LaDyP: Phase-Aware Action Chunking

Anonymous authors
Paper under double-blind review

Abstract

Action-chunking policies predict several future actions from the current observation, so a single chunk often spans a grasp, a release, or the switch to the next sub-task. On long-horizon instructions these transitions are where most rollouts fail. We introduce LaDyP, a phase-conditioning interface for Flow-based chunk policies. An offline teacher labels demonstrations with recurring interaction roles, within-role progress, and control-boundary evidence; a causal estimator predicts a soft phase belief, local progress, and current and upcoming boundary likelihoods from observation history. These signals condition the action generator through a global token, chunk-position embeddings, and a boundary-gated residual velocity. We attach LaDyP to a Flow policy trained from scratch and to the pretrained SmolVLA, and evaluate both on all four LIBERO suites with 1,500 paired trials per suite and method. On LIBERO-Long, LaDyP raises success from 49.2% to 57.5% for the Flow policy and from 71.8% to 78.7% for SmolVLA, while gains on the short suites are small. Delaying the signals by 32 steps or replacing all of them with signals from another episode removes most or all of the gain, and failures during engagement, release, and sub-task switches decrease by 31% and 30% for the two backbones while approach and transport failures stay near their Base counts.

1 Introduction

Consider the LIBERO-Long instruction *put both moka pots on the stove*. A policy must approach the first pot, close the gripper, carry the pot, release it on the burner, and repeat the sequence for the second pot. Action-chunking policies predict a block of future actions from the current observation (Zhao et al., 2023; Chi et al., 2024; Black et al., 2025a; Shukor et al., 2025), so one chunk generated during the approach may already contain the actions that should follow contact. When the chunk places the gripper closure a few steps early or carries transport motion past the release point, the policy must recover from a state that no demonstration visited. On this benchmark, the Flow policy we train from scratch fails 762 of 1,500 LIBERO-Long trials, and 514 of those failures occur during engagement, release, or the switch to the next sub-task (Section 4.5).

Recent observations can implicitly encode where the robot is in the task, but they do not tell the action generator that a transition is imminent, and they do not separate the first grasp from the second grasp of the same instruction. Prior methods add temporal structure through a discrete phase or mode that selects an action type, an expert, an input mask, or a language motion (Belkhale et al., 2023; Xu et al., 2026; Fan et al., 2025; Belkhale et al., 2024), which requires a hard decision at each switch. Conditioning on elapsed episode time avoids hard decisions but conflates tasks of different length.

We introduce LaDyP, a phase-conditioning interface for Flow-based chunk policies (Figure 1). An offline teacher labels demonstrations with seven recurring interaction roles, progress within the current role, and control-boundary evidence from gripper and motion changes. A causal estimator predicts these quantities from observation history as a soft phase belief, local progress, and the likelihoods of a boundary now and within the next 16 steps. The action generator receives them through a global token, position-dependent chunk embeddings, and a residual velocity whose gain rises with the predicted boundary likelihood, so it keeps one generator and one chunk length and never commits to a discrete mode.

Our contributions are the following.

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.

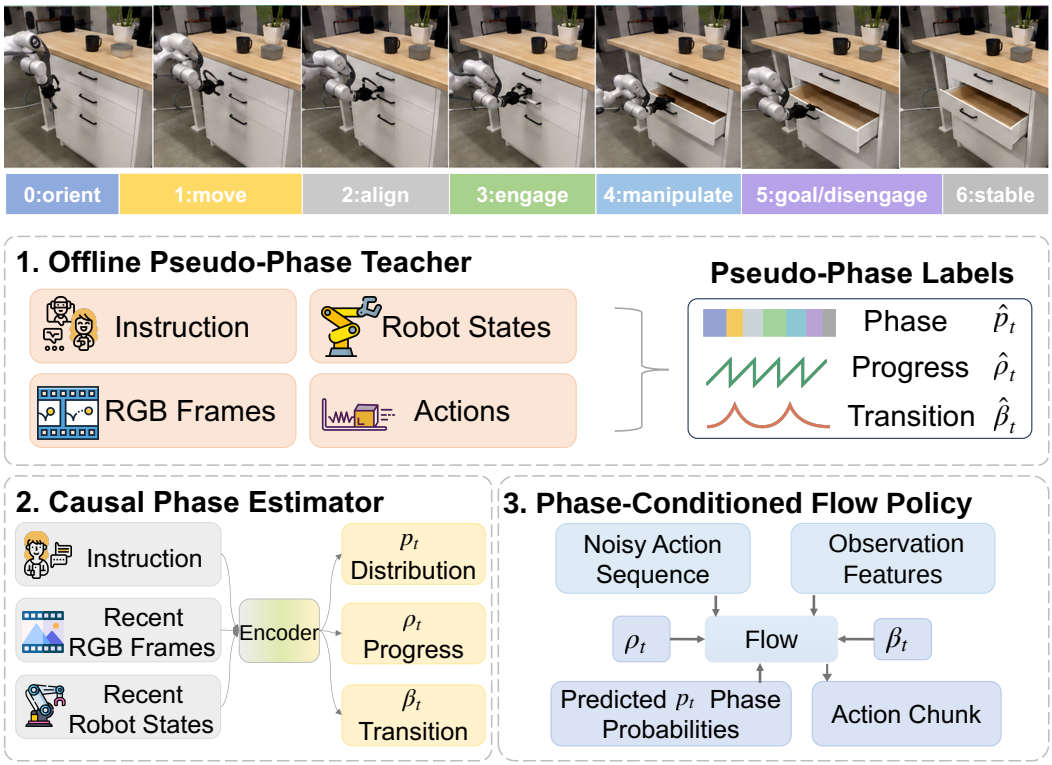

Figure 1: LaDyP overview. An offline teacher labels complete demonstrations with phase, progress, and transition targets ($\hat{p}_t, \hat{\rho}_t, \hat{\beta}_t$); a causal encoder predicts the corresponding signals (p_t, ρ_t, β_t) from recent observations, and these condition action-chunk generation. The encoder and Flow action generator use Transformers; β_t abbreviates the current and horizon channels. The drawer sequence illustrates the seven interaction roles.

- A causal phase interface, consisting of a teacher, an estimator, and three conditioning paths, that attaches to a Flow policy trained from scratch and to the pretrained SmolVLA without changing their chunk length or execution cadence.
- On all four LIBERO suites with 1,500 paired trials per suite and method, LaDyP improves LIBERO-Long success by 8.3 percentage points (pp) for the Flow policy and 6.9 pp for SmolVLA, with smaller changes on the short suites.
- Controls with the same backbone and training budget show that the gain depends on temporally aligned signals: no shuffled, clock, discrete-option, or clustered-segment signal reaches LaDyP, and a 32-step delay removes most or all of the gain. Failure attribution locates the change at engagement, release, and sub-task switches.

2 Related Work

Action-Chunking Policies. ACT predicts fixed-length action chunks with a Transformer, and Diffusion Policy generates receding-horizon chunks by iterative denoising (Zhao et al., 2023; Chi et al., 2024). Flow matching replaces denoising with an integrated velocity field and underlies π_0 and SmolVLA (Lipman et al., 2023; Black et al., 2025a; Shukor et al., 2025). Generalist vision-language-action models such as RT-2, Octo, and OpenVLA inherit pretrained visual and language representations (Brohan et al., 2023; Octo Model Team et al., 2024; Kim et al., 2024), and OpenVLA-OFT studies parallel continuous decoding with action chunks (Kim et al., 2025). LaDyP keeps their chunked structure and adds a prediction of where the current chunk lies relative to interaction transitions.

Temporal Structure in Control. HYDRA switches between sparse waypoints and dense actions, RT-H predicts a language motion before each action, and HULC learns discrete latent plans (Belkhale et al., 2023; 2024; Mees et al., 2022). Long-VLA masks visual inputs according to a phase label, and Move-Then-Operate routes movement and interaction to separate action experts (Fan et al., 2025; Xu et al., 2026). $\pi_{0.5}$ and Hi Robot predict a language subtask before low-level action generation, and CoT-VLA predicts subgoal images (Physical Intelligence et al., 2025; Shi et al., 2025; Zhao et al., 2025). These methods decide which mode or subtask comes next. LaDyP instead estimates where the current chunk lies relative to the next interaction transition and conditions one action generator on a soft phase belief and two continuous boundary likelihoods, so it is compatible with subtask prediction.

Learning Temporal Labels. TACO aligns task sketches with demonstrations, CompILE learns latent segments, and PRISE compresses action sequences into temporal abstractions (Shiarlis et al., 2018; Kipf et al., 2019; Zheng et al., 2024). Time-contrastive networks and temporal cycle-consistency learn video representations aligned in time, and XIRL and GVL estimate task progress for reward or value learning (Sermanet et al., 2018; Dwibedi et al., 2019; Zakka et al., 2021; Ma et al., 2024). The LaDyP teacher is rule-based, separates semantic segment divisions from control events, and defines progress within each interaction role, which conditions the policy rather than scoring it. We compare it with clustered segments in Section 4.3.

Execution Timing. RTC and REMAC address asynchronous chunk execution, PACE adapts the replanning horizon, and bidirectional decoding selects among sampled chunks for temporal consistency (Black et al., 2025b; Wang et al., 2026; Nie et al., 2026; Liu et al., 2024). LaDyP keeps the generation and execution horizons fixed and changes only the content of each chunk.

3 METHOD

3.1 PROBLEM SETTING

A language-conditioned policy receives an instruction ℓ and, at environment step t, an observation $o_t = (I_t, s_t)$ with RGB images I_t and proprioceptive state s_t. It predicts an action chunk $A_t = (a_t, \dots, a_{t+K-1}) \in \mathbb{R}^{K \times d_a}$, executes the first M actions, and replans. We use $M = 8$ for both backbones; the chunk length is $K = 16$ for the Flow policy and $K = 50$ for SmolVLA, and $d_a = 7$ (end-effector translation, rotation, and gripper command). The Flow policy conditions on its last four replan observations, and SmolVLA on its current observation. LaDyP adds a signal estimator whose history h_t contains the 32 most recent replan observations up to step t. Flow interpolation time $\tau \in [0, 1]$ is a separate variable internal to action generation.

The estimator outputs

$$z_t = f_\psi(h_t, \ell) = (p_t, \rho_t, \beta_t^{\text{now}}, \beta_t^{\text{hor}}), \qquad p_t \in \Delta^6, \quad \rho_t, \beta_t^{\text{now}}, \beta_t^{\text{hor}} \in [0, 1]. \tag{1}$$

The phase belief p_t is a distribution over seven recurring interaction roles, ρ_t is progress within the current role, and β_t^{now} and β_t^{hor} are the likelihoods that a control transition occurs at t and within the next $H = 16$ steps. All four depend only on observations available at t; complete demonstrations are used only to build training targets.

3.2 OFFLINE PSEUDO-PHASE TEACHER

The teacher assigns every demonstration step a role label $\hat{p}_t$, a progress target $\hat{\rho}_t$, and boundary targets $\hat{\beta}_t^{\text{now}}, \hat{\beta}_t^{\text{hor}}$ (Figure 2). The seven roles are orient, move, align, engage, manipulate, goal/disengage, and stable. A demonstration can omit roles, and a multi-object instruction revisits them.

Control Events. For each training demonstration, the teacher scores every record with a weighted sum e_t of five normalized cues: a gripper change, the action-change magnitude, end-effector acceleration, the change of frozen visual features, and a motion start/stop indicator (Equation 8). Local maxima above 0.35 become control events, and events with $e_t \geq 0.55$, such as gripper changes with a large action change, are strong. The current-boundary target $\hat{\beta}_t^{\text{now}}$ equals e_t at events and zero

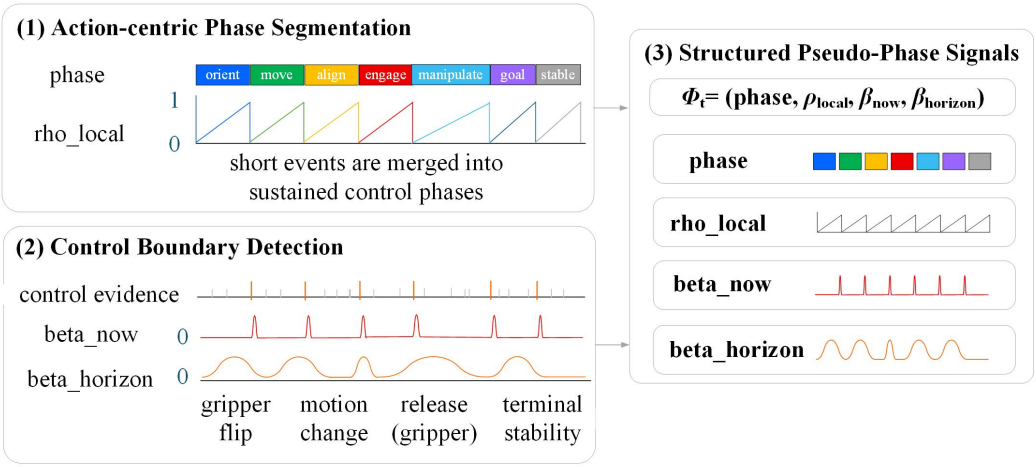

Figure 2: Offline pseudo-phase teacher on a schematic two-atom timeline. (a) Role segmentation gives the phase target $\hat{p}_t$ and the local progress $\hat{\rho}_t$, which resets at every segment. (b) Control events give $\hat{\beta}_t^{\text{now}}$; the horizon target $\hat{\beta}_t^{\text{hor}}$ (Equation 2) holds the largest event score within the next H steps. (c) The four targets supervise the estimator.

elsewhere, and the horizon target summarizes the next H steps:

$$\hat{\beta}_t^{\text{hor}} = \max_{0 \le j < H} \hat{\beta}_{t+j}^{\text{now}}. \tag{2}$$

Semantic Segments. A language model parses each instruction once, offline, into ordered atoms with an action family, object, target, and release flag (Appendix B.2). Atom divisions are placed at the strong events that complete each atom, such as a release. Within an atom, engage starts two records before the gripper-closing event, manipulate follows once the end effector moves away, and goal/disengage starts at the last strong event before the release; align covers the ten records before engage, move the records before align, orient the initial settling period, and stable a low-motion tail after the final release. The orient/move, move/align, and align/engage divisions are placed at fixed offsets and create no control events, so semantic divisions and control events remain distinct targets. For every segment $[u, v]$, progress is $\hat{\rho}_t = (t - u)/\max(1, v - u)$, which resets at each segment and differs from elapsed episode time. Divisions derived from interaction evidence receive higher confidence than those at fixed offsets; the confidences weight the estimator loss and are never policy inputs. Algorithm 3 gives the construction, and Figure 8 in Appendix B.3 shows the output for a two-object demonstration. Against 800 frames labeled by two annotators, the teacher assigns the resolved role on 87.6% of frames, and its control events reach an F1 of 81.5% at a five-step tolerance (Appendix B.5).

3.3 Causal Signal Estimator

The estimator encodes each history record with frozen SigLIP 2 features (Tschannen et al., 2025): 49 pooled spatial tokens and a global token from the agent-view image, a projection of the 15-dimensional state, and the pooled instruction embedding. A four-query attention pool summarizes the spatial tokens of each record, and a four-layer causal Transformer (width 512, eight heads) processes the 32 records. Separate heads on the final record predict seven phase logits, sigmoid-bounded progress, and the two boundary likelihoods. The objective is

$$\mathcal{L}_{\text{est}} = \mathcal{L}_p + 0.5\,\mathcal{L}_\rho + 1.2\,\mathcal{L}_\beta, \tag{3}$$

where $\mathcal{L}_p$ is confidence-weighted cross-entropy, $\mathcal{L}_\rho$ is confidence-weighted absolute error, and $\mathcal{L}_\beta$ averages a focal binary cross-entropy with positive weight four over the current and horizon channels (Appendix B.4). The positive weight and focal term compensate for the sparsity of current-boundary

targets, which are nonzero only at events. The estimator is trained on training demonstrations and then frozen, so the action loss cannot change the signal definitions; Section 4.4 reports a joint-training alternative.

3.4 Phase-Conditioned Action Generation

The estimator output is expanded with phase interactions and uncertainty:

$$r_t = [p_t, \rho_t, \beta_t, p_t\rho_t, \mathrm{vec}(p_t\beta_t^\top), H(p_t)] \in \mathbb{R}^{32}, \qquad \beta_t = [\beta_t^{\mathrm{now}}, \beta_t^{\mathrm{hor}}]. \tag{4}$$

The entropy $H(p_t) = -\sum_c p_{t,c} \log p_{t,c}$ records uncertainty about the current role, and the product terms let the same progress or boundary value carry a different meaning in different roles. The representation reaches the action generator through three paths. First, an MLP maps r_t to a global token appended to the observation sequence attended by the action generator. Second, chunk position j with normalized index $u_j = j/(K-1)$ receives

$$e_{t,j} = \mathrm{MLP}_{\mathrm{local}}([u_j, r_t, u_j r_t]), \qquad j = 0, \ldots, K-1, \tag{5}$$

which is added to the embedding of the j-th noisy action, so that the effect of a fixed phase estimate can differ between early and late actions of the chunk. Third, let $v_\theta(X_\tau, \tau, c_t)$ be the velocity predicted by the backbone for noisy chunk X_τ, where c_t contains visual, language, state, and phase tokens. LaDyP adds a residual branch R_θ that reads the projected noisy actions, the local embeddings $e_t = (e_{t,j})_j$, and the Flow-time embedding:

$$\tilde{v}_\theta = v_\theta(X_\tau, \tau, c_t) + \alpha\, g_t\, R_\theta(X_\tau, e_t, \tau), \qquad g_t = \max(\beta_t^{\mathrm{now}}, \beta_t^{\mathrm{hor}}), \quad \alpha = 0.5. \tag{6}$$

The gate concentrates the residual correction on chunks near a predicted transition and leaves the backbone velocity nearly unchanged when both boundary likelihoods are low. The last layer of R_θ is initialized to zero, so a pretrained backbone starts from its original behavior.

For a demonstration chunk A, Gaussian noise ϵ, and $\tau \sim \mathrm{Beta}(1.5, 1)$, training uses $X_\tau = (1-\tau)\epsilon + \tau A$ and the conditional flow-matching loss

$$\mathcal{L}_{\mathrm{act}} = \mathbb{E}_{A,\epsilon,\tau}\left[\|\tilde{v}_\theta(X_\tau, \tau, c_t) - (A - \epsilon)\|^2_{\mathrm{mask}}\right], \tag{7}$$

where the mask removes padded actions at the end of a demonstration. At inference, ten Euler steps integrate $\tilde{v}_\theta$ from Gaussian noise, and r_t is computed once per replan.

3.5 Backbone Integration

The first backbone is an eight-block Transformer action expert (width 768) over frozen SigLIP 2 observation tokens, trained from scratch; with the adapter it has approximately 80M trainable parameters, plus 15M in the estimator (Appendix A.3). The second is the pretrained SmolVLA (approximately 450M parameters), whose vision-language module stays frozen while its action expert is fine-tuned (Shukor et al., 2025). For SmolVLA, the global token is appended to the prefix read by the action expert, and the local embeddings and gated residual act on its action tokens and output velocity (Appendix A.4). The estimator reads frozen SigLIP 2 features, which it shares with the Flow policy and computes separately for SmolVLA.

Training has two stages. The estimator is fitted on teacher targets and frozen; the policy with its adapter is then trained with Equation 7 on cached estimator predictions computed from truncated histories. At each replan, the controller encodes the new observation, predicts z_t, integrates one chunk, and executes its first eight actions (Algorithms 1 and 2). Environment steps therefore have the same meaning for all methods, and the added computation appears only in per-replan latency (Section 4.6).

4 Experiments

We evaluate LaDyP on two backbones across the four LIBERO suites and then analyze the signal source, the components, the failure stages, and the execution cost; Appendices D.3 and D.4 vary design parameters and the demonstration budget.

Table 1: Success rate (%) on the four LIBERO suites. Published rows are copied from Kim et al. (2024) (Diffusion Policy, Octo, OpenVLA), Shukor et al. (2025), and Kim et al. (2025) (π_0, OpenVLA-OFT) and use their own protocols. Rows in each backbone block share data, budget, and 1,500 paired trials per suite; values are seed means with seed standard deviations. Params count the policy including frozen VLM modules but not the SigLIP 2 encoder; the estimator is listed separately.

Method	Params	Spatial	Object	Goal	Long	Mean
Published results						
Diffusion Policy	-	78.3	92.5	68.3	50.5	72.4
Octo	0.09B	78.9	85.7	84.6	51.1	75.1
OpenVLA	7B	84.7	88.4	79.2	53.7	76.5
SmolVLA	0.45B	90.0	96.0	92.0	71.0	87.3
π_0	3.3B	96.8	98.8	95.8	85.2	94.2
OpenVLA-OFT	7B	97.6	98.4	97.9	94.5	97.1
Flow policy, trained from scratch						
Base	78M	76.9 $\pm$2.4	85.5 $\pm$4.0	75.7 $\pm$1.5	49.2 $\pm$2.0	71.8 $\pm$0.6
BiggerBase	116M	79.7 $\pm$1.3	87.2 $\pm$0.9	76.8 $\pm$2.1	49.2 $\pm$2.4	73.2 $\pm$1.5
NullAdapter	80M	79.1 $\pm$0.4	86.0 $\pm$0.7	74.1 $\pm$2.0	51.1 $\pm$2.7	72.6 $\pm$1.2
Clock	80M	80.1 $\pm$2.2	87.5 $\pm$0.3	74.4 $\pm$1.9	52.5 $\pm$2.0	73.6 $\pm$0.6
LaDyP	80M+15M	81.3 $\pm$0.5	88.1 $\pm$2.4	75.9 $\pm$2.5	57.5 $\pm$2.2	75.7 $\pm$1.2
SmolVLA, pretrained and fine-tuned						
Base	0.45B	89.9 $\pm$1.8	95.5 $\pm$1.2	91.5 $\pm$1.2	71.8 $\pm$1.6	87.2 $\pm$1.0
NullAdapter	0.45B	89.9 $\pm$1.3	95.2 $\pm$0.5	90.3 $\pm$1.6	72.6 $\pm$1.2	87.0 $\pm$0.6
Clock	0.45B	89.9 $\pm$3.0	95.2 $\pm$0.6	90.7 $\pm$1.0	72.2 $\pm$0.2	87.0 $\pm$0.9
LaDyP	0.45B+15M	90.6 $\pm$0.9	95.4 $\pm$0.5	91.6 $\pm$0.9	78.7 $\pm$0.1	89.1 $\pm$0.4

4.1 Experimental Setup

Benchmark and Protocol. LIBERO has four suites of ten tasks: Spatial varies object layouts, Object varies the manipulated object, Goal varies the instruction within one scene, and Long (LIBERO-10) contains long-horizon instructions, nine of which combine two sub-goals (Liu et al., 2023). Each backbone is trained per seed on all four suites with 10% of demonstrations held out; Flow checkpoints are selected by validation loss, and SmolVLA uses the final checkpoint. Every task is evaluated from the benchmark's 50 initial states for three training seeds, giving 1,500 trials per suite and method, with step limits of 220, 280, 300, and 520 for Spatial, Object, Goal, and Long and no retries or interventions. Trials are paired across methods by task, initial state, seed index, and sampling key (Appendix C).

Controls and Statistics. All variants of a backbone share demonstrations, observations, update budget, and checkpoint rule. *Base* is the unmodified backbone. *NullAdapter* adds the full adapter with constant signals equal to their training-set means, so all conditioning paths are active without temporal information. *Clock* feeds elapsed time over the step limit into the progress channel of the same adapter. *BiggerBase* gives the Flow policy twelve instead of eight blocks (116M parameters), more than Base, adapter, and estimator combined. Differences carry 95% cluster-bootstrap intervals that resample initial states within each task and keep the three seeds of an initial state together; we do not perform hypothesis tests.

4.2 Main Results

On LIBERO-Long, LaDyP increases success from 49.2% to 57.5% for the Flow policy and from 71.8% to 78.7% for SmolVLA (Table 1), with paired intervals of [5.7, 10.9] and [4.7, 9.2] pp and a positive difference in every training seed (Appendix D.2). Capacity does not account for the gain: BiggerBase reaches 49.2%, NullAdapter 51.1% and 72.6%, and Clock 52.5% and 72.2%. On the short suites the Flow policy gains 4.4, 2.6, and 0.2 pp on Spatial, Object, and Goal, and the SmolVLA differences lie between -0.1 and 0.7 pp with intervals that include zero. Part of the Flow policy's short-suite gain does not require phase information: on Spatial, NullAdapter, BiggerBase, and Clock reach 79.1%, 79.7%, and 80.1% against 76.9% for Base. The equal-weight mean increases by 3.9 pp

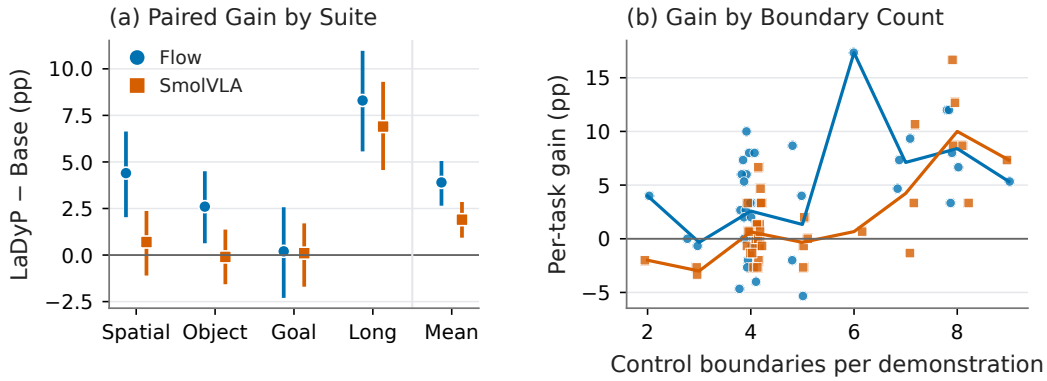

Figure 3: Distribution of the LaDyP gain. (a) Paired LaDyP−Base difference per suite and for the equal-weight mean, with 95% cluster-bootstrap intervals. (b) Per-task difference against the teacher's median number of control boundaries for all 40 tasks; lines connect means per boundary count, and points are jittered horizontally.

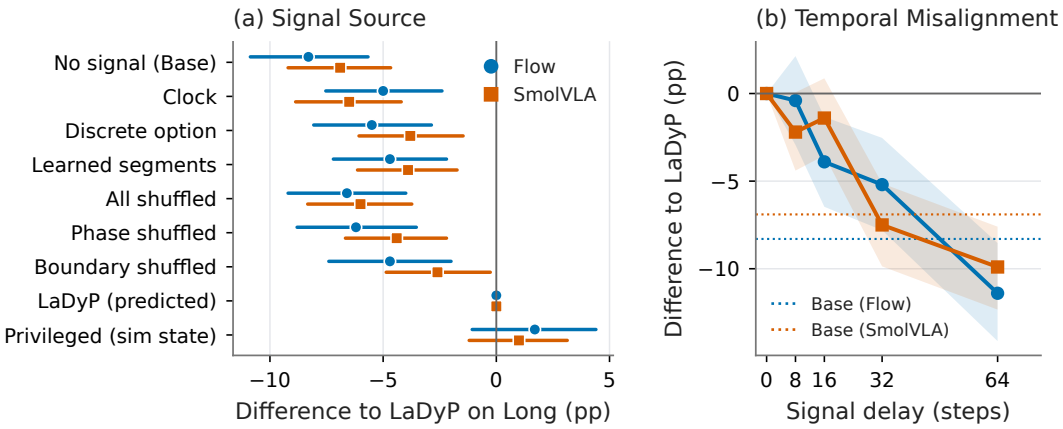

Figure 4: Signal source and alignment on LIBERO-Long. (a) Policies retrained with different temporal signals. (b) The trained policy with signals delayed by the given number of steps; dotted lines mark Base. Values are differences to LaDyP with 95% cluster-bootstrap intervals over 1,500 paired trials.

[2.8, 5.0] and 1.9 pp [1.0, 2.8]. SmolVLA with LaDyP stays below the Long results reported for π_0 and OpenVLA-OFT.

Figure 3b relates the per-task gain to the number of control boundaries in the task's demonstrations. Pooled over 40 tasks, the Spearman correlation is 0.47 for the Flow policy and 0.57 for SmolVLA, but within the Long suite it is −0.03 and 0.54, and within the short suites 0.06 and 0.26. For the Flow policy the pooled correlation thus reflects that Long tasks have more boundaries, and for SmolVLA the within-Long value rests on ten tasks, so boundary count and suite cannot be separated here. Negative point estimates occur on 8 tasks for the Flow policy and 15 for SmolVLA; all but one belong to the short suites (Appendix D.1).

4.3 Signal Source and Alignment

We retrain the policy with alternative temporal signals on LIBERO-Long (Figure 4a; Appendix C.4). The discrete option replaces the soft belief by a one-hot predicted role that selects one of seven action heads and has no progress, boundary, or residual inputs; it represents hard switching as in prior work, not an isolated test of soft conditioning, and stays 5.5 and 3.8 pp below LaDyP. Segments

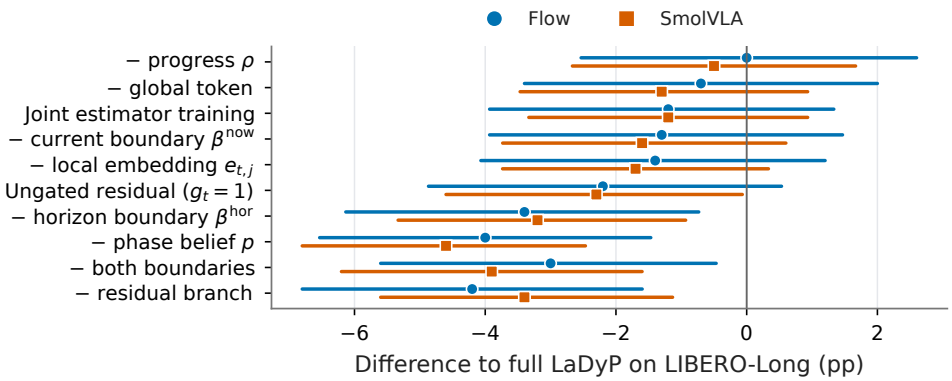

Figure 5: Component ablation on LIBERO-Long. Each row retrains the policy with one change and reports the difference to full LaDyP with a 95% cluster-bootstrap interval over 1,500 paired trials. Removed signals are replaced by their training-set mean, removed modules are deleted, and interaction features in Equation 4 are recomputed.

from K-means clustering of visual and state features stay 4.7 and 3.9 pp below. The shuffled controls draw signals from another training episode of the same task, removing their alignment with the rollout. All shuffled stays within two points of Base on both backbones, and the single-component shuffles lie between All shuffled and LaDyP, with the boundary-shuffled policy closest to LaDyP. A privileged labeler that computes roles and boundaries from simulator state exceeds LaDyP by 1.7 and 1.0 pp with intervals that include zero, so the predicted signals achieve 83% and 88% of the Base-to-privileged improvement.

Figure 4b delays the signals given to a trained policy. One replan of delay (8 steps) changes success by -0.4 and -2.2 pp, 32 steps by -5.2 and -7.5 pp, which places SmolVLA below Base, and at 64 steps both policies are below Base. Since the delayed signals come from the same rollout, the policy relies on their timing.

4.4 Component Ablation

Figure 5 removes one component at a time (pairs of values refer to the Flow policy and SmolVLA). Removing the residual branch, the phase belief, both boundary channels, or only the horizon channel reduces success by 3 to 5 pp on both backbones (for example -4.2 and -3.4 pp without the residual branch), with intervals that exclude zero. Removing only the current channel changes success by -1.3 and -1.6 pp with intervals that include zero and overlap those of the horizon channel. A constant gate $g_t = 1$ changes success by -2.2 and -2.3 pp, less than removing the residual branch, and its Flow interval includes zero. For progress, the global token, the local embedding, and joint estimator training, the intervals include zero on both backbones, so this evaluation does not resolve their contribution.

4.5 Failure Attribution

We assign each failed Long trial to the stage of its last failure event detected from simulator state: Approach (stop before contact), Engage (missed grasp), Transport (drop), Place/Release (misplacement), or Sub-task switch (undone sub-goal or wrong next object; Appendix C.5). Engage and Place/Release are the most frequent Base failure stages on both backbones (Figure 6). LaDyP reduces Engage, Place/Release, and Sub-task-switch failures from 514 to 356 for the Flow policy and from 276 to 193 for SmolVLA (31% and 30% fewer), whereas Approach and Transport failures combined change from 210 to 232 and from 115 to 110, and stalls from 38 to 50 and from 32 to 16. Trials that pass one stage can fail at a later one (typical cases in Appendix D.6).

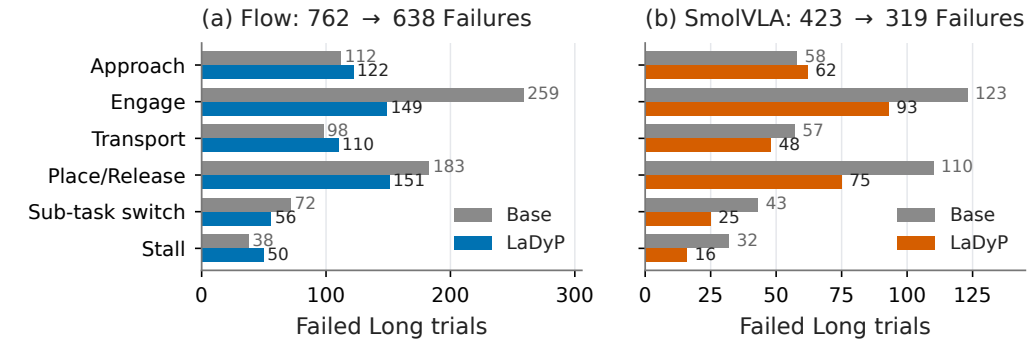

Figure 6: Failed LIBERO-Long trials (out of 1,500) by the stage of the last failure event, detected from simulator state; *Stall* marks trials without a failure event that do not finish within 520 steps.

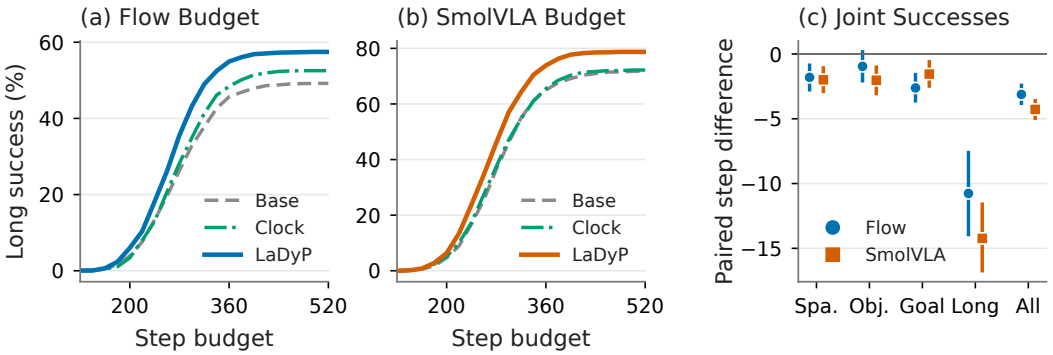

Figure 7: Execution steps. (a) and (b) Percentage of all 1,500 Long trials completed within each step budget. (c) Mean paired step difference on jointly solved trials with 95% cluster-bootstrap intervals; negative values favor LaDyP.

4.6 Execution Steps and Latency

On LIBERO-Long, where budgeted completion counts every trial, LaDyP leads Base by 6.5 and 9.1 pp already at 260 steps, half of the limit (Figure 7a,b). On the 585 and 958 Long trials solved by both methods, it finishes 10.8 and 14.2 steps earlier on average, and by at most three steps on the short suites, where the Flow policy's Object interval includes zero (Figure 7c; Appendix D.5).

The added computation runs once per replan: on one RTX 4090 with batch size one and bfloat16, a replan takes 21.3 versus 24.6 ms for the Flow policy (+15%) and 51.1 versus 64.2 ms for SmolVLA (+26%); most of the SmolVLA overhead is the estimator's separate SigLIP 2 encoder. Each replan precedes eight actions, a 400 ms window at 20 Hz.

5 Conclusion and Limitations

LaDyP conditions action chunks on causally predicted phase, progress, and boundary signals. On LIBERO it improves Long success by 8.3 pp for a Flow policy trained from scratch and 6.9 pp for SmolVLA; the gain concentrates on the suite with the most control boundaries, depends on signal timing, and coincides with fewer failures at engagement, release, and sub-task switches.

All results are simulated on one benchmark family and do not establish performance on physical robots, under visual or dynamics shift, or on unseen instructions. The teacher relies on gripper events and rule-based thresholds, so tasks without gripper events may receive weaker boundary targets. The backbones have at most 0.45B parameters, the analyses rest on three seeds and one automatic failure labeler, and LaDyP adds 26% per-replan latency to SmolVLA.

AI USE STATEMENT

AI tools assisted with literature search, manuscript organization and writing edits, discussion of experimental design, and code for figures and analysis. The authors are responsible for the experiments, numerical results, references, analyses, and final manuscript.

ETHICS STATEMENT

We evaluate in simulated manipulation benchmarks using public demonstration data and conduct human-reference annotation of task phases in simulated trajectories. No physical robot deployment is reported. A phase-conditioned policy can still make incorrect or delayed transitions, and simulation performance does not establish safe physical deployment, which requires evaluation of control limits, sensing conditions, and failure handling.

REPRODUCIBILITY STATEMENT

Appendix A specifies the data, architectures, parameter counts, optimization settings, hyperparameter selection, and training and inference procedures for both backbones; Appendix B gives the teacher construction with all thresholds, the parser prompt, and the estimator objective; Appendix C defines tasks, trial pairing, step limits, control conditions, failure labels, and the bootstrap procedure.

REFERENCES

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. HYDRA: Hybrid Robot Actions for Imitation Learning, 2023. URL https://arxiv.org/abs/2306.17237.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quan Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. RT-H: Action Hierarchies using Language. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.049. URL https://www.roboticsproceedings.org/rss20/p049.html.

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*, Los Angeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-Time Execution of Action Chunking Flow Policies, 2025b. URL https://arxiv.org/abs/2506.07339.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control, 2023. URL https://arxiv.org/abs/2307.15818.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal Cycle-Consistency Learning, 2019. URL https://arxiv.org/abs/1904.07846.

Yiguo Fan, Pengxiang Ding, Shuanghao Bai, Xinyang Tong, 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, 2025. URL https://arxiv.org/abs/2508.19958.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, 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, 2024. URL https://arxiv.org/abs/2406.09246.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success, 2025. URL https://arxiv.org/abs/2502.19645.

Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. CompILE: Compositional Imitation Learning and Execution, 2019. URL https://arxiv.org/abs/1812.01483.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal Loss for Dense Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Lin_Focal_Loss_for_ICCV_2017_paper.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling, 2023. URL https://arxiv.org/abs/2210.02747.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning, 2023. URL https://arxiv.org/abs/2306.03310.

Yuejiang Liu, Jubayer Ibn Hamid, Annie Xie, Yoonho Lee, Maximilian Du, and Chelsea Finn. Bidirectional Decoding: Improving Action Chunking via Guided Test-Time Sampling, 2024. URL https://arxiv.org/abs/2408.17355.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Yecheng Jason Ma, Joey Hejna, Ayzaan Wahid, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, Jonathan Tompson, Osbert Bastani, Dinesh Jayaraman, Wenhao Yu, Tingnan Zhang, Dorsa Sadigh, and Fei Xia. Vision Language Models are In-Context Value Learners, 2024. URL https://arxiv.org/abs/2411.04549.

Oier Mees, Lukas Hermann, and Wolfram Burgard. What Matters in Language Conditioned Robotic Imitation Learning over Unstructured Data, 2022. URL https://arxiv.org/abs/2204.06252.

Junnan Nie, Jiayi Li, Jiachen Zhang, Junyi Lao, Chenghao Liu, Tianle Zhang, Liang Lin, and Songfang Huang. PACE: Phase-Aware Chunk Execution for Robot Policies with Action Chunking, 2026. URL https://arxiv.org/abs/2606.00537.

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, 2024. URL https://arxiv.org/abs/2405.12213.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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, 2025. URL https://arxiv.org/abs/2504.16054.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-Contrastive Networks: Self-Supervised Learning from Video, 2018. URL https://arxiv.org/abs/1704.06888.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi Robot: Open-Ended Instruction Following with Hierarchical Vision-Language-Action Models, 2025. URL https://arxiv.org/abs/2502.19417.

Kyriacos Shiarlis, Markus Wulfmeier, Sasha Salter, Shimon Whiteson, and Ingmar Posner. TACO: Learning Task Decomposition via Temporal Alignment for Control, 2018. URL https://arxiv.org/abs/1803.01840.

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, 2025. URL https://arxiv.org/abs/2506.01844.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual Vision-Language Encoders with Improved Semantic Understanding, Localization, and Dense Features, 2025. URL https://arxiv.org/abs/2502.14786.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Yuzhang Shang, Ramana Rao Kompella, and Gaowen Liu. Real-Time Robot Execution with Masked Action Chunking, 2026. URL https://arxiv.org/abs/2601.20130.

Haoming Xu, Lei Lei, Jie Gu, Chu Tang, Jingmin Chen, and Ruiqi Wang. Move-Then-Operate: Behavioral Phasing for Human-Like Robotic Manipulation, 2026. URL https://arxiv.org/abs/2604.23620.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. XIRL: Cross-embodiment Inverse Reinforcement Learning, 2021. URL https://arxiv.org/abs/2106.03911.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Ming-Yu Liu, Donglai Xiang, Gordon Wetzstein, and Tsung-Yi Lin. CoT-VLA: Visual Chain-of-Thought Reasoning for Vision-Language-Action Models, 2025. URL https://arxiv.org/abs/2503.22020.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware, 2023. URL https://arxiv.org/abs/2304.13705.

Ruijie Zheng, Ching-An Cheng, Hal Daumé III, Furong Huang, and Andrey Kolobov. PRISE: LLM-Style Sequence Compression for Learning Temporal Action Abstractions in Control, 2024. URL https://arxiv.org/abs/2402.10450.