NOT JUST FASTER: LEARNING FROM MIXED-SPEED DEMONSTRATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Demonstrations teach robot policies both task behavior and its timing. Can faster training experience improve task completion without changing deployment settings? Retiming changes how much of a reference trajectory falls within a fixed action window, motivating training on both original and accelerated experience. We investigate this idea through an empirical study of vision-language-action and world-action models across simulated and physical manipulation. Our comparisons distinguish changes in training configuration from changes in a trained policy's speed condition or execution schedule. On CALVIN, mixed-speed configurations improve completion for both model families under the normal condition and unchanged execution settings. Execution sweeps further show that stronger chunk compression can reduce completion and increase control steps among successful trials. These findings support treating demonstration pace as a choice in training supervision, while assessing deployment pace separately through its task-dependent effects on success and execution cost.

1 INTRODUCTION

Robots must complete tasks reliably and within practical time budgets. Demonstration learning has enabled policies to generate action sequences from visual and language context (Zhao et al., 2023; Chi et al., 2023; Kim et al., 2025b), but these policies can also inherit the pauses and conservative motion of their training data (Guo et al., 2025). Accelerating action generation reduces inference latency (Kim et al., 2025a; Yang et al., 2025), and asynchronous prediction overlaps computation with control (Black et al., 2025). At deployment, retiming or compressing predicted action chunks can shorten nominal execution (Kim et al., 2026; Wu et al., 2026). Yet changing execution timing also changes tracking, contact, and opportunities for feedback: fewer commands per prediction need not yield a faster completed task. This motivates a complementary question: *can faster training experience improve what a policy learns, before changing its execution schedule?*

Prior work already establishes that demonstration pace can influence learning. Demonstration acceleration (Guo et al., 2025; Sanchez et al., 2026), variable-speed playback (Masuya et al., 2025), and speed-conditioned training (Jing et al., 2026) provide ways to learn from different temporal scales; TempoVLA also reports improvements at its default pace. We examine these observations through the temporal reach of action supervision. A policy predicts a fixed number of actions, but those labels need not span the same interval of the source demonstration. Under accelerated retiming, an H-action window can reach later reference targets than an original-speed window aligned at the same starting point (Section 3.2). Original demonstrations retain intermediate targets, while accelerated experience can supply targets over a longer span of the reference trajectory.

We hypothesize that mixing these temporal spans can provide complementary supervision to a shared policy. If such training improves task completion, the benefit may already be visible under a normal condition with unchanged execution settings; selecting a fast condition and compressing execution are separate choices. We organize our study around three comparisons: original-only versus mixed-speed training at fixed execution settings, normal versus fast condition profiles within one mixed checkpoint, and execution-horizon changes with the checkpoint and condition held fixed. The first comparison evaluates complete training configurations; fixed execution settings do not imply

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.

identical realized motion speeds. The resulting task outcomes test the value of these configurations, while temporal complementarity remains a possible explanation for the gains.

We carry out this study with the vision-language-action (VLA) policy $\pi_{0.5}$ (Physical Intelligence et al., 2025) and the world-action model (WAM) OpenWAM-α (Wang et al., 2026b). For simulated WAM training, we retime joint references under prescribed velocity limits (Pham & Pham, 2017), replay them to collect the resulting observations and actions, and retain successful episodes. Replay accounts for the interaction induced by a faster reference; filtering excludes candidates that fail to complete the task. The physical VLA study instead uses offline-retimed recorded demonstrations. Mixed policies learn from original and accelerated windows with normal and fast conditions. We compare both model families on CALVIN, examine accelerated-data coverage on BEHAVIOR-1K, and evaluate complete training and deployment configurations on physical insertion tasks. A separate WAM sweep on CALVIN and LIBERO resamples each *complete* H-action prediction into h commands and executes them, allowing us to vary deployment pace without retraining.

Our empirical contribution is a comparison of training pace, conditioned behavior, and execution pace, with three principal findings:

1. **Mixed-speed training can improve completion at fixed execution settings.** On CALVIN, the mixed / normal configuration improves five-task completion for both model families. On BEHAVIOR-1K, broader accelerated-task coverage accompanies a higher task score at approximately matched training volume (Section 4.2).
2. **A fast profile can improve completion while using fewer successful-trial steps.** At the fixed execution settings in our cross-model CALVIN comparison, selecting fast rather than normal from the same mixed checkpoint yields higher completion and fewer control steps among successful sequences for both models (Section 4.2).
3. **Training gains do not prescribe one execution pace.** The WAM sweep reveals task-dependent success–step trade-offs, and aggressive compression can worsen both measures. At matched horizons with recorded step counts, mixed / fast combines higher completion with fewer successful-trial steps on CALVIN, but trades lower completion for fewer steps on LIBERO (Section 4.3).

2 Related Work

Robot action models. Action chunking predicts several controls from one observation, allowing a policy to act between queries (Zhao et al., 2023; Chi et al., 2023). Vision-language-action (VLA) policies ground these predictions in visual and linguistic context (Zitkovich et al., 2023; Kim et al., 2025b; Black et al., 2024; Physical Intelligence et al., 2025); world-action models (WAMs) also predict future visual states (Zhu et al., 2025; Shen et al., 2025; Ye et al., 2026; Wang et al., 2026b). Their predictions extend through time, making both the pace of the training demonstrations and the way each chunk is executed consequential. We study these choices with the existing $\pi_{0.5}$ and OpenWAM-α architectures.

Learning from faster demonstrations. DemoSpeedup accelerates selected demonstration segments using action entropy (Guo et al., 2025). TempoVLA trains a speed-conditioned VLA over a range of retimed demonstrations and reports gains even at its default pace (Jing et al., 2026). VOLT uses vision-language cues to preserve precision-critical segments while accelerating others (Sanchez et al., 2026). In physical settings, Variable-Speed Teaching-Playback records the environmental response to different playback speeds (Masuya et al., 2025), while I2RLC refines accelerated references from measured playback errors (Yamane et al., 2026). Other work changes which points are supervised, how trajectories are resampled, or how speed is chosen over a task (Shi et al., 2023; Yang et al., 2026; Dai et al., 2026; Hu et al., 2026). These studies establish that training pace matters.

Policy computation and execution. Faster action generation addresses policy latency (Kim et al., 2025a; Yang et al., 2025; Xu et al., 2025), and efficient action tokenization reduces training cost (Pertsch et al., 2025). Real-time chunking overlaps prediction and control (Black et al., 2025). Other methods change execution itself: RACE retimes predicted paths (Kim et al., 2026), and Speedup Patch learns a scheduler to downsample chunks from a frozen policy (Wu et al., 2026).

SAIL combines tracking and adaptive speed modulation (Arachchige et al., 2025); VLASH uses future-state-aware asynchronous inference with optional action quantization (Tang et al., 2025). AutoHorizon, PACE, and DEHP adapt how much of a predicted chunk to execute before replanning; they use, respectively, attention-based cues, motion-phase boundaries, and a learned horizon predictor (Wang et al., 2026a; Nie et al., 2026; Zhao et al., 2026). These studies also show that success need not vary monotonically with execution horizon.

TempoVLA's default-pace result makes the training-side question especially concrete, while horizon-selection work shows that deployment scheduling can change success without retraining. We bring these observations into one empirical comparison: fixed-setting results test mixed training configurations across VLA and WAM policies, normal and fast profiles probe one trained checkpoint, and the OpenWAM-α sweep changes execution by resampling the *complete* predicted chunk rather than taking an h-action prefix. We read task completion alongside successful-trial control steps, and approximate physical task time, so the evidence distinguishes a shorter nominal chunk from a faster completed task.

3 Training Experience and Deployment Pace

3.1 Pace Variables and Comparisons

We vary pace at two points: in the experience used for training and in the execution of a learned prediction (Fig. 1). Let $\mathcal{D}_{\mathrm{n}}$ contain original-speed demonstrations $\tau = \{(o_t, a_t)\}_{t=0}^{N_\tau - 1}$ paired with task instruction l, and let $\mathcal{D}_{\mathrm{f}}$ denote the accelerated training domain, constructed as described below. In simulated OpenWAM-α data generation, $s \in \mathcal{S}$ selects a retiming configuration. For a speed-conditioned mixed checkpoint, $c \in \{\mathrm{normal}, \mathrm{fast}\}$ selects a categorical behavior profile. In the OpenWAM-α execution sweep, $h \leq H$ controls how many commands execute a resampled H-action prediction. These variables describe distinct choices; closed-loop rollouts determine their task-level outcomes.

On CALVIN, we first compare the original-only configuration with the normal profile of a mixed-speed checkpoint at fixed execution settings. We then compare normal and fast profiles within that mixed checkpoint. The OpenWAM-α sweep further varies h while holding the checkpoint and condition fixed. A profile includes its task-text prompt and, for the physical VLA, matching normalization; the physical trials compare complete training and deployment configurations.

3.2 Constructing Training Experience

For simulated OpenWAM-α, let q_τ be a source demonstration's joint path. TOPP-RA (Pham & Pham, 2017) retimes that reference path under the joint-velocity limits selected by s, producing $\mathcal{R}_s(q_\tau)$. Replaying the candidate in the simulator yields $\widetilde{\tau}_{\tau,s} = \mathcal{E}_\tau(\mathcal{R}_s(q_\tau))$, with newly recorded observations, states, action traces, and a task outcome. The simulated accelerated domain retains successful replays:

$$\mathcal{D}_{\mathrm{f}}^{\mathrm{WAM}} = \{(\widetilde{\tau}_{\tau,s}, l) : (\tau, l) \in \mathcal{D}_{\mathrm{n}},\ s \in \mathcal{S},\ \mathrm{Success}(\widetilde{\tau}_{\tau,s}) = 1\}. \tag{1}$$

Retiming changes the timing of the reference path; replay records the interaction that actually follows. Successful episodes supply accelerated examples for training, and policy completion is evaluated separately.

The physical $\pi_{0.5}$ tasks use another construction: complete demonstrations are retimed offline, with numerical states and actions interpolated on the new timeline and camera observations drawn from the nearest recorded frames. This supplies accelerated windows from the existing physical corpus. The BEHAVIOR-1K $\pi_{0.5}$ comparison uses TOPP-RA-retimed training data from a subset or all 50 tasks. Further data details appear in Section A.1 and Appendices B and C.

Retiming can change the temporal reach of a training window. Write a source reference as $q_\tau(u)$ and a retimed reference as $q_\tau(\phi(t))$, where ϕ maps retimed time to source time. For H labels sampled at control period Δt, the window spans the following interval in source time:

$$R_\phi(t, H) = \phi\big(t + (H-1)\Delta t\big) - \phi(t), \qquad R_0 = (H-1)\Delta t. \tag{2}$$

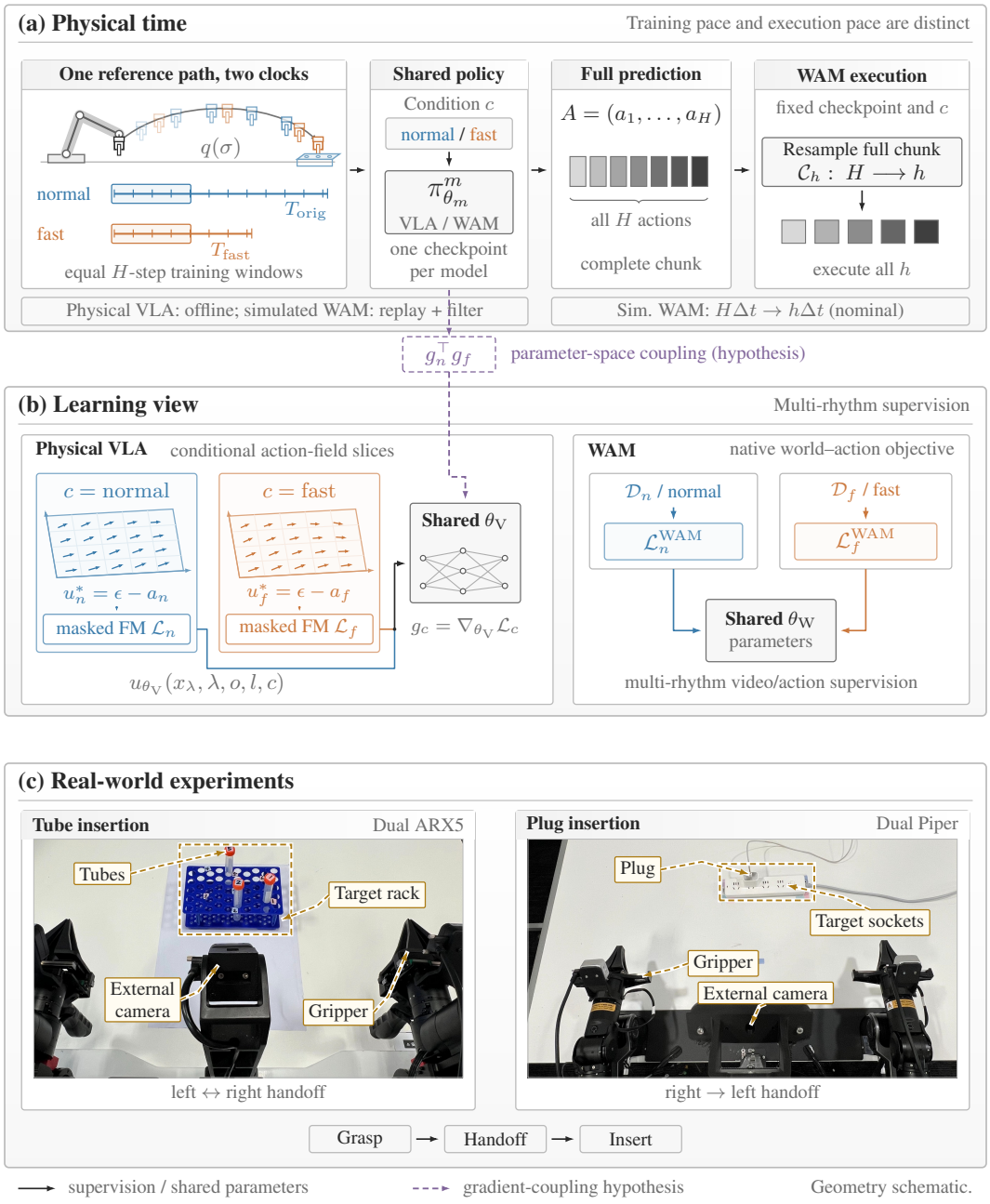


Figure 1: **Training and execution pace in robot action models.** (a) Normal and accelerated experience train a shared policy. Physical VLA uses offline retiming; simulated WAM uses successful replays and separately resamples a complete H-action prediction to h commands at execution. (b) Normal and fast supervision meet in shared parameters. The dashed $g_n^\top g_f$ bridge marks a gradient-coupling hypothesis, not a measured mechanism. (c) Dual-arm tube and plug insertion setups; each task requires grasping, handoff, and insertion. Motion and field geometry are schematic.

When $R_\phi(t, H) > R_0$, a window aligned at the same source starting point reaches a later reference-time target within the same H labels. Co-training then presents both the original intermediate targets and these longer-span targets to a shared policy. Its effect on closed-loop completion is measured in Section 4.

3.3 Mixed-Speed Training and Conditioning

For OpenWAM-α and the physical $\pi_{0.5}$ tasks, mixed training samples original and accelerated windows with equal probability, using task text to identify normal and fast examples. Let ℓ_{n} and ℓ_{f} denote each model's native loss under its domain-specific sampling and conditioning. Their 1:1 training objective is

$$\mathcal{L}(\theta) = \tfrac{1}{2}\,\mathbb{E}_{\mathcal{D}_{\mathrm{n}}}[\ell_{\mathrm{n}}(\theta;\tau,l)] + \tfrac{1}{2}\,\mathbb{E}_{\mathcal{D}_{\mathrm{f}}}[\ell_{\mathrm{f}}(\theta;\tau,l)]. \tag{3}$$

The expectations use each domain's window sampler, so equal probability does not require equal numbers of source episodes. OpenWAM-α shares original-data normalization across conditions; the physical $\pi_{0.5}$ model uses domain-specific statistics and selects the speed prompt and normalization together at deployment (Section A.4). The BEHAVIOR-1K progression compares retimed task-subset and all-task coverage at 1:1 sampling, followed by a 2:1 fast:normal continuation. OpenWAM-α compares original-only and mixed training at equal update budgets. The physical mixed policies continue from original-speed checkpoints; their results reflect this continuation together with the fast deployment profile.

3.4 Deployment Pace and Outcome Measures

For OpenWAM-α, let $A = (a_1, \ldots, a_H)$ be a complete predicted action chunk. The execution operator resamples it into $\mathcal{C}_h(A) = (\widetilde{a}_1, \ldots, \widetilde{a}_h)$ and executes those commands, stopping early when the task completes. It spans the full prediction and retains its final target: $32 \to 16$ compresses all 32 predicted actions into 16 commands rather than taking the first 16. Resampling changes the intermediate commands and the opportunity to replan. With $H = 32$, we evaluate $h \in \{2, 4, 8, 16, 32\}$. At a fixed control period Δt, nominal chunk duration changes from $H\Delta t$ to $h\Delta t$.

The physical co-trained $\pi_{0.5}$ policy deploys its fast profile under asynchronous RTC; the separately configured TOPP-RA baseline retimes each predicted chunk immediately before execution. For LIBERO and CALVIN, we report completion rate p and, where recorded, mean issued control steps $\bar{N}_{\mathrm{succ}} = \mathbb{E}[N \mid \mathrm{Success} = 1]$. These step means describe successful trials and are read alongside completion. BEHAVIOR-1K reports Task Success Score, while physical trials report success and approximate full-task time; benchmark-specific definitions appear in Sections C.1 and 4.1.

4 Experiments

The recorded results first compare VLA and WAM configurations on CALVIN and examine a 50-task BEHAVIOR-1K training-data progression for $\pi_{0.5}$. We then map the OpenWAM-α execution sweep on CALVIN and LIBERO. Physical insertion trials complete the empirical sequence.

4.1 Experimental Setup

We evaluate $\pi_{0.5}$ (Physical Intelligence et al., 2025) and OpenWAM-α (Wang et al., 2026b). OpenWAM-α predicts 32 actions together with future visual states; its normal and fast domains use the same original-data normalization. Within each benchmark, the recorded OpenWAM-α original-only and mixed-speed runs start from the same pretrained model, use training seed 42, and have equal update budgets (85,460 updates on LIBERO; 107,860 on CALVIN). The resulting corpora contain 22,966 original and 17,272 accelerated CALVIN episodes, and 1,693 and 1,428 LIBERO episodes, respectively (Table 5). The mixed run samples the two domains equally and is evaluated with both normal and fast prompts from one checkpoint. The $\pi_{0.5}$ CALVIN comparison also uses a fixed execution setting. OpenWAM-α and physical-task $\pi_{0.5}$ training configurations appear in Sections A.6 and B.2.

OpenWAM-α is evaluated on LIBERO (Liu et al., 2023) with 40 tasks and 50 trials per task (2,000 trials per condition), and on CALVIN (Mees et al., 2022) with the same 1,000 five-instruction sequences and initial states across conditions, using EGL rendering. We report LIBERO success rate (SR), CALVIN five-task completion (SR_5), and CALVIN average completed sequence length. Steps averages issued environment control steps over successful LIBERO trials, excluding 30 initial stabilization steps; Steps_5 averages over CALVIN sequences completing all five subtasks. Thus each reported step mean is conditional on success and may involve different episodes across policies. The

Table 1: Training-pace results. CALVIN uses fixed execution settings (10 steps for $\pi_{0.5}$; $h = 32$ for OpenWAM-α); mixed / normal and mixed / fast share a checkpoint. The 50-task BEHAVIOR Challenge reports Task Success Score: its first three runs have approximately matched training volume (about three original-data epochs), while 2:1 is a later continuation. Ratios denote fast:normal sampling; Steps_5 conditions on five-task completion.

Policy	Training / condition	CALVIN		
		SR_5 (%) ↑	Avg. length ↑	Steps_5 ↓
$\pi_{0.5}$	Original / unprefixed	67.70	3.904	341.60
$\pi_{0.5}$	Mixed / normal	69.00	4.006	338.10
$\pi_{0.5}$	Mixed / fast	77.80	4.393	279.45
OpenWAM-α	Original / unprefixed	69.60	4.022	363.05
OpenWAM-α	Mixed / normal	73.40	4.206	351.09
OpenWAM-α	Mixed / fast	74.40	4.234	340.87
BEHAVIOR-1K (2025 Challenge, 50 tasks)				
Policy	Training data / schedule			TSS (0–1) ↑
$\pi_{0.5}$	Original-only			0.244
$\pi_{0.5}$	Mixed, accelerated task subset (1:1)			0.278
$\pi_{0.5}$	Mixed, accelerated all tasks (1:1)			0.356
$\pi_{0.5}$	Mixed, accelerated all tasks (2:1; continued)			0.419

$\pi_{0.5}$ CALVIN comparison reports SR_5 and average completed length at its recorded fixed execution horizon. Further protocol details and the complete OpenWAM-α sweep appear in Appendix C.

The success endpoints test different events: LIBERO evaluates individual task trials, whereas CALVIN SR_5 requires a five-instruction chain. We therefore compare each condition with its benchmark-specific baseline rather than treating the raw success percentages as directly interchangeable.

The BEHAVIOR-1K (Li et al., 2023) result uses the same 50 tasks and Task Success Score as the 2025 BEHAVIOR Challenge. This score gives partial credit for satisfied goal predicates at the end of an episode; it is not a binary task-success rate. The first three $\pi_{0.5}$ training configurations process approximately the amount of training corresponding to three epochs of the original data. Since adding accelerated data changes the mixture size, we compare training volume rather than exact epoch counts. The fourth result follows an additional continuation stage.

4.2 Training Pace Across Tasks

At fixed execution settings on CALVIN, mixed / normal improves SR_5 over original-only by 1.30 points for $\pi_{0.5}$ and 3.80 points for OpenWAM-α (Table 1). These gains appear under the normal request, before selecting fast behavior from the mixed checkpoint. Selecting its fast condition adds 8.80 and 1.00 points, respectively, reaching 77.80% and 74.40%. Average completed sequence length also rises across the original-only and mixed / fast rows; for OpenWAM-α, Steps_5 falls from 363.05 to 340.87. The comparison across rows therefore has two parts: a change in the complete training configuration at a fixed execution setting, and a change of condition within one trained checkpoint. The latter does not account for the former.

The BEHAVIOR-1K panel extends the training-data comparison to 50 long-horizon household tasks. With approximately matched training volume, mixing TOPP-RA-retimed data from a task subset raises $\pi_{0.5}$'s Task Success Score from 0.244 to 0.278; extending accelerated data across all tasks reaches 0.356, a 0.112 gain over original-only training. A subsequent continuation with fast:normal sampling increased from 1:1 to 2:1 reaches 0.419. The progression shows increasing task score as accelerated experience covers more tasks, with the last row representing a later training stage.

4.3 Execution Pace and Its Interaction with Training

Varying h holds the checkpoint and condition fixed, exposing the effect of the execution choice within each condition (Table 2 and **??**). Resampling the full prediction into fewer commands short-

Table 2: CALVIN five-subtask completion under full-chunk compression ($H = 32$), reported as SR_5 (%). The same checkpoints are evaluated at every h; Mixed / normal and Mixed / fast share one checkpoint. Complete sequence lengths and step counts are reported in **??**.

h	Original	Mixed / normal	Mixed / fast
2	7.9	14.9	14.8
4	33.5	47.6	50.5
8	51.8	61.8	63.4
16	67.7	73.8	73.5
32	69.6	73.4	74.4

ens its nominal duration and changes when the policy next replans. On CALVIN, moving from $h = 8$ to $h = 16$ raises SR_5 for all three conditions, whereas more aggressive compression at $h = 2$ sharply lowers completion. Executing fewer commands per prediction can therefore require more task-level steps even among successful sequences; the nominal ratio H/h is not an end-to-end speedup.

The mixed-training gain is visible across the full CALVIN sweep, not only at the uncompressed setting. Both mixed conditions complete longer sequences on average than Original at every h. At $h = 8, 16, 32$, where all step means are recorded, they also pair higher SR_5 with fewer $Steps_5$ among completed five-task sequences. Mixed / normal exceeds Original in SR_5 at every horizon by 3.8–14.1 percentage points, whereas switching the same mixed checkpoint from normal to fast changes SR_5 by -0.3 to $+2.9$ points. The normal-condition gain is therefore already present across execution schedules; the fast request selects a distinct behavior profile rather than producing the whole gain.

Varying h holds the checkpoint and condition fixed, exposing the effect of an execution choice within each row (Table 2 and **??**). Resampling the full prediction into fewer commands shortens its nominal duration and changes when the policy next replans. Across all six OpenWAM-α rows, moving from $h = 8$ to $h = 16$ raises completion and reduces steps among successful trials. More aggressive compression at $h = 2$ sharply lowers completion on both benchmarks. Executing fewer commands per prediction can thus require more task-level steps even among successful trials; the nominal ratio H/h is not an end-to-end speedup.

On CALVIN, the mixed-training gain is visible across this sweep, not only at the uncompressed setting. Both mixed conditions complete longer sequences on average than original-only training at every h. At $h = 8, 16, 32$, where all step means are recorded, they also pair higher SR_5 with fewer steps among completed five-task sequences. Mixed / normal exceeds original-only SR_5 at every horizon by 3.8–14.1 points, whereas switching the same mixed checkpoint from normal to fast changes it by -0.3 to $+2.9$ points. The normal-condition gain is therefore already present across execution schedules; the fast request selects a distinct behavior profile rather than producing the whole gain.

LIBERO traces a different success–step relationship. At $h = 32$, mixed / normal and original-only have nearly identical success (98.60% and 98.55%), while mixed / fast uses fewer steps among successful trials but completes fewer tasks (132.47 versus 144.80 steps; 96.10% versus 98.55% SR). At $h = 16$, mixed / fast similarly uses fewer steps than original-only (129.04 versus 144.39) with lower success (96.45% versus 99.25%). CALVIN instead pairs higher SR_5 and fewer $Steps_5$ for mixed / fast at the same $h = 16$ (73.5% versus 67.7%; 355.80 versus 385.11). These within-benchmark comparisons show why success and cost must be read together: a learned gain does not prescribe one execution pace across tasks. The accelerated training domain pools several retiming settings; individual-speed layouts remain in **??**.

4.4 Real-World VLA Results

Following the real-world task design of RoboDojo (Chen et al., 2026), we evaluate separate $\pi_{0.5}$ policies on dual-arm plug and tube insertion. In each task, one arm grasps an object, hands it to the other arm, and the receiving arm inserts it into a target on the opposite side. Plug insertion uses dual Piper arms at 30 Hz, and tube insertion uses dual ARX5 arms at 50 Hz; both use three RGB views

Table 3: Real-world $\pi_{0.5}$ results on two RoboDojo-style tasks. Co-trained / fast uses original and offline-retimed training data; TOPP-RA uses a separate model configuration. Times are coarse means across repeated trials. Deployment details are in Table 8.

Task	Setting	Success	SR (%) ↑	Approx. time (s) ↓
Plug	Original	30/40	75.0	24
	Deployment TOPP-RA	34/40	85.0	21
	Co-trained / fast	32/40	80.0	18
Tube	Original	27/35	77.1	18
	Deployment TOPP-RA	17/35	48.6	14
	Co-trained / fast	29/35	82.9	15

Table 4: Per-position real-world $\pi_{0.5}$ results. Times are approximate means across repeated full-task trials.

Task	Setting	P1	P2	P3	P4	Total	SR	Time (s)
Plug	Original	7/10	6/10	8/10	9/10	30/40	75.0%	24
	Deployment TOPP-RA	9/10	8/10	10/10	7/10	34/40	85.0%	21
	Co-trained / fast	8/10	8/10	9/10	7/10	32/40	80.0%	18

Task	Setting	P1	P2	P3	P4	P5	P6	P7	Total	SR	Time (s)
Tube	Original	3/5	5/5	4/5	3/5	4/5	5/5	3/5	27/35	77.1%	18
	Deployment TOPP-RA	2/5	4/5	3/5	1/5	1/5	2/5	4/5	17/35	48.6%	14
	Co-trained / fast	5/5	5/5	2/5	5/5	4/5	4/5	4/5	29/35	82.9%	15

and a 50-action prediction (Table 8). The three evaluated configurations are the original policy, a co-trained checkpoint deployed with its fast prompt and matching normalization under asynchronous RTC, and an independently configured TOPP-RA baseline that retimes predicted chunks before execution.

On both tasks, co-trained / fast improves observed success over the original policy while shortening approximate task time: from 75.0% to 80.0% and 24 to 18 seconds on plug insertion, and from 77.1% to 82.9% and 18 to 15 seconds on tube insertion. The separate deployment-time TOPP-RA baseline is also faster than the original policy, but reaches 85.0% success on plug insertion and 48.6% on tube insertion. These physical results give an initial task-level example of training and deployment acceleration reaching different success–time points. Per-position results appear in Table 4.

5 Discussion

At fixed execution settings on CALVIN, mixed / normal improves five-task completion for both model families before the fast condition is selected. The OpenWAM-α gain persists across execution horizons, while broader accelerated-task coverage accompanies a higher BEHAVIOR-1K score. These results locate part of the observed benefit in learning, rather than execution alone.

Deployment pace selects a different operating point. Within fixed OpenWAM-α checkpoints, aggressive horizon compression reduces completion. At $h = 16$, mixed / fast uses fewer successful-trial steps but lowers success relative to original-only on LIBERO, while improving both measures on CALVIN. The physical trials likewise place mixed-speed policies and the separately configured TOPP-RA baseline at different success–time points. A useful training pace need not prescribe the best execution pace.

We hypothesize that this distinction begins with the temporal reach of supervision. Within a fixed prediction horizon, original windows retain intermediate targets that may support local adjustment,

while retimed windows can reach later reference targets (Section 3.2). The mixture offers a range of progress spans to one shared policy. The two implementations supply these examples differently: simulated OpenWAM-α replay records new observations and actions after retiming, whereas the physical $\pi_{0.5}$ pipeline pairs offline-retimed future actions with frames from the recorded demonstrations.

For the physical $\pi_{0.5}$ model, this variation enters a conditional flow-matching objective. Given an action window a, noise $\epsilon \sim \mathcal{N}(0, I)$, and interpolation time $\widetilde{t}$, the training target is

$$x_{\widetilde{t}} = \widetilde{t}\epsilon + (1 - \widetilde{t})a, \qquad u^* = \epsilon - a, \tag{4}$$

as detailed in Section A.5. Retiming changes the action window and its observation pairing, presenting different conditioned targets to one learned velocity field. OpenWAM-α receives analogous multi-span supervision through its native video–action objective.

Shared parameters provide a second possible link between the two spans. Let g_{n} and g_{f} be the gradients of the expected normal- and fast-domain losses. Under a small gradient step along $-g_{\mathrm{f}}$ with step size $\eta > 0$, the change in normal-domain loss is $-\eta\langle g_{\mathrm{n}}, g_{\mathrm{f}}\rangle + O(\eta^2)$. If this inner product is positive, learning from one domain locally helps the other as well. This gives a precise route for complementary supervision to become a shared gain, while the actual gradient interaction remains a hypothesis for future measurement. Training-mixture design and deployment scheduling therefore remain distinct choices: the former shapes the policy, and the latter selects how its predictions are used.

6 Conclusion

Mixed-speed configurations improve CALVIN completion at fixed execution settings for both model families, while $\pi_{0.5}$ reaches a higher BEHAVIOR-1K score as accelerated-task coverage broadens. Physical insertion tasks offer an initial real-world signal. The OpenWAM-α sweep shows that deployment pace separately selects a task-dependent success–cost operating point. Speed is thus a source of training supervision as well as a deployment choice.

7 Limitations

The recorded OpenWAM-α results use one training seed; step means condition on success and do not establish wall-clock speedup. Physical task times are approximate because a strict endpoint is difficult to impose, and the deployment-time TOPP-RA baseline uses a separately configured model. The mixed physical policies continue training from original-speed checkpoints; full-corpus physical replay remains future work due to cost. Evaluation on additional simulated household-task suites remains future work. The multi-timescale explanation in Section 5 is a hypothesis, not a measured mechanism.

8 AI Use Statement

LLMs are used for polishing writing.

References

Nadun Ranawaka Arachchige, Zhenyang Chen, Wonsuhk Jung, Woo Chul Shin, Rohan Bansal, Pierre Barroso, Yu Hang He, Yingyan Celine Lin, Benjamin Joffe, Shreyas Kousik, and Danfei Xu. SAIL: Faster-than-Demonstration Execution of Imitation Learning Policies. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 721–749. PMLR, 2025. URL https://proceedings.mlr.press/v305/arachchige25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael 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,

James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. arXiv preprint arXiv:2410.24164, 2024. URL https://arxiv.org/abs/2410.24164.

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

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, et al. RoboDojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026. URL https://arxiv.org/abs/2607.04434.

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 *Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://roboticsproceedings.org/rss19/p026.html.

Mingtong Dai, Guanqi Peng, Yongjie Bai, Feng Yan, Chunjie Chen, Lingbo Liu, Liang Lin, and Xinyu Wu. SkiP: When to Skip and When to Refine for Efficient Robot Manipulation. arXiv preprint arXiv:2605.15536, 2026. URL https://arxiv.org/abs/2605.15536.

Lingxiao Guo, Zhengrong Xue, Zijing Xu, and Huazhe Xu. DemoSpeedup: Accelerating Visuomotor Policies via Entropy-Guided Demonstration Acceleration. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 599–609. PMLR, 2025. URL https://proceedings.mlr.press/v305/guo25a.html.

Qingda Hu, Ziheng Qiu, Jieru Zhao, Zhongxue Gan, and Wenchao Ding. AutoSpeed: Annotation-Free Stage-Adaptive Motion Speed Learning for Robot Manipulation. In *European Conference on Computer Vision*, 2026. URL https://arxiv.org/abs/2607.01051.

Dong Jing, Jingchen Nie, Tianqi Zhang, Jiaqi Liu, Huaxiu Yao, Zhiwu Lu, and Mingyu Ding. TempoVLA: Learning Speed-Controllable Vision-Language-Action Policies. arXiv preprint arXiv:2606.06491, 2026. URL https://arxiv.org/abs/2606.06491.

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*, 2025a. doi: 10.15607/RSS.2025.XXI.017. URL https://www.roboticsproceedings.org/rss21/p017.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Sunwoo Kim, Jeongjun Kim, and Joseph J. Lim. Time Optimal Execution of Action Chunk Policies Beyond Demonstration Speed. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/fab80bb9d97e9b9ff5c19f91f72838c6-Abstract-Conference.html.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. BEHAVIOR-1K: A Benchmark for Embodied AI with 1,000 Everyday Activities and Realistic Simulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 80–93. PMLR, 2023. URL https://proceedings.mlr.press/v205/li23a.html.

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, 2023. doi: 10.52202/075280-1939. URL https://papers.nips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Nozomu Masuya, Hiroshi Sato, Koki Yamane, Takuya Kusume, Sho Sakaino, and Toshiaki Tsuji. Variable-Speed Teaching-Playback as Real-World Data Augmentation for Imitation Learning. *Advanced Robotics*, 2025. doi: 10.1080/01691864.2025.2497423. URL https://arxiv.org/abs/2412.03252.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022. URL https://arxiv.org/abs/2112.03227.

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. arXiv preprint arXiv:2606.00537, 2026. URL https://arxiv.org/abs/2606.00537.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient Action Tokenization for Vision-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.012. URL https://www.roboticsproceedings.org/rss21/p012.html.

Hung Pham and Quang-Cuong Pham. A New Approach to Time-Optimal Path Parameterization based on Reachability Analysis. arXiv preprint arXiv:1707.07239, 2017. URL https://arxiv.org/abs/1707.07239.

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. arXiv preprint arXiv:2504.16054, 2025. URL https://arxiv.org/abs/2504.16054.

Robert Ramirez Sanchez, Daniel J. Evans, Dylan P. Losey, and Siddarth Jain. VOLT: Vision and Language Trajectory Segmentation for Faster-than-Demonstration Policies. arXiv preprint arXiv:2606.06323, 2026. URL https://arxiv.org/abs/2606.06323.

Yichao Shen, Fangyun Wei, Zhiying Du, Yaobo Liang, Yan Lu, Jiaolong Yang, Nanning Zheng, and Baining Guo. VideoVLA: Video Generators Can Be Generalizable Robot Manipulators. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/89a3b655a8b68ae1c76b768152c9c19d-Abstract-Conference.html.

Lucy Xiaoyang Shi, Archit Sharma, Tony Z. Zhao, and Chelsea Finn. Waypoint-Based Imitation Learning for Robotic Manipulation. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2195–2209. PMLR, 2023. URL https://proceedings.mlr.press/v229/shi23b.html.

Jiaming Tang, Yufei Sun, Yilong Zhao, Shang Yang, Yujun Lin, Zhuoyang Zhang, James Hou, Yao Lu, Zhijian Liu, and Song Han. VLASH: Real-Time VLAs via Future-State-Aware Asynchronous Inference. arXiv preprint arXiv:2512.01031, 2025. URL https://arxiv.org/abs/2512.01031.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Ramana Rao Kompella, and Gaowen Liu. VLA Knows Its Limits: Adaptive Execution Horizons for Robot Policies. In *European Conference on Computer Vision*, 2026a. URL https://arxiv.org/abs/2602.21445.

Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, Yu-Kai Wang, Yuhan Rui, Shenzhe Yao, Zhen Yuan, Zhenhao Shen, Kefei Zhu, Zijie Zhu, Ning Gao, Xiaowei Chi, Guanqi He, Shanghang Zhang, Hao Dong, Lin Shao, and Hang Zhao. OpenWAM: An Open, Modular Exploration Towards Systematic World-Action Model Pretraining. arXiv preprint arXiv:2609.07398, 2026b. URL https://arxiv.org/abs/2609.07398.

Zhichao Wu, Junyin Ye, Zhilong Zhang, Yihao Sun, Haoxin Lin, Jiaheng Luo, Haoxiang Ren, Lei Yuan, and Yang Yu. Speedup Patch: Learning a Plug-and-Play Policy to Accelerate Embodied Manipulation. arXiv preprint arXiv:2603.20658, 2026. URL https://arxiv.org/abs/2603.20658.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-Cache: Efficient Vision-Language-Action Manipulation via Adaptive Token Caching. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5484. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/f062da1973ac9ac61fc6d44dd7fa309f-Abstract-Conference.html.

Koki Yamane, Cristian C. Beltran-Hernandez, Steven Oh, Masashi Hamaya, and Sho Sakaino. Refinement of Accelerated Demonstrations via Incremental Iterative Reference Learning Control for Fast Contact-Rich Imitation Learning. arXiv preprint arXiv:2604.16850, 2026. URL https://arxiv.org/abs/2604.16850.

Licheng Yang, Lingfeng Qian, Fei Zheng, Yonghao He, Wei Sui, Shuangshuang Li, and Hu Su. Improving Robotic Imitation Learning via Trajectory Standardization. arXiv preprint arXiv:2606.22907, 2026. URL https://arxiv.org/abs/2606.22907.

Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. EfficientVLA: Training-Free Acceleration and Compression for Vision-Language-Action Models. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 40891–40914. Curran Associates, Inc., 2025. doi: 10.52202/085713-1365. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/3a2ef31a1e45908901adc0ca853a8faf-Paper-Conference.pdf.

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. URL https://arxiv.org/abs/2602.15922.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems*, 2023. URL https://roboticsproceedings.org/rss19/p016.html.

Yuchi Zhao, Miroslav Bogdanovic, Arjun Sohal, Liyu Tao, Kourosh Darvish, Alán Aspuru-Guzik, Florian Shkurti, and Animesh Garg. Dynamic Execution Horizon Prediction for Chunk-based Robot Policies. arXiv preprint arXiv:2606.11408, 2026. URL https://arxiv.org/abs/2606.11408.

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 *Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.015. URL https://www.roboticsproceedings.org/rss21/p015.html.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander

Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.