action models generate actions together with predicted images, video, or scene representations (Zhu et al., 2025; Ye et al., 2026; Lin et al., 2026; Xu et al., 2026a). Trajectory-diffusion methods instead generate state-action plans (Janner et al., 2022) from which actions are recovered through inverse dynamics (Ajay et al., 2023). Diffusion Forcing introduces independent token-level corruption during training, enabling flexible denoising schedules at inference (Chen et al., 2024). PAM draws on these joint-generation and scheduling principles to couple a non-executable motion sketch with an executable action chunk. Directed attention allows actions to depend on the sketch without feeding action information back into sketch generation, while staggered denoising lets the sketch become cleaner ahead of the actions.

3 METHOD

3.1 PROBLEM SETUP

We consider demonstration trajectories $\tau = \big(\ell, \{(o_t, \mathbf{q}_t, \mathbf{a}_t)\}_{t=1}^{T}\big)$, where ℓ is an optional task specification and T is the trajectory length. At time t, o_t denotes the policy observation, $\mathbf{q}_t \in \mathbb{R}^{d_q}$ is the robot configuration, and $\mathbf{a}_t \in \mathbb{R}^{d_a}$ is the action. The short-horizon action target is $\mathbf{A}_t = [\mathbf{a}_t, \ldots, \mathbf{a}_{t+H-1}] \in \mathbb{R}^{H \times d_a}$, where H is the prediction horizon. We additionally introduce a proprioceptive sketch $\mathbf{Z}_t$ of the remaining configuration path $\mathbf{q}_{t:T}$. PAM models the joint policy

$$(\mathbf{Z}_t, \mathbf{A}_t) \sim \pi_\theta(\cdot \mid o_t, \mathbf{q}_t, \ell). \tag{1}$$

Denoising generative policies. We instantiate π_θ as a conditional denoising generative model. For a clean normalized token $\mathbf{x}^0$ and a noise coordinate $\gamma \in [0, 1]$, with $\gamma = 0$ denoting clean data and $\gamma = 1$ pure noise, the corrupted token is

$$\mathbf{x}^\gamma = \alpha(\gamma)\mathbf{x}^0 + \sigma(\gamma)\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2}$$

where the schedule (α, σ) is set by the generative formulation. A denoiser f_θ takes corrupted tokens, their noise coordinates, and the conditioning $\mathbf{c}_t = (o_t, \mathbf{q}_t, \ell)$, and predicts a regression target $\mathbf{u}$: the noise $\boldsymbol{\epsilon}$ in diffusion models (Ho et al., 2020) or the velocity $\boldsymbol{\epsilon} - \mathbf{x}^0$ in rectified flow (Liu et al., 2023b). Sampling starts from Gaussian noise and applies K reverse updates, such as DDIM (Song et al., 2021) or Euler steps, that move γ from one to zero. PAM applies this procedure jointly to two token streams, the sketch $\mathbf{Z}_t$ (Section 3.2) and the action chunk $\mathbf{A}_t$, with a single transformer denoiser (Section 3.3).

3.2 THE PROPRIOCEPTIVE SKETCH

The proprioceptive sketch represents the geometry of the remaining configuration path without specifying its execution timing (Figure 2). We parameterize the path by arc length and encode it with several B-spline coefficients.

Phase. For a remaining path $\mathbf{q}_{t:T}$, we define cumulative arc length and normalized phase as

$$s_u = \sum_{v=t}^{u-1} \|\mathbf{q}_{v+1} - \mathbf{q}_v\|_2, \qquad \phi_u = \frac{s_u}{L}, \tag{3}$$

for $u \in \{t, \dots, T\}$ and $L > 0$. Here, s_u is the cumulative configuration-space distance from time t to u, and $L = s_T$ is the total length of the remaining path. Phase measures the fraction of path length traversed rather than elapsed time. Stationary intervals therefore produce no phase increment. When $L = 0$, we define the sketch to be zero.

Representation. We encode the phase-indexed path with a clamped cubic B-spline using $M \geq 4$ control points and uniformly spaced interior knots. The sketch is the coefficient matrix $\mathbf{Z}_t = [\mathbf{z}_{t,0}, \dots, \mathbf{z}_{t,M-1}]^\top \in \mathbb{R}^{M \times d_q}$, which decodes to a relative configuration path

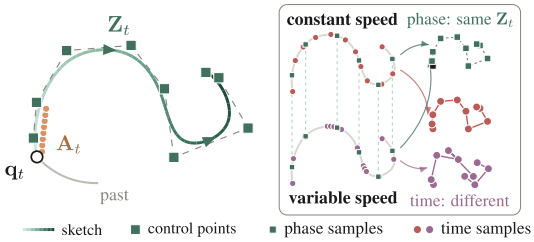


Figure 2: The proprioceptive sketch. Left, B-spline coefficients $\mathbf{Z}_t$ represent the remaining configuration path, while only the action chunk $\mathbf{A}_t$ is executed. Right, re-timing the same continuous path changes its time-indexed representation but preserves its phase-indexed sketch.

$$\Delta\hat{\mathbf{q}}_t(\phi) = \sum_{m=0}^{M-1} B_{m,3}(\phi)\, \mathbf{z}_{t,m}, \phi \in [0, 1]. \tag{4}$$

where $B_{m,3}$ denotes a cubic B-spline basis function. The corresponding absolute path is $\hat{\mathbf{q}}_t(\phi) = \mathbf{q}_t + \Delta\hat{\mathbf{q}}_t(\phi)$. We fix $\mathbf{z}_{t,0} = \mathbf{0}$ so that the path begins at the current configuration. PAM generates only the remaining $M - 1$ coefficients regardless of the length of the remaining path.

Training target and representation capacity. We construct sketch targets by fitting the remaining configuration path in each demonstration. We first express the path relative to $\mathbf{q}_t$ and resample it uniformly in arc-length phase. We then fit the B-spline by least squares, fixing the first and last control points to $\mathbf{0}$ and $\mathbf{q}_T - \mathbf{q}_t$, respectively. All control points except the fixed zero anchor form the $M - 1$ prediction target. The number of control points M controls the trade-off between compactness and geometric fidelity. We use the RMS per-step configuration change in the training demonstrations as a reference for reconstruction accuracy. The adopted coefficient budgets keep the spline reconstruction RMSE below this reference for each task. Appendix A details the knot construction, fitting procedure, normalization, and capacity audit.

Temporal invariance. Changing the speed of traversal along a fixed continuous path does not change its arc-length parameterization. The sketch therefore remains unchanged under temporal reparameterization, including pauses, for any fixed coefficient budget M. This property concerns the underlying continuous path; finite sampling and measurement noise can introduce differences between separately recorded demonstrations.

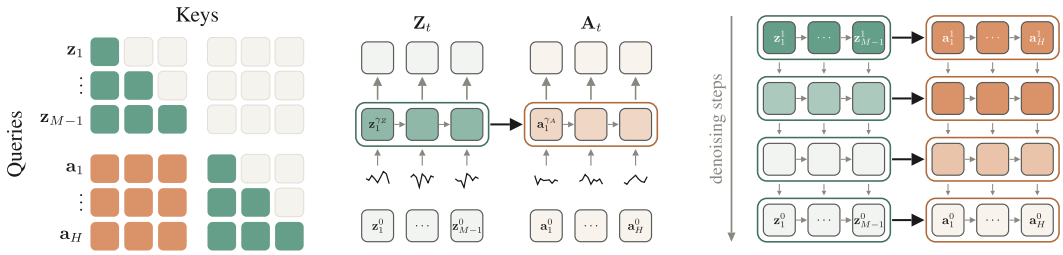
3.3 Joint Sketch and Action Generation

PAM jointly generates the sketch and action chunk with a single transformer denoiser f_θ (Peebles & Xie, 2023). The $M - 1$ predicted sketch control points and H action steps form two token streams, with separate input projections into a shared embedding space. The denoiser is conditioned on o_t, $\mathbf{q}_t$, the optional task specification ℓ, and the noise level of each stream. As shown in Figure 3a, a block-causal attention mask establishes directed sketch-to-action information flow. Sketch token $m \in \{1, \dots, M-1\}$ attends only to sketch tokens with indices at most m. Action token $j \in \{1, \dots, H\}$ attends to all sketch tokens and to action tokens with indices at most j. The sketch ordering follows the order of the B-spline basis supports along phase, while the action ordering follows execution time.

Training with independent noise levels. Following Diffusion Forcing (Chen et al., 2024), we sample the sketch and action noise coordinates γ_Z and γ_A independently and uniformly for each example; all tokens within a stream share its coordinate, while the Gaussian noise is sampled independently for each token (Figure 3b). We optimize

$$\mathcal{L}_{\text{PAM}} = \sum_{s \in \{Z, A\}} \lambda_s\, \mathbb{E}\left\| \mathbf{U}_s - f_\theta^s(\mathbf{Z}_t^{\gamma_Z}, \mathbf{A}_t^{\gamma_A}, \gamma_Z, \gamma_A, \mathbf{c}_t) \right\|^2, \tag{5}$$

where $\mathbf{U}_s$ stacks the per-token targets of stream s, f_θ^s is the corresponding output head, the squared error is averaged over tokens and feature dimensions, and λ_Z and λ_A are loss weights.



(a) Block-causal attention (b) Stream-wise training (c) Staggered denoising

Figure 3: Joint sketch and action generation. (a) Block-causal attention allows actions to read the sketch while preventing information flow in the reverse direction. (b) Training assigns independent noise levels to the sketch and action streams, with a shared level within each stream. (c) Staggered denoising advances the sketch ahead of the action chunk.

Staggered denoising. At inference, we initialize both streams with Gaussian noise and use a schedule that denoises the sketch ahead of the actions. We use a sketch lead $\eta \in [0, 1]$ and assign the stream-wise schedules

$$\gamma_Z = \mathrm{clip}(b - \eta, 0, 1), \quad \gamma_A = \mathrm{clip}(b, 0, 1), \quad (6)$$

where b is a shared scheduling variable that starts at $1 + \eta$ and decreases by $1/K$ until both coordinates reach zero, with K the number of base sampling steps (Algorithm 1). A positive lead advances sketch denoising relative to action denoising, providing cleaner sketch inputs during action generation (Figure 3c), while $\eta = 0$ assigns the same noise level to both streams at every step. At each step, f_θ reads both streams at their current noise levels, and UPDATE applies one reverse step of the base sampler (DDPM, DDIM, or Euler) to each stream from its current to its next noise coordinate; it is the identity when the coordinate is unchanged. The sampler uses $K + \lceil \eta K \rceil$ NFE. When ηK is an integer, the first ηK iterations update only the sketch and the last ηK update only the actions. In this case, each stream receives exactly K updates.

Algorithm 1 Staggered denoising

Input: $o_t, \mathbf{q}_t, \ell$; lead η; base steps K
$\mathbf{Z}, \mathbf{A} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $k = 0, \ldots, K + \lceil \eta K \rceil - 1$ **do**
  $b \leftarrow 1 + \eta - k/K$; $\gamma_Z, \gamma_A \leftarrow$ Eq. (6)
  $(\mathbf{u}_Z, \mathbf{u}_A) \leftarrow f_\theta(\mathbf{Z}, \mathbf{A}, o_t, \mathbf{q}_t, \ell, \gamma_Z, \gamma_A)$
  $\mathbf{Z} \leftarrow \textsc{Update}(\mathbf{Z}, \mathbf{u}_Z, \gamma_Z)$
  $\mathbf{A} \leftarrow \textsc{Update}(\mathbf{A}, \mathbf{u}_A, \gamma_A)$
end for
Output: $\mathbf{A}$ for execution; $\mathbf{Z}$ for inspection only

4 EXPERIMENTS

Our experiments address three questions. **(Q1)** Does PAM improve task performance over action-only policies in simulation and on real robots? **(Q2)** How do the sketch representation, directed attention, and denoising schedule contribute to performance? **(Q3)** To what extent does action generation depend on the predicted sketch?

4.1 SETUP

Tasks. We evaluate PAM on two simulation benchmarks and four real-world bimanual tasks, shown in Figure 4. Push-T (Chi et al., 2023) requires pushing a T-shaped block to a target pose and supports evaluation with both state and image observations. LIBERO-Long (Liu et al., 2023a) comprises ten language-conditioned, long-horizon manipulation tasks on a simulated Franka arm. Our real-world tasks are *handoff*, *close lid*, *fold cloth*, and *measure*, performed on a dual-UR3 platform.

Policies and baselines. We evaluate PAM-DP on Push-T and the real-world tasks, and PAM-VLA on LIBERO-Long. PAM-DP is trained from scratch using diffusion, with transformer-based Diffusion Policy (Chi et al., 2023) as its action-only baseline. PAM-VLA adds the sketch to the rectified-flow action expert of FLOWER (Reuss et al., 2025), which serves as its pretrained action-only baseline. We also compare with B-spline Policy (Han et al., 2026), which represents executable actions as spline curves. We compare with the transformer-based Diffusion Policy rather than its convolutional variant, because PAM represents the sketch and actions as token streams in a transformer.

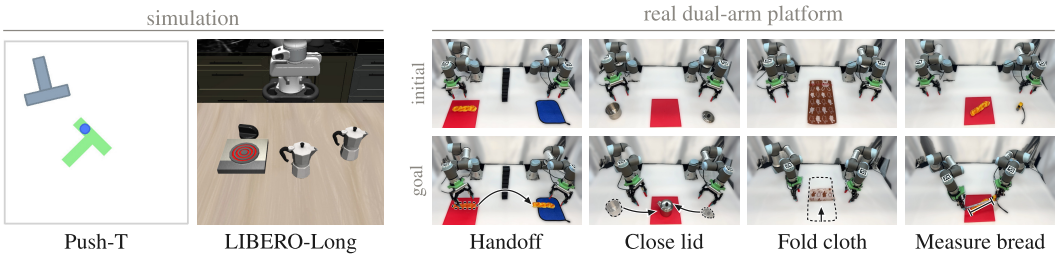


Figure 4: Overview of the simulation and real-world tasks. Push-T and LIBERO-Long in simulation, and four tasks on a real dual-arm platform shown at the initial state (top) and at the goal state (bottom). The Push-T goal is the green target pose; each LIBERO-Long task is specified by a language instruction (shown: *put both moka pots on the stove*).

Table 1: Push-T coverage with state and image observations. We report the maximum and the mean over the last 10 checkpoints.

Method	State	Image
LSTM-GMM (Mandlekar et al., 2022)	0.67 / 0.61	0.69 / 0.54
IBC (Florence et al., 2022)	0.90 / 0.84	0.75 / 0.64
Diffusion Policy (Chi et al., 2023)	0.95 / 0.79	0.78 / 0.66
B-spline Policy (Han et al., 2026)	0.92 / 0.74	0.76 / 0.63
PAM-DP (ours)	**0.96 / 0.88**	**0.82 / 0.72**

Table 2: LIBERO-Long success rate. PAM-VLA results report mean $\pm$ standard deviation over 5 training seeds.

Method	Success (%)
π_0 (Black et al., 2025b)	85.2
$\pi_{0.5}$-KI (Driess et al., 2025)	85.8
OpenVLA-OFT (Kim et al., 2025)	94.5
FLOWER (Reuss et al., 2025)	94.9 $\pm$ 1.2
PAM-VLA (ours)	**95.2 $\pm$ 0.4**

Implementation. Both PAM-DP and PAM-VLA independently sample sketch and action noise levels, with a shared level within each stream. We use a sketch lead of $\eta = 1/8$ across all tasks unless otherwise specified. We use $M = 12$ control points for Push-T, $M = 16$ for LIBERO-Long, and $M = 12$, 14, 16, and 14 for handoff, close lid, fold cloth, and measure, respectively. In the real-world tasks, the successful evaluation episodes span roughly 250 to 370 control steps on average. These capacities satisfy the reconstruction criterion audited in Appendix A. Appendix B describes the model implementation, and Appendix C summarizes training and evaluation settings.

4.2 Simulation

Push-T. We evaluate PAM-DP with state and image observations using the dataset and evaluation protocol of Chi et al. (2023). The configuration $\mathbf{q}_t$ is the measured two-dimensional pusher position, and the sketch represents its remaining path in both observation settings. The state-based setting allows us to examine the sketch without the additional challenge of visual perception. We measure performance by target coverage, the fraction of the target region covered by the block. Table 1 reports the maximum over checkpoints and the mean over the last ten checkpoints, following Chi et al. (2023), with 3 seeds and 50 initial conditions per evaluation. PAM-DP achieves higher coverage than the action-only transformer and B-spline Policy in both observation settings. Compared with Diffusion Policy, the last-ten checkpoint average increases from 0.79 to 0.88 with state observations and from 0.66 to 0.72 with image observations. The corresponding maximum scores increase from 0.95 to 0.96 and from 0.78 to 0.82. The gains in the checkpoint average indicate that the differences are not limited to the best checkpoint. B-spline Policy, which uses a spline to parameterize executable actions, does not improve over the dense transformer baseline in this setting, whereas using the spline to represent the remaining path alongside dense actions does.

LIBERO-Long. We evaluate PAM-VLA on the ten language-conditioned tasks of LIBERO-Long following the protocol of Reuss et al. (2025), with 50 trials per task, and report mean success rate and standard deviation over 5 training seeds. FLOWER also serves as the direct action-only baseline. Table 2 shows that PAM-VLA increases mean success rate from FLOWER's 94.9% to 95.2% under the same evaluation protocol, with a standard deviation of 0.4 percentage points across five training seeds. Since FLOWER already approaches the performance ceiling on LIBERO-Long, the remaining margin for improvement is small.

Table 3: Real-world task performance. Success is reported as completed episodes out of 20 trials per task. Completion time is measured in seconds over successful episodes.

Method	Metric	Handoff	Close lid	Fold cloth	Measure	Overall
Diffusion Policy	Success	9/20	9/20	10/20	10/20	47.5%
	Time (s)	36.2 ± 4.6	44.6 ± 9.2	41.9 ± 1.3	42.8 ± 3.6	41.4
B-spline Policy	Success	8/20	8/20	10/20	9/20	43.8%
	Time (s)	39.9 ± 7.2	42.2 ± 11.9	42.0 ± 1.2	40.6 ± 5.3	41.2
PAM-DP (ours)	Success	**13/20**	**14/20**	**18/20**	**15/20**	**75.0%**
	Time (s)	35.8 ± 8.0	45.3 ± 11.1	42.4 ± 2.4	45.2 ± 6.4	42.3

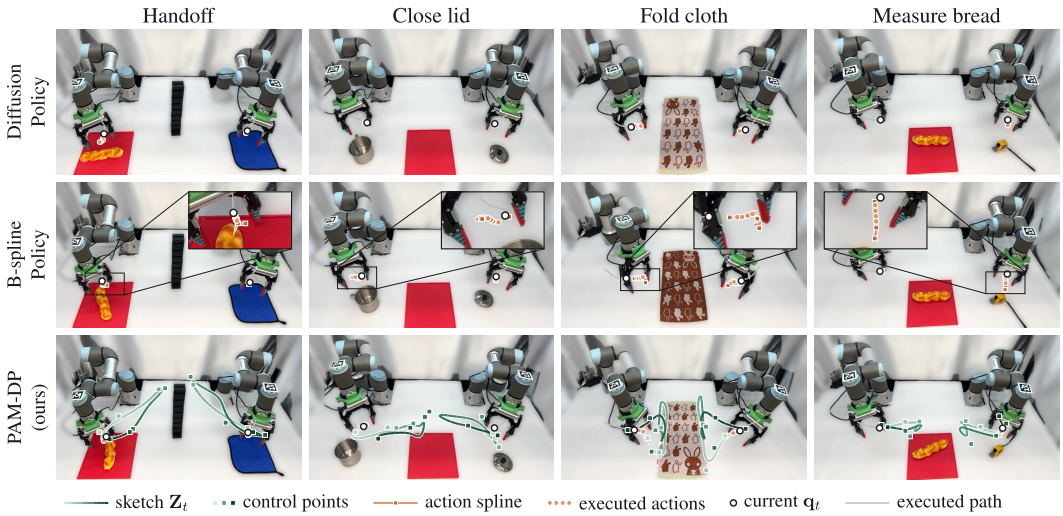


Figure 5: Policy predictions on the four real-world tasks, projected into the external camera view. For B-spline Policy, orange squares and curves show the control points and the action spline.

4.3 Real-World Dual-Arm Manipulation

We evaluate PAM-DP on a dual-UR3 platform equipped with parallel grippers and three RGB cameras. The policy observes the camera images and robot joint positions, and outputs 14-dimensional joint and gripper commands at 8 Hz. We collect 81, 81, 48, 89 demonstrations for handoff, close lid, fold cloth, and measure, respectively, with randomized initial object placements. In *handoff*, the arms transfer a loaf of bread from the red board to the blue mat while clearing a barrier. In *close lid*, one arm places a pot on the red plate while the other aligns and places its lid on the pot. The *fold cloth* task requires coordinated corner grasps and successive folds, while *measure* requires the arms to position and hold a tape measure along a loaf. We train and evaluate PAM-DP, transformer-based Diffusion Policy, and B-spline Policy on these tasks. Each method is evaluated over 20 episodes per task, with initial object placements sampled from the same distribution.

Task performance. Table 3 reports success rates and completion times. PAM-DP succeeds in 60 of 80 episodes (75.0%), compared with 38 (47.5%) for Diffusion Policy and 35 (43.8%) for B-spline Policy (two-sided Fisher exact test, $p < 0.001$ for both). The improvement holds on all four tasks despite their different objects and coordination demands. B-spline Policy does not exceed Diffusion Policy on any task, suggesting that the gain comes from representing the remaining path rather than from spline parameterization itself. PAM-DP takes slightly longer on successful episodes, consistent with the additional NFE from the sketch lead and the extra sketch tokens processed at each evaluation.

Predicted motion. Figure 5 visualizes policy predictions early in task execution. The recording camera is separate from the three policy cameras and is used only for visualization. We map predicted joint configurations to tool positions through forward kinematics and project them into the

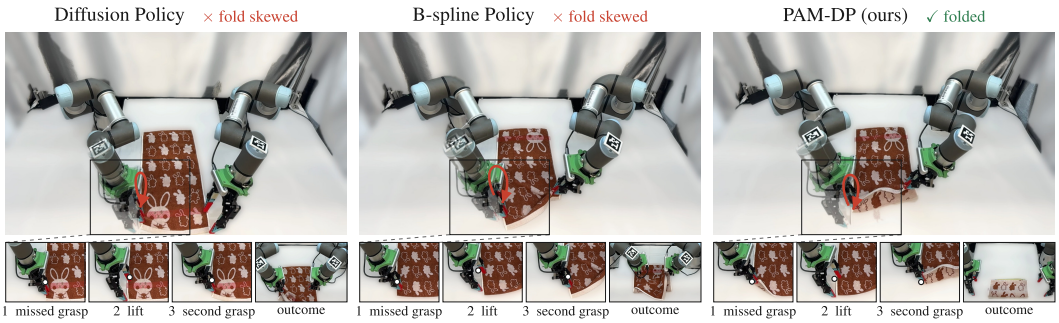


Figure 6: Grasp retries on *fold cloth*. Each column shows an episode in which the first grasp missed. The large image shows the scene after the second grasp, with a translucent overlay marking the gripper at the top of its lift and an arrow indicating the lift and re-descent.

recording view. Diffusion Policy directly generates a short-horizon action chunk, while B-spline Policy predicts control points that define an executable spline from which the actions are sampled. The action predictions describe upcoming commands, while PAM-DP's sketch extends over the remaining motion. In *handoff*, for example, the sketch paths extend toward the transfer region near the barrier, whereas the action chunks remain near the current gripper positions.

Failure analysis. Table 4 groups failed episodes into missed grasps, object loss after grasping, and coordination errors between the arms. Coordination errors include transfer misalignment and collisions with the barrier. Each failed episode receives one label corresponding to the error that directly caused task failure. When errors occur in sequence, we record the earliest error that led to failure. Errors recovered within an ultimately successful episode are not included in these counts.

Table 4: Failed episodes across 80 real-world trials per method.

Method	Total	Miss	Slip	Coord.
Diffusion Policy	42	22	12	8
B-spline Policy	45	28	15	2
PAM-DP (ours)	**20**	**10**	**9**	**1**

Compared with Diffusion Policy, PAM-DP has fewer failed episodes attributed to missed grasps, decreasing from 22 to 10, and coordination errors, decreasing from 8 to 1. Failures attributed to object loss decrease from 12 to 9. Figure 6 provides qualitative examples of grasp retries on *fold cloth*. Following an unsuccessful initial grasp, all three policies lift the gripper and re-approach the cloth. In the displayed PAM-DP episode, the second grasp reaches the cloth corner and the fold completes. The displayed Diffusion Policy and B-spline Policy episodes instead grasp away from the corner and produce skewed folds. These examples illustrate individual recovery attempts rather than a comparison of overall recovery success rates.

4.4 Analysis

Component ablations. We study the sketch parameterization, attention structure, and training noise distribution on image-based Push-T (Table 5). All variants are derived from the same PAM-DP configuration and use training seed 42. Except for the specified ablation, we hold the backbone, conditioning, data, optimizer, training budget, and evaluation episodes fixed. We use 100 DDPM sampling steps with zero sketch lead to isolate representation and training effects from staggered denoising and its additional NFE. We report mean coverage and standard deviation across the last ten evaluated checkpoints, each evaluated on the same 50 initial conditions.

Table 5: Component ablations on image-based Push-T.

Variant	Coverage
PAM-DP, zero lead	$\mathbf{0.79} \pm 0.02$
Action-only backbone	0.69 ± 0.07
Auxiliary sketch only	0.75 ± 0.04
Time-indexed sketch	0.74 ± 0.04
Bidirectional attention	0.72 ± 0.06
Shared-time training	0.74 ± 0.07

The *action-only backbone* removes the sketch tokens and sketch loss from PAM-DP. The *auxiliary-only* variant retains sketch prediction and supervision but removes attention from action queries to sketch keys. Sketch supervision can therefore affect actions through shared parameters, but actions

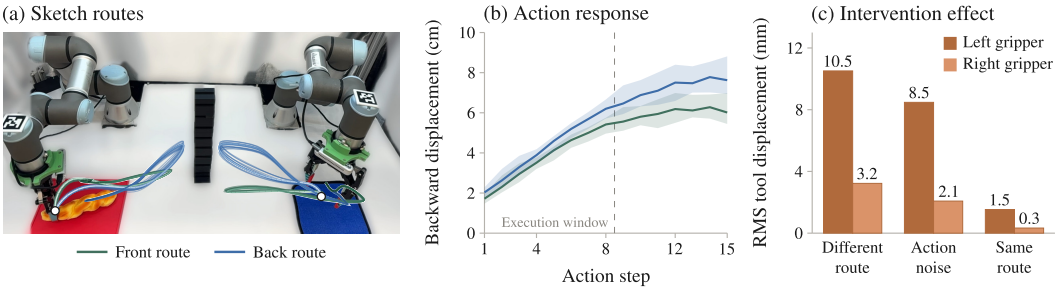


Figure 7: Offline sketch intervention. (a) Alternative sketch routes. (b) Predicted left-gripper motion. (c) Action sensitivity to sketch and noise changes.

cannot read the sketch tokens. The *time-indexed sketch* variant replaces uniform arc-length resampling with uniform-time resampling before B-spline fitting, retaining the same control-point count, token budget, loss weights, and attention structure. The *bidirectional* variant allows all target tokens to attend to one another and to the complete observation memory. The *shared-time* variant ties the sketch and action noise levels during training, whereas zero lead ties them only during sampling. PAM-DP with zero lead achieves mean coverage of 0.79, compared with 0.69 for the action-only backbone and 0.75 for auxiliary-only prediction. Time-indexed sketches, bidirectional attention, and shared-time training reduce coverage to 0.74, 0.72, and 0.74, respectively. Each component therefore contributes to performance, even without staggered denoising.

Sketch-lead sensitivity. We vary the sketch lead with a 16-step DDIM base budget (Table 6). We report mean coverage and standard deviation over three sampler-noise seeds, each evaluated on the same 50 initial conditions. All positive leads yield higher mean coverage than zero lead. The default $\eta = 1/8$, highlighted by shading, increases coverage from 0.779 to 0.805 with 18 rather than 16 NFE. Increasing the lead to $\eta = 1$ yields 0.809 coverage but requires 32 NFE. These results support a modest lead as a practical trade-off between coverage and computation. Appendix C.2 details the lead-evaluation protocol.

Table 6: Sketch lead ablation on Push-T with a 16-step DDIM base budget.

Lead η	Offset	NFE	Coverage	Delay (ms)
0	0	16	0.779 ± 0.015	153.0
1/16	1	17	0.801 ± 0.018	155.2
1/8	2	18	0.805 ± 0.011	158.1
1/4	4	20	0.785 ± 0.029	193.1
1/2	8	24	0.788 ± 0.023	230.0
1	16	32	$\mathbf{0.809} \pm 0.027$	280.0

Sketch intervention. We test sketch dependence offline at a fixed handoff observation using routes passing in front of or behind the barrier (Figure 7a). We hold the observation and initial action noise fixed and replace the sketch denoising trajectory at matching noise levels. Back-route sketches induce greater predicted backward motion of the left gripper (Figure 7b). At action step 8, different-route replacement changes the predicted left-tool position by 10.5 mm RMS, compared with 8.5 mm for action-noise resampling and 1.5 mm for same-route replacement (Figure 7c). Changing the sketch route alters the predicted motion even when the observation and action noise remain fixed. Appendix C.3 describes donor selection, noise pairing, and the displacement metric.

5 Conclusion

We introduced Proprioceptive Action Models, which jointly generate a timing-free B-spline sketch of the remaining configuration-space path and a short executable action chunk through directed attention and staggered denoising. Adding the sketch improves over the corresponding action-only policies on Push-T, LIBERO-Long, and four real-world bimanual tasks. Several limitations remain. The sketch represents the robot's motion without modeling object dynamics or contact. Staggered denoising requires additional NFE, and our lead sweep does not separate the schedule from this added computation. Future work will explore sketches that capture object interactions and evaluate denoising schedules under matched inference budgets.

AI USE STATEMENT

In this work, we used generative AI tools, including agentic coding assistants, for code implementation and data analysis. We did not use generative AI tools for synthetic data generation, methodology design, formulating mathematical claims or proofs, hypothesis refinement, dataset cleaning, or interpreting experimental results; theoretical model development and translation support are not applicable to this work. Additionally, we used generative AI tools to edit the manuscript for clarity and readability and to assist with literature search; we did not use them for figure creation. We have reviewed all AI-assisted work: AI-assisted code was reviewed and tested by the authors, and all reported results were verified by the authors against the evaluation outputs. We take responsibility for the final content of this work, including text, claims, references, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix A describes sketch construction, temporal invariance, and reconstruction audits. Appendix B specifies the model, training objectives, and sampling procedures. Appendix C collects training configurations, dataset splits, evaluation protocols, and the sketch intervention procedure. We will release the code upon acceptance.

REFERENCES

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is Conditional Generative Modeling All You Need for Decision Making? In *International Conference on Learning Representations*, 2023.

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Rich Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-Shot Robotic Manipulation with Pre-Trained Image-Editing Diffusion Models. In *International Conference on Learning Representations*, 2024.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion Forcing: Next-token Prediction Meets Full-Sequence Diffusion. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24081–24125. Curran Associates, Inc., 2024. doi: 10.52202/079017-0759.

Xinzhe Chen, Sihua Ren, Liqi Huang, Haowen Sun, Mingyang Li, Xingyu Chen, Zeyang Liu, and Xuguang Lan. OASIS: Observation-Action Space Alignment via $SE(3)$ Trajectory Prediction for Robotic Manipulation. arXiv preprint arXiv:2605.25829, 2026.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026.

Carl de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer, 2001. ISBN 978-0-387-95366-3.

Danny Driess, Jost Tobias Springenberg, Brian Ichter, Lili Yu, Adrian Li-Bell, Karl Pertsch, Allen Z. Ren, Homer Walke, Quan Vuong, Lucy Xiaoyang Shi, and Sergey Levine. Knowledge Insulating Vision-Language-Action Models: Train Fast, Run Fast, Generalize Better. arXiv preprint arXiv:2505.23705, 2025.

Yiguo Fan, Shuanghao Bai, Xinyang Tong, Pengxiang Ding, Yuyang Zhu, Hongchao Lu, Fengqi Dai, Wei Zhao, Yang Liu, Siteng Huang, Zhaoxin Fan, Badong Chen, and Donglin Wang. Long-VLA: Unleashing Long-Horizon Capability of Vision Language Action Model for Robot Manipulation. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2018–2037. PMLR, 2025.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A. Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit Behavioral Cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 158–168. PMLR, 2022.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. RT-Trajectory: Robotic Task Generalization via Hindsight Trajectory Sketches. In *International Conference on Learning Representations*, 2024.

Xiaoshen Han, Haoyu Xiong, Haonan Chen, Chaoqi Liu, Antonio Torralba, Yuke Zhu, and Yilun Du. B-spline Policy: Accelerating Manipulation Policies via B-spline Action Representations. arXiv preprint arXiv:2607.09648, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video Prediction Policy: A Generalist Robot Policy with Predictive Visual Representations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24328–24346. PMLR, 2025.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with Diffusion for Flexible Behavior Synthesis. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 2022.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.017.

Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. SkillDiffuser: Interpretable Hierarchical Planning via Skill Abstractions in Diffusion-Based Task Execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16467–16476, 2024.

Minghui Lin, Pengxiang Ding, Shu Wang, Zifeng Zhuang, Yang Liu, Xinyang Tong, Wenxuan Song, Shangke Lyu, Siteng Huang, and Donglin Wang. HiF-VLA: Hindsight, Insight and Foresight through Motion Representation for Vision-Language-Action Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20732–20742, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023a. doi: 10.52202/075280-1939.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*, 2023b.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, Zhengping Che, Jian Tang, Pheng-Ann Heng, and Shanghang Zhang. LaST$_0$: Latent Spatio-Temporal Chain-of-Thought for Robotic Vision-Language-Action Model. arXiv preprint arXiv:2601.05248, 2026.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What Matters in Learning from Offline Human Demonstrations for Robot Manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1678–1690. PMLR, 2022.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Data-Efficient Hierarchical Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 2018.

Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. LLARVA: Vision-Action Instruction Tuning Enhances Robot Learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3333–3355. PMLR, 2025.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, 2023.

Jingjing Qian, Boyao Han, Chen Shi, Lei Xiao, Long Yang, Shaoshuai Shi, and Li Jiang. GeoPredict: Leveraging Predictive Kinematics and 3D Gaussian Geometry for Precise VLA Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13529–13539, 2026.

Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. FLOWER: Democratizing Generalist Robot Policies with Efficient Vision-Language-Flow Models. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3736–3761. PMLR, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*, 2021.

Mengze Tian, Yiming Li, Sichao Liu, Auke Ijspeert, and Sylvain Calinon. Spline Policy: A Structured Representation for Robot Policies. arXiv preprint arXiv:2606.07386, 2026.

Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive Inverse Dynamics Models are Scalable Learners for Robotic Manipulation. In *International Conference on Learning Representations*, pp. 92033–92052, 2025.

Junjie Wen, Yichen Zhu, Minjie Zhu, Zhibin Tang, Jinming Li, Zhongyi Zhou, Xiaoyu Liu, Chaomin Shen, Yaxin Peng, and Feifei Feng. DiffusionVLA: Scaling Robot Foundation Models via Unified Diffusion and Autoregression. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66558–66574. PMLR, 2025.

Chongyang Xu, Haipeng Li, Shen Cheng, Haoqiang Fan, Ziliang Feng, and Shuaicheng Liu. Action-Geometry Prediction with 3D Geometric Prior for Bimanual Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35036–35046, 2026a.

Xiaoxu Xu, Hao Li, Jinhui Ye, Yilun Chen, Jia Zeng, Xinyi Chen, Linning Xu, Dahua Lin, Weixin Li, and Jiangmiao Pang. FutureVLA: Joint Visuomotor Prediction for Vision-Language-Action Model. arXiv preprint arXiv:2603.10712, 2026b.

Fan Yang, Peiguang Jing, Kaihua Qu, Ningyuan Zhao, and Yuting Su. ABPolicy: Asynchronous B-Spline Flow Policy for Real-Time and Smooth Robotic Manipulation. arXiv preprint arXiv:2602.23901, 2026.

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

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic Control via Embodied Chain-of-Thought Reasoning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3157–3181. PMLR, 2025.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems*, 38: 24195–24228, 2026.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. CoT-VLA: Visual Chain-of-Thought Reasoning for Vision-Language-Action Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713, 2025.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé, III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual Trace Prompting Enhances Spatial-Temporal Awareness for Generalist Robotic Policies. In *International Conference on Learning Representations*, pp. 54277–54296, 2025.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.015.

A Sketch Construction and Theoretical Properties

A.1 Path Preprocessing and Phase Parameterization

At each timestep t, we use the remaining measured configuration path $\mathbf{q}_{t:T}$ to construct the sketch target. For the dual-arm platform, we concatenate the two arms' joint positions and exclude gripper states. We subtract $\mathbf{q}_t$ and compute cumulative Euclidean arc length with equal weight across configuration dimensions, following Eq. 3.

Gripper commands remain in the action stream. Excluding grippers from the sketch prevents their discrete state changes from affecting the arm-motion phase. We linearly interpolate the relative path onto $N = 64$ uniformly spaced phase samples, including both endpoints. Before interpolation, we retain the initial sample and each subsequent sample whose arc-length increment from the preceding original sample exceeds a numerical tolerance ϵ. After interpolation, we explicitly set the first and last positions to $\mathbf{0}$ and $\mathbf{q}_T - \mathbf{q}_t$. The resulting samples are used for the spline fit.

We set $\epsilon = \max(10^{-8}R, \epsilon_{\text{mach}})$, where R is the largest per-coordinate range over the full configuration array supplied to target construction and ϵ_{mach} is double-precision machine epsilon. In the dual-arm implementation, this array contains the configurations from the entire replay buffer. If the remaining path has a single sample or total length $L < \epsilon$, we set its sketch target to zero. The tolerance ϵ removes numerically repeated samples but does not threshold measurement noise; joint-encoder jitter during stationary intervals therefore contributes small phase increments.

A.2 B-spline Representation and Target Fitting

We fit the resampled relative path with a clamped cubic B-spline using M control points. The knot vector contains four repeated knots at each endpoint, zero and one, with interior knots at $j/(M-3)$ for $j = 1, \ldots, M-4$. We use the standard Cox–de Boor basis $B_{m,3}$ (de Boor, 2001).

Let $\Delta\mathbf{q}_{t,n}$ denote the resampled relative configuration at phase $\bar{\phi}_n = n/(N-1)$ for $n = 0, \ldots, N-1$. We fit the coefficients by endpoint-constrained least squares,

$$\mathbf{Z}_t^\star = \arg\min_{\mathbf{z}_t} \sum_{n=0}^{N-1} \left\| \sum_{m=0}^{M-1} B_{m,3}(\bar{\phi}_n)\, \mathbf{z}_{t,m} - \Delta\mathbf{q}_{t,n} \right\|_2^2, \quad \text{subject to} \quad \mathbf{z}_{t,0} = \mathbf{0}, \qquad \mathbf{z}_{t,M-1} = \mathbf{q}_T - \mathbf{q}_t. \tag{7}$$

We subtract the fixed endpoint contribution and solve for the interior coefficients using a pseudoinverse. The basis matrix and pseudoinverse are precomputed for each choice of M and N.

Temporal invariance. For a Lipschitz path $\mathbf{q} : [0, D] \to \mathbb{R}^{d_q}$ of positive arc length and an absolutely continuous, non-decreasing, surjective time map $\rho : [0, D'] \to [0, D]$, cumulative arc length satisfies $s_{\mathbf{q}\circ\rho} = s_{\mathbf{q}} \circ \rho$. The normalized arc-length path is therefore unchanged, so a fixed phase grid, basis, and coefficient budget M yield identical fitting targets, endpoint constraints, and pseudoinverse solutions. This invariance includes changes in speed and inserted pauses that preserve traversal order, but not reversal or additional retracing. Finite sampling, measurement noise, and numerical tolerances can still produce differences between separately recorded demonstrations. Removing timing holds for every fixed M, whereas geometric fidelity depends on the capacity selected below. For the dual-arm platform, the invariance applies to a common reparameterization of both arms. Changing the timing of one arm relative to the other alters the path in the joint configuration space, so the sketch retains the relative coordination of the two arms.

A.3 Capacity Audit and Normalization

We audit the adopted coefficient budgets after training using a common reconstruction metric. We measure $E_{\mathrm{rec}}(M)$ as the scalar RMSE between each fitted path and its 64 uniformly spaced phase targets, weighting anchors, phase samples, and configuration dimensions equally. All capacity audits use training episodes. Push-T and the dual-arm tasks use training-sampler anchors, while LIBERO uses every frame. We compute the per-step motion reference δ_{step} as the root mean square of configuration increments over all adjacent frame pairs within training episodes and all configuration dimensions, retaining stationary steps. LIBERO recordings are sampled at 20 Hz.

Figure 8 shows reconstruction errors normalized by each task's per-step motion reference across the common candidate set and marks the capacities used in the policy experiments. The adopted capacities satisfy $E_{\mathrm{rec}}(M) < \delta_{\mathrm{step}}$ on all six tasks; we do not require the smallest budget satisfying this criterion. For LIBERO-Long, the per-step reference is 0.0116 rad and the reconstruction RMSE at $M = 16$ is 0.0098 rad, giving a normalized error of approximately 0.845. These post-hoc audits verify the reconstruction quality of the capacities used in training.

We normalize each predicted coefficient and configuration dimension independently using training-set minima and maxima. In the dual-arm implementation, these statistics use targets indexed by the training sampler; nonconstant dimensions map to $[-1, 1]$, while nearly constant dimensions retain unit scale and only subtract the training minimum. Separately from the training-set capacity audit, we audit clipping on fitted validation targets by normalizing their coefficients, clamping to $[-1, 1]$, inverting the transform, and decoding the paths. Scalar reconstruction RMSE changes from 0.568 to 0.571 degrees for handoff, 0.528 to 0.541 for close lid, 0.563 to 0.566 for fold cloth, and 0.596 to 0.602 for measure. Push-T remains at 3.08 pixels at the reported precision. These values measure distortion of fitted targets rather than model prediction error.

B Model Implementation

This section specifies token processing, training targets, and sampling updates for the joint policy in Section 3.3. Experiment-specific dimensions, optimization settings, and inference budgets are collected in Appendix C.

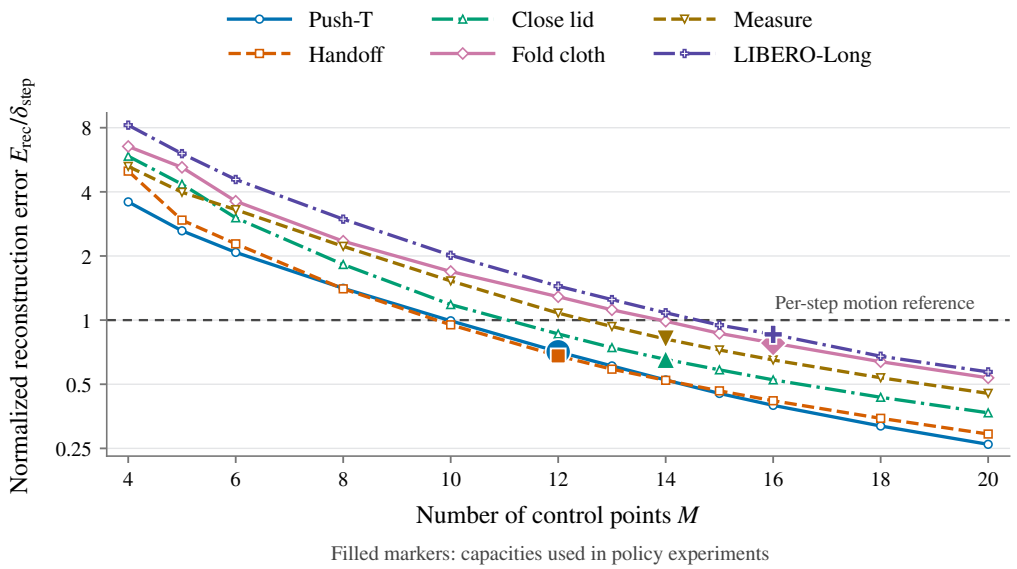


Figure 8: Post-hoc sketch capacity audit on training anchors. Reconstruction RMSE is normalized by the task-specific per-step motion reference and shown on a logarithmic vertical axis. The dashed line marks a ratio of one; filled markers indicate the capacities used in the policy experiments. The reference values are 8.04 pixels for Push-T and 0.870, 0.700, 0.765, 0.660, and 0.665 degrees for handoff, close lid, fold cloth, measure, and LIBERO-Long, respectively.

B.1 Architecture

Token streams. PAM-DP projects each of the $M-1$ normalized sketch coefficients and each of the H normalized action steps into a shared embedding space using separate linear layers. Each stream has learned positional embeddings, and sketch tokens receive an additional learned type embedding. The streams are concatenated, processed by a shared transformer decoder, and mapped back to their respective dimensions by separate output heads.

Attention and observations. Sketch tokens attend causally within the sketch stream and never read action tokens. Action tokens attend to all sketch tokens and causally within the action stream.

The observation encoder processes each timestep independently, combining image and low-dimensional inputs into a feature vector. These features are projected, combined with learned positional embeddings, and passed through a position-wise MLP to form the cross-attention memory. There is no temporal mixing in this conditioning network. Sketch queries read all observation positions, whereas action queries read observation positions up to their own time index, following the transformer Diffusion Policy.

Noise conditioning. The main Push-T model uses adaLN-zero conditioning. The sketch and action timesteps are embedded separately and broadcast to the tokens of their respective streams. Each decoder block uses these embeddings to modulate the normalized self-attention and feed-forward inputs and to gate the self-attention, observation cross-attention, and feed-forward residual updates. The modulation projections and output heads are zero-initialized. The final normalization is also time-conditioned.

PAM-VLA. PAM-VLA augments FLOWER's rectified-flow action expert with sketch prediction. Separate MLPs encode sketch coefficients and actions into a shared DiT, and separate linear heads predict their velocities. Rotary positional indices restart at zero for each stream. Sketch self-attention is causal and excludes action keys; actions attend to all sketch tokens and causally to other actions. Projected VLM features provide cross-attention context. Noise-level and control-frequency embeddings condition the tokens and adaptive normalization. The pretrained Florence-2 backbone, including its vision tower, is fine-tuned rather than frozen.

B.2 TRAINING AND SAMPLING

Diffusion training. The Push-T model uses a 100-step cosine diffusion schedule. For each example, it independently samples one discrete timestep for the sketch and another for the actions, uniformly over the training schedule. Every token in a stream shares its timestep, while Gaussian noise is sampled independently for every token and feature. In Eq. 2, the diffusion scales are $\alpha = \sqrt{\bar{\alpha}_k}$ and $\sigma = \sqrt{1 - \bar{\alpha}_k}$ at scheduler index k, where $\bar{\alpha}_k$ is the cumulative signal-retention product. Here, the prediction target is the sampled noise $\boldsymbol{\epsilon}$, and f_θ estimates this target as $\hat{\boldsymbol{\epsilon}}$. Each stream's mean squared error is averaged over the batch, tokens, and feature dimensions before applying the weights in Eq. 5.

The real-world experiments use adaLN-zero conditioning and a 100-step cosine diffusion schedule with noise prediction. The independent-time model samples separate sketch and action timesteps, shared within each stream, while the shared-time control uses one timestep across both streams. Offline checkpoint validation uses synchronous 16-step DDIM. For the main real-world results, inference uses a sketch lead of $\eta = 1/8$.

Rectified-flow training. PAM-VLA independently samples one uniform noise coordinate for the sketch stream and one for the action stream, shared by all tokens within the respective stream. It uses linear interpolation with $\alpha(\gamma) = 1 - \gamma$ and $\sigma(\gamma) = \gamma$, where zero is clean and one is noise. The velocity target is $\boldsymbol{\epsilon}_i - \mathbf{x}_i^0$, and each stream's squared velocity error is averaged before weighting.

Discrete sketch lead. The Push-T sampler realizes Algorithm 1 as an offset in the inference timestep list, where the normalized coordinates γ_Z and γ_A index a list of K scheduler timesteps and UPDATE is one DDPM or DDIM step. The requested lead η is converted to an integer offset $k_{\text{lead}} = \lfloor \eta K + 0.5 \rfloor \in \{0, \ldots, K\}$, rounding half-integer offsets upward. For discrete sampling, Algorithm 1 uses k_{lead}/K in place of η. For $K = 100$ and $\eta = 1/8$, the offset is 13, giving 113 NFE and exactly 100 updates per stream. Both streams start from independent Gaussian noise. The sampler first performs k_{lead} sketch updates while holding the action noise fixed. It then performs K action updates, with the sketch advancing through its own schedule k_{lead} positions ahead. Once the sketch has completed all K updates, it remains fixed while action denoising finishes. Each model call reads both streams at their current local timesteps, and each active stream uses its own predicted noise $\mathbf{u}$ in the update. This schedule requires $K + k_{\text{lead}}$ NFE. Zero lead updates both streams synchronously; a lead of K completes sketch generation before updating actions. We use $\eta = 1/8$ for the main results. Once the sketch is fixed, its conditioning timestep remains the final index in the scheduler's inference list.

The Push-T training configuration evaluates checkpoints with 100-step DDPM and a sketch lead of $\eta = 1/8$. Subsequent lead sweeps use deterministic DDIM with $\eta_{\text{DDIM}} = 0$, retaining the same 100-step training noise schedule. The main-result sketch lead is applied at inference; it does not change the training corruption distribution. Per-experiment samplers and base budgets are reported in Appendix C.

Rectified-flow sampling. PAM-VLA runs Algorithm 1 with UPDATE as an explicit Euler step: token i is updated as $\mathbf{x}_i \leftarrow \mathbf{x}_i - (\gamma_i^{\text{old}} - \gamma_i^{\text{new}})\mathbf{u}_i$, where $\mathbf{u}_i$ is the predicted velocity, and tokens whose noise level is unchanged remain fixed. The stream-wise noise coordinates follow Eq. 6 directly, without discrete lead rounding. LIBERO inference uses $K = 8$ and $\eta = 1/8$, giving nine NFE. Final normalized sketch and action outputs are clipped to $[-1, 1]$.

Action execution and ablations. After sampling, the policy inverts the sketch and action normalizers and restores the sketch's zero anchor. Push-T uses two observation steps and predicts 16 action positions, executing eight actions before replanning. The sketch is decoded for inspection but is never executed.

Shared-time training ties the two corruption levels during optimization. Zero-lead sampling instead ties their schedules only at inference and can be applied to an independently trained model. These are distinct ablations.

Table 7: PAM-DP training settings. Real-world settings are shared by the four dual-arm tasks. Both configurations use EMA and a 100-step cosine diffusion schedule with noise prediction.

Setting	Push-T image	Real-world
Decoder layers / heads / width	8 / 4 / 256	8 / 4 / 256
Conditioning	adaLN-zero	adaLN-zero
Attention dropout	0.01	0
Observation / prediction / execution steps	2 / 16 / 8	2 / 16 / 8
Image crop	84×84	84×84
Batch size	64	64
Optimizer	AdamW	AdamW
Learning rate	10^{-4}	10^{-4}
Adam coefficients	$0.9,\ 0.95$	$0.9,\ 0.95$
Denoiser weight decay	10^{-1}	10^{-3}
Observation-encoder weight decay	10^{-6}	10^{-6}
Learning-rate schedule	Cosine	Cosine
Warmup updates	1,000	500
Training budget	3,050 epochs	Matched across methods

Table 8: PAM-VLA configuration on LIBERO-Long. The sketch has $M = 16$ control points and predicts 15 coefficient vectors after fixing the first to zero.

Setting	PAM-VLA
Vision-language backbone	Florence-2-large
Decoder layers / heads / width	18 / 16 / 1024
Positional encoding	RoPE
Attention / residual / MLP dropout	0.1 / 0.1 / 0.1
Observation / prediction / execution steps	1 / 10 / 10
Batch size	16
Optimizer	AdamW
Peak learning rate	2×10^{-5}
Adam coefficients	$0.9,\ 0.95$
Weight decay	0.05
Training limit	45 epochs, at most 1,000 batches per epoch
Precision	BF16 mixed precision
Sketch / action loss weights	0.2 / 1.0
Sampler / base steps / lead / NFE	Euler / 8 / $1/8$ / 9

C Experimental Configurations and Evaluation Protocols

C.1 Training and Inference Configurations

Table 7 summarizes the PAM-DP training settings. Push-T uses the image-based $H = 16$ model and adopts the transformer hyperparameters reported by Chi et al. (2023). The real-world settings are shared by handoff, close lid, fold cloth, and measure. The sketch loss weight is 0.15 for close lid and 0.2 for the other three real-world tasks. The action loss weight is one.

Table 8 reports the PAM-VLA settings. Florence-2-large, including its vision tower, is fully fine-tuned. The learning rate rises from 2×10^{-6} to 2×10^{-5} over 1,500 updates, remains constant for 3,000 updates, and follows cosine decay to 10^{-5} over 25,500 updates. It remains at 10^{-5} thereafter. EMA uses an inverse-power warmup with power $2/3$, inverse scale one, and maximum decay 0.9999; evaluation uses the EMA weights.

PAM-DP also uses inference lead $1/8$ for the main results. The 16-step DDIM lead experiment has $k_{\text{lead}} = 2$ and therefore uses $K + k_{\text{lead}} = 18$ NFE at this lead.

Baselines. The direct comparisons use action-only Diffusion Policy for PAM-DP and FLOWER for PAM-VLA. B-spline Policy predicts executable action curves. On Push-T, B-spline Policy is trained with the transformer backbone for three training seeds and 3,050 epochs under the same protocol

as Diffusion Policy. In the real-world experiments, all three methods use the same visual encoder, EMA, and number of optimizer updates on each task.

C.2 Datasets and Evaluation Protocols

Demonstrations. Table 9 gives the available demonstrations and the seed-42 training/validation splits. Push-T follows the data split of Chi et al. (2023). LIBERO uses all 500 demonstrations for training. Validation episodes are excluded from training-normalizer statistics.

Table 9: Demonstration counts and seed-42 splits. Push-T is not listed and follows the split of Chi et al. (2023). LIBERO has no demonstration-level validation split.

Task	Available	Training	Validation
LIBERO-Long	500	500	0
Handoff	81	73	8
Close lid	81	73	8
Fold cloth	48	41	7
Measure	89	80	9

Push-T. Coverage is the maximum target overlap achieved during an episode, averaged over evaluation initial conditions. Table 1 reports checkpoint aggregates over 50 initial conditions and three training seeds. For image-based training, the evaluation launcher assigns 50 consecutive initial-condition seeds starting at $100000(s+1)$ for training seed s. Following Chi et al. (2023), we report the maximum coverage over checkpoints and the mean over the last ten checkpoints, with checkpoints evaluated every 50 epochs. Results for LSTM-GMM, IBC, and Diffusion Policy in Table 1 are taken from Chi et al. (2023); we train B-spline Policy and PAM-DP under the same protocol and report the same two aggregates.

The component ablations in Table 5 use training seed 42, the same 50 initial conditions, 100-step DDPM, and zero lead. Their mean and standard deviation describe the last ten evaluated checkpoints, not independent training runs.

Lead evaluation. Table 6 uses a single seed-42 final-epoch checkpoint, without score-based checkpoint selection, and deterministic 16-step DDIM reconstructed from the training schedule. Each lead is evaluated on initial-condition seeds 4300000–4300049 with three sampler-noise seeds. Initial noise is paired by episode and decision step across leads; reported standard deviations are across sampler-noise seeds. We report computational cost as NFE per action chunk.

LIBERO-Long. We evaluate each of the ten tasks over 50 trials per policy and report the mean success rate across tasks. The main table aggregates PAM-VLA over five training seeds. Baseline results in Table 2, including FLOWER, are taken from their original publications, except for π_0, whose LIBERO-Long result is taken from Kim et al. (2025).

Real-world tasks. Each method is evaluated over 20 episodes per task with initial object placements sampled from the same distribution. Each episode runs for at most 120 s without human intervention; the policy may regrasp autonomously within this limit, and an episode that has not succeeded by 120 s is counted as a failure. In *handoff*, one arm grasps the loaf and passes it to the other arm around the barrier, which places it on the blue mat. In *close lid*, the lid rests on the pot and the pot rests on the red plate; the lid need not be fully seated. In *fold cloth*, the cloth is folded twice into the goal configuration shown in Figure 4. In *measure*, the arms extend the tape measure along the loaf and then release it. Completion time is reported as mean $\pm$ standard deviation over successful episodes only; the overall time pools successful episodes across tasks. Each failed episode receives one label for the earliest error that led to failure, and recovered errors in successful episodes are not counted in Table 4.

For the real-world tasks, offline checkpoint selection uses EMA predictions and episode-averaged physical joint-action RMSE, averaged over five inference seeds with synchronous 16-step DDIM. All three methods use this selection rule, and robot evaluation episodes are not used for checkpoint selection.

C.3 SKETCH INTERVENTION PROTOCOL

The offline intervention in Figure 7 fixes one handoff observation and the initial action noise, then replaces the sketch denoising trajectory with a donor trajectory at matching noise levels. We use EMA weights and deterministic 16-step DDIM. Donor sketches are expressed relative to the current joint configuration. They are replayed at each action-updating model call, so noisy sketch inputs retain the appropriate corruption level.

At inference step 120 of one episode, we generate 20 native front-route sketches. We take 20 additional front-route donors from the preceding observation and 20 back-route donors from a later observation in the same episode. Each set is paired with the same ten action noises. The different-route comparison uses the two transplanted sets; the same-route comparison uses native and transplanted front-route sets. The noise control varies action noise while holding the sketch fixed.

We measure the root mean squared Euclidean displacement between paired predicted tool positions at each action step, using forward kinematics with the fitted tool reference. This tests action dependence at a fixed observation, not closed-loop task success or collision clearance.

Visualization. Robot predictions are projected through UR3 forward kinematics into a calibrated recording camera separate from the policy cameras. Figure 5 uses a nominal flange offset of 0.20 m; these overlays are qualitative. Projected coefficient markers are not image-space spline control points. In Figure 7a, paths are translated in the image plane to start at the visible gripper center for display only; quantitative panels retain the original tool-reference coordinates.