

FACT: Failure-Aware Causal Training for World-Action Models

Quanquan Peng*, Yutong Liang*, Rui Yan, Nicklas Hansen, Xiaolong Wang
University of California San Diego

<https://fact-wam.github.io>

Abstract: Recent world-action models (WAMs) show that co-training policies with future prediction can provide physical priors for action generation. Building on the future-prediction ability of video models, many WAMs generate future videos and recover actions with inverse-dynamics models, or use these predicted videos as goal conditions for action generation. In both cases, the world model is trained mostly on successful demonstrations and has little reason to predict the consequences of bad actions. We introduce **FACT**, a causal World-Action Model that predicts future video and task progress conditioned on the executed action. This action-conditioned interface allows failure rollouts to supervise action consequences, turning bad actions into valid future targets rather than being discarded. Failure-aware training makes the progress predictor aware of both successful and failed action outcomes, which can optionally be used to score sampled action candidates at inference. Extensive experiments on simulation and real-world bi-manual manipulation tasks show that FACT outperforms many existing baselines, improves as failure data are incorporated into training, and reduces success-biased future hallucination under bad actions.

Keywords: World-Action Models, Robot Foundation Models, Manipulation

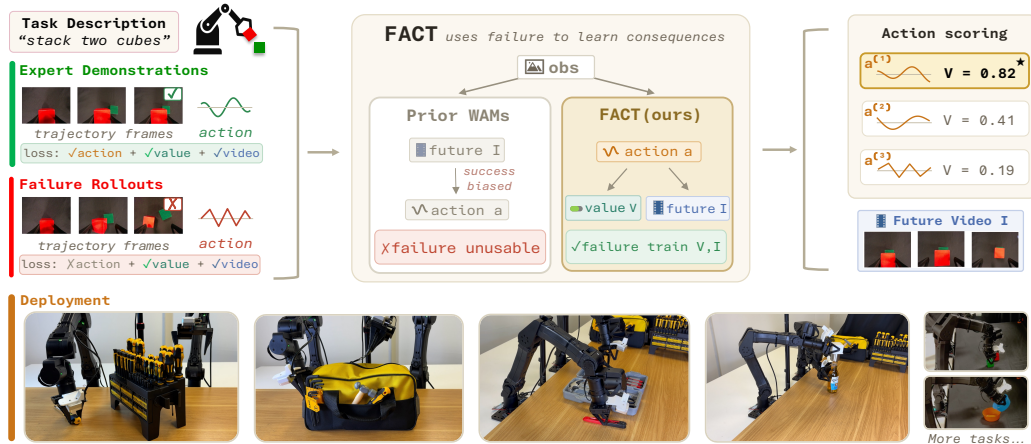


Figure 1: **FACT** first generates an action and then rolls out the resulting future video and task-progress value. Because each future unfolds from the action that produced it, failure rollouts directly supervise the future-prediction branch under their own bad actions, teaching the world model what wrong behavior actually leads to.

1 Introduction

Building general-purpose robot policies is a central goal in robot learning. Vision-language-action (VLA) models have emerged as a promising paradigm toward this aim: given image observations and a language instruction, they learn to output robot actions [1, 2, 3, 4]. A complementary line

* denotes equal contribution.

of work, world-action models (WAMs), augments this paradigm with future visual prediction, using video models or future-prediction objectives to learn control together with how the scene may evolve under robot interaction [5, 6, 7, 8, 9, 10]. By coupling action modeling with predicted future observations, WAMs provide a natural way to bring temporal dynamics and physical priors into robot policy learning.

Existing WAMs commonly use predicted futures as an intermediate signal for choosing actions. One way is to imagine future videos and decode actions with an inverse dynamics model, as in video-first systems [5, 11]. This design benefits from a strong world prior, but action decoding depends on a second-stage network and often requires the future video to be fully denoised before control can proceed. A second line uses predicted future frames or latents as conditions for action prediction [6, 8, 10, 12]. These designs make future prediction useful as an auxiliary signal, yet their future targets are usually expert demonstrations: the model sees plausible futures paired with good actions, but not the consequences of bad actions. As a result, a bad action at test time can still be paired with a success-biased future [13, 14]. The key challenge is not merely adding more data, but using rollouts that fail to complete the desired task *without* turning them into bad demonstrations.

This raises the question: *can we use failure rollouts as consequence supervision without treating failed actions as imitation targets?* In this work, we propose **FACT** (Failure-Aware Causal Training), a causal World-Action Model that separates what to imitate from what to predict. FACT first proposes an action, then predicts the future video and task-progress value conditioned on the executed action. For successful demonstrations, actions, future video, and progress are all supervised. For failure rollouts, the action imitation loss is masked, but the observed failed future and lower progress value remain valid supervision. This turns failure data into action-conditioned consequence supervision and mitigates the tendency to hallucinate only successful futures under incorrect actions.

By learning action-conditioned consequences from both successful demonstrations and failed rollouts, FACT produces a progress estimate that is sensitive to action quality. At inference time, the model can either execute the sampled action directly or optionally use this estimate to rank action candidates. **Overall, our contributions are:**

1. We propose **FACT, a causal World-Action Model with an action-then-future sequence**, enabling co-training on failure trajectories without corrupting action decoding.
2. We introduce a teacher-forced action-conditioned mask that **separates action generation from future prediction**, allowing failed actions to supervise future and value learning without undermining policy.
3. **We validate the model in simulation and real-world benchmarks**, showing improved policy success, reduced success-biased future hallucination under bad actions, and a progress predictor that can optionally support candidate ranking.

2 Related Work

Vision-language-action models for robotic control. Vision-language-action (VLA) models have emerged as a leading paradigm for generalist robot policies, transferring semantic priors from vision-language models to low-level action generation. Early large-scale robot transformers and open-source generalist policies learn language-conditioned manipulation from heterogeneous robot datasets [1, 2, 15, 16], while recent flow- or diffusion-based policies further scale continuous action modeling and open-world generalization [3, 4, 17, 18, 19]. Beyond direct action prediction, a growing line of work incorporates predictive dynamics into VLA policies. BageiVLA [10] interleaves language reasoning, visual forecasting, and action generation; DreamZero [11] and other world-action models use future visual prediction for action decoding, planning, or data generation [5, 6, 7, 8, 9, 20, 21, 22, 23]. These methods show that robotic policies can benefit from modeling not only what action to execute, but also how the world may evolve. In contrast, our work focuses on the causal role of action-conditioned futures: failure rollouts provide structured supervision about undesirable consequences rather than merely serving as low-quality demonstrations.

Data sources for robotic policy learning. The progress of robotic foundation models is closely tied to broader and more diverse training data. One line of work scales real robot demonstrations through shared datasets, distributed collection, and low-cost teleoperation platforms [24, 25, 26, 27, 28, 29]. Another line reduces collection cost by using human-centric interfaces or egocentric human videos to transfer manipulation priors to robots [30, 31, 32, 33, 34, 35]. Simulation provides complementary supervision through scalable task generation and language-conditioned benchmarks [36, 37, 38, 39, 40, 41]. Recently, reinforcement learning and self-improvement have also re-emerged as alternatives to purely offline imitation, improving policies through rollouts, verified rewards, value models, or world-model-based interaction [42, 43, 44]. FACT follows this broader trend of extracting supervision beyond successful demonstrations, but specifically studies how failure data [45, 46, 47, 48] can be converted to action-conditioned future and value supervision for robust policy learning.

3 Method

We propose FACT, a causal World-Action Model that generates actions before predicting their consequences. In this section, we first define the task setting and prediction targets (Sec. 3.1), and then introduce an action-conditioned architecture that separates action imitation from future prediction, allowing failure rollouts to supervise the world branch without becoming imitation targets (Sec. 3.2). Finally, the training and inference strategy closes this loop: failure rollouts provide supervision about action consequences, and the learned progress estimate provides an optional interface for scoring candidate actions (Sec. 3.3).

3.1 Problem Formulation

We formulate language-conditioned robotic manipulation as a sequential decision-making problem. At time t , the robot receives a task instruction ℓ and the current observation $o_t = (I_t^{\text{main}}, I_t^{\text{wristL}}, I_t^{\text{wristR}}, s_t)$, where the images are multi-view RGB observations and $s_t \in \mathbb{R}^{d_s}$ is the robot proprioceptive state. The goal is to choose an action chunk $a_{t:t+H} \in \mathbb{R}^{H \times d}$ that advances the task specified by ℓ . A standard language-conditioned policy models this decision directly as

$$p_\theta(a_{t:t+H} \mid o_t, \ell). \quad (1)$$

Meanwhile, the model is additionally asked to predict what the chosen action will lead to. We therefore augment Eq. (1) with an action-conditioned world branch:

$$p_\theta(o'_{t:t+K}, v_t(a_{t:t+H}) \mid o_t, \ell, a_{t:t+H}), \quad (2)$$

where $o'_{t:t+K}$ denotes future-video observations over the prediction horizon and $v_t(a_{t:t+H}) \in [0, 1]$ is the predicted task progress after executing action chunk $a_{t:t+H}$. To keep this value target comparable across episodes with different lengths, we first define a normalized progress variable from an episode-level return. Let r_k be a progress reward and $G_t = \sum_{k=1}^t r_k$ be the cumulative progress up to time t . The normalized progress is

$$p_t = \frac{G_t}{G_T} \in [0, 1], \quad (3)$$

where $p_t = 0$ denotes the beginning of the task and $p_t = 1$ denotes completion. The learned value head is queried as $V_\theta(o_t, \ell, a_{t:t+H})$ and predicts the action-conditioned progress target.

3.2 Model Architecture

Action-conditioned future prediction. The model must learn future outcomes from failure rollouts without imitating failed actions. To separate these two effects, we introduce a clean ground-truth action condition $a_{t:t+H}^{\text{gt}}$ for the world branch. Under this teacher-forcing design, Eq. (2) becomes

$$p_\theta(o'_{t:t+K}, v_t(a_{t:t+H}^{\text{gt}}) \mid o_t, \ell, a_{t:t+H}^{\text{gt}}). \quad (4)$$

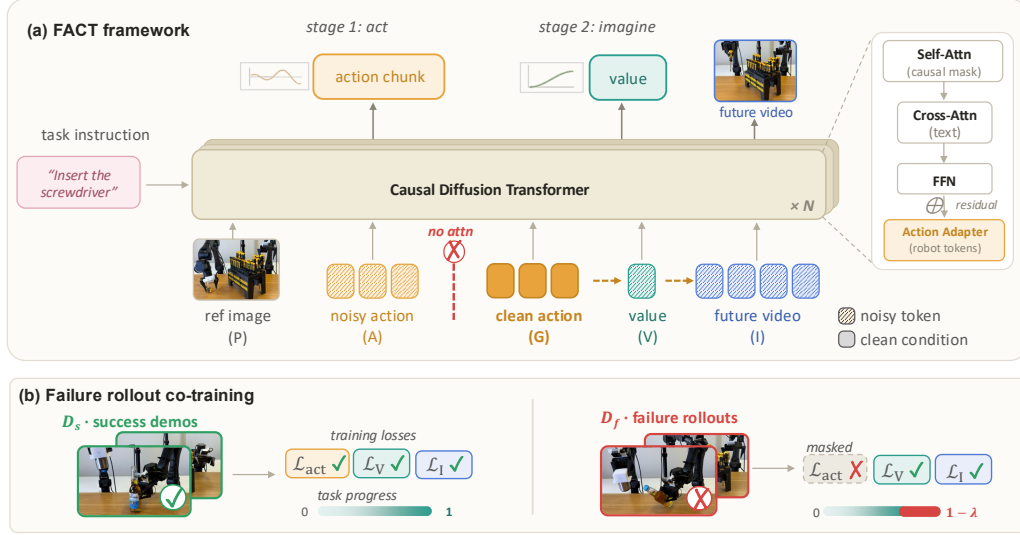


Figure 2: **FACT architecture.** (a) A shared causal diffusion transformer denoises action, value, and future-video tokens; value and future video condition on the clean action slot G , not the noisy A . At inference, Stage 1 denoises an action chunk, which fills G for Stage 2 value and future-video prediction. (b) Success demonstrations supervise all three losses; failure rollouts mask $\mathcal{L}_{\text{act}}$ but keep value and future-video supervision with a lowered progress target—failures teach consequences, not behavior.

This teacher-forced action token conditions future-video and value prediction, while a separate predicted-action segment is used for action denoising. FACT implements this factorization with the token order

$$z = [z_{\text{ref}}^P \parallel z_{\text{pred}}^A \parallel z_{\text{gt}}^G \parallel z_{\text{value}}^V \parallel z_{\text{future}}^I], \quad (5)$$

where P is the observation prefix, A is the noisy predicted-action segment, G is the clean teacher-forced action segment, V is the value segment, and I is the future-video segment. World-side predictions depend on G instead of A : successful demonstrations supervise both action and world prediction, while failure trajectories mask the action imitation loss but keep future-video and value supervision active.

The attention mask in Figure 3 enforces this separation. During training, value and future-video tokens attend to the clean ground-truth action G rather than the noisy predicted action A , while A cannot see G . During inference, no G segment is available: Stage 1 denoises A from $[P, A]$, and Stage 2 denoises value and future-video tokens conditioned on the clean action from Stage 1.

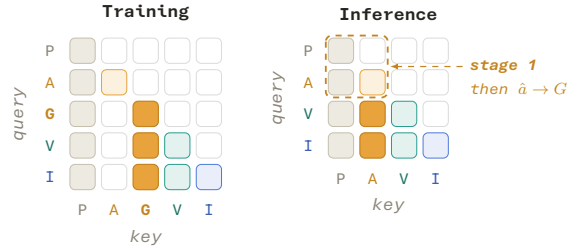


Figure 3: **Training and inference masks.** Rows and columns denote prefix image (P), predicted action (A), ground-truth action (G), value (V), and future video (I); cells mark allowed attention, colored by the attended segment. The dashed block is Stage 1 action denoising; the denoised action then fills G for Stage 2 (Sec. 3.3).

Action adapter on a shared video backbone. FACT uses a shared video diffusion transformer for action, value, and future-video prediction (Figure 2). We attach a lightweight action adapter to robot tokens after the feed-forward network in each transformer block, giving the model action-specific capacity while preserving the pretrained video pathway. Since failure rollouts supervise only future video and value, this shared backbone lets future-outcome losses affect action generation instead of being confined to a separate world expert, as in Mixture-of-Transformers (MoT) designs [5, 7, 10].

3.3 Training and Inference Strategy

Failure-aware value targets. We denote successful demonstrations by $\mathcal{D}_s$ and failure rollouts by $\mathcal{D}_f$. Each episode is annotated with its final outcome and, for failed episodes, the failure onset. We instantiate Eq. (3) as an action-conditioned progress target:

$$v_t(a_{t:t+H}^{\text{gt}}) = \begin{cases} p_{t+H}, & \text{if success,} \\ \text{clip}(p_{t+H} - \lambda_{\text{fail}} \mathbf{1}_{\text{fail}}(t+H), 0, 1), & \text{if fail,} \end{cases} \quad (6)$$

where $\mathbf{1}_{\text{fail}}(t+H)$ indicates whether failure happened by executing $a_{t:t+H}^{\text{gt}}$. In our experiments, we use a uniform progress reward for simplicity, so Eq. (3) reduces to $p_t = t/T$ for an episode of length T . This target preserves temporal progress while lowering the score assigned to action-conditioned futures that enter failure. Thus, under a failed action sequence, the model is trained to predict both the observed failed future and a lower action-conditioned progress target.

Joint denoising losses. We train all predicted modalities with the same flow-matching denoising objective [49]. For a target modality $x \in \{a, v, I\}$, let z_0^x be the clean target token and $z_1^x \sim \mathcal{N}(0, I)$ be Gaussian noise. With interpolation $z_\tau^x = (1 - \tau)z_0^x + \tau z_1^x$, the modality loss is

$$\mathcal{L}_x = \mathbb{E}_{z_0^x, z_1^x, \tau} \left[\|u_\theta^x(z_\tau^x, \tau; z) - (z_1^x - z_0^x)\|_2^2 \right], \quad (7)$$

where u_θ^x is the predicted velocity and z denotes the full token sequence in Eq. (5). Success demonstrations supervise action, value, and future prediction:

$$\mathcal{L}_{\mathcal{D}_s} = w_a \mathcal{L}_a + w_v \mathcal{L}_v + w_I \mathcal{L}_I. \quad (8)$$

For failure rollouts, we remove the action imitation term while retaining value and future-video supervision:

$$\mathcal{L}_{\mathcal{D}_f} = w_v \mathcal{L}_v + w_I \mathcal{L}_I. \quad (9)$$

Thus, failure data teaches the model the consequences and lower action-conditioned progress of failed actions without making those actions policy targets.

Two-stage inference. Inference runs the same transformer in two stages. Stage 1 denoises $[P_{\text{state}}, P_{\text{ref}}, A_{\text{noisy}}]$ for K_{denoise} flow-Euler steps and returns a clean action chunk $\hat{a}_{t:t+H}$ (in Figure 3). If only an action is required, FACT stops here and skips world prediction. When candidate scoring or consequence prediction is requested, Stage 2 places $\hat{a}_{t:t+H}$ into the clean action-conditioning slot and denoises value and, optionally, future-video latents. Since the prefix P is shared across candidates, prefix key-value caching is used in action-only inference.

Optional candidate scoring. Candidate ranking [50] is meaningful in FACT because the progress predictor is trained on action-conditioned failed outcomes. With success-only training, progress prediction is calibrated mainly on the expert action manifold and may assign overly optimistic scores to poor actions. We therefore treat value-guided selection as an optional deployment interface and diagnostic of failure-aware consequence learning, rather than an independent source of supervision. For value-guided inference, Stage 1 samples N action candidates $\{a^{(k)}\}_{k=1}^N$ in parallel. Stage 2 predicts an action-conditioned progress score $\hat{v}^{(k)} = V_\theta(o_t, \ell, a^{(k)})$ for each candidate, and the executed action is selected as

$$a^* = \arg \max_{k \in \{1, \dots, N\}} \hat{v}^{(k)} = \arg \max_{a \in a^{(1:N)}} V_\theta(o_t, \ell, a). \quad (10)$$

This selection rule uses the value head to score the future implied by each candidate action, without training a separate critic.

Implementation details. We initialize the model weights from WAN2.2-5B [51], which serves as the video diffusion backbone, and train with AdamW [52]. The learning rates are 2×10^{-4} for the action FFN and 2×10^{-5} for the WAN backbone; loss weights are $w_a = 20$ and $w_v = w_I = 1$. The action chunk length is $H = 48$ and $\lambda_{\text{fail}} = 1$. Future-video supervision uses the current frame plus four future offsets, corresponding to $[0, H/4, H/2, 3H/4, H]$. Unless otherwise specified, inference uses 20 flow-Euler denoising steps. Appendix A summarizes the failure-aware co-training loop, rollout failure collection, and two-stage inference procedure in pseudocode.

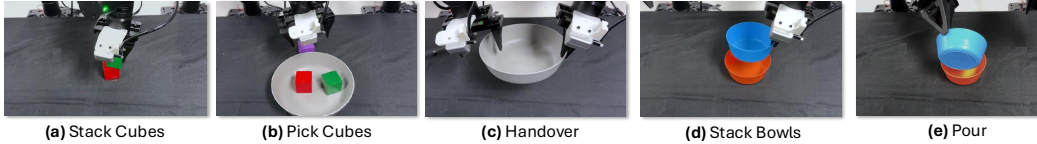


Figure 4: **Real-world task overview.** We show intermediate frame results for five tasks.

4 Experiments

In this section, we evaluate FACT through benchmark comparisons, controlled variants, and diagnostic analyses. We first describe the experimental setup and compare our model with robot foundation policies and recent WAMs in simulation (Sec. 4.1) and real-world benchmarks (Sec. 4.2). We further analyze the model variants in Sec. 4.3, focusing on how each design choice affects the overall results. Our experiments are organized around three key questions:

1. How does FACT compare with existing WAMs and robot foundation policies across simulation and real-world settings?
2. Does action-conditioned future prediction benefit action generation?
3. Does incorporating failure data improve training and reduce the tendency to hallucinate success-biased futures?

Experimental Setup We evaluate FACT in both simulation and real-world bimanual manipulation. Across settings, failure data are mainly collected from model rollouts and used as additional consequence supervision. We compare against representative robot foundation policies and recent WAM baselines. Additional real-world task details are provided in Appendices B and C.

4.1 Simulation Results on RoboTwin

In simulation, we train on 50 RoboTwin tasks and add about 1.3K rollout failures. Table 1 shows that video co-training improves FACT from 81.8% to 85.6% average success, and failure co-training further improves it to 87.5%. This brings FACT close to Motus on this benchmark (87.5% vs. 87.8%), while running roughly $3\times$ faster at deployment (see Appendix E). Appendix D reports the per-task RoboTwin results over all 50 tasks.

Table 1: **RoboTwin** simulation results.

Method	Clean	Rand.	Average
<i>External baselines</i>			
π_0 [3]	65.9	58.4	62.2
X-VLA [17]	72.9	72.8	72.9
$\pi_{0.5}$ [4]	82.7	76.8	79.8
Gigaworld-Policy [9]	87.0	85.0	86.0
Motus [7]	88.7	87.0	87.8
<i>FACT variants</i>			
Ours	86.3	84.9	85.6
Ours w/ failure	<u>88.4</u>	<u>86.6</u>	<u>87.5</u>
<i>Ablations</i>			
Ours w/o video co-train	82.5	81.0	81.8

4.2 Real-World Results

We evaluate five seen tasks (Fig. 4) and three held-out unseen variants that change object colors, shapes, and corresponding instructions. Cube-manipulation tasks use 200 expert demonstrations, the remaining seen tasks use 50, and we collect ~ 30 failure rollouts per cube task for co-training. Each cell in Tables 2 and 3 averages 20 trials; rows marked *optional* use $N=4$ candidate scoring, while other variants use single-sample inference.

Table 2: Real-world results on **seen** tasks.

Method	Stack Cubes	Pick Cubes	Hand over	Stack Bowls	Pour	Avg.
<i>External baselines</i>						
Cosmos [6]	5	45	25	35	15	25
π_0 [3]	35	70	40	50	45	48
$\pi_{0.5}$ [4]	75	100	85	80	100	88
Motus [7]	50	70	55	85	60	64
<i>FACT variants</i>						
Ours	70	85	90	80	85	82
Ours w/ failure	75	<u>95</u>	<u>85</u>	<u>95</u>	<u>95</u>	<u>89</u>
Ours w/ failure						
+ scoring (optional)	85	100	85	100	90	92
<i>Ablations</i>						
Ours + scoring	80	80	70	90	75	79
Ours w/o causal mask	50	75	95	85	80	77
Ours w/ failed-action loss	45	55	75	65	75	63
Ours w/o video co-train	60	55	35	85	55	58

Table 3: **Unseen** real-world task results.

Method	Stack Cubes	Pick Cubes	Stack Bowls	Avg.
<i>External baselines</i>				
Cosmos [6]	15	10	0	8
π_0 [3]	30	65	75	57
$\pi_{0.5}$ [4]	65	<u>90</u>	100	85
Motus [7]	55	60	70	62
<i>FACT variants</i>				
Ours	45	75	80	67
Ours w/ failure	<u>60</u>	85	<u>85</u>	77
Ours w/ failure				
+ scoring (optional)	65	95	<u>85</u>	<u>82</u>

On seen tasks, FACT outperforms Motus (82% vs. 64%); failure-aware training raises this from 82% to 89%, and optional scoring further reaches 92%. Notably, scoring alone without failed outcomes does not help (79%), confirming that the value head only becomes useful after consequence training. On unseen variants, failure-aware training raises success from 67% to 77% and optional scoring to 82%, close to $\pi_{0.5}$ at 85% despite $\pi_{0.5}$ benefiting from large-scale robot pretraining that FACT does not use. Appendix E compares these with measured inference latency.

4.3 Ablation Studies

Policy-performance ablations. The controlled variants in Tables 1 and 2 isolate two design choices of FACT. Removing video co-training reduces RoboTwin average success from 85.6% to 81.8% and real-world seen-task success from 82% to 58%, indicating that future prediction acts as an important regularizer for action generation. The causal-mask ablation is trained without failure data, matching the ‘Ours’ setting except that it removes the clean ground-truth action condition G in Figure 3 and jointly denoises action, value, and future-video tokens. Its lower real-world seen-task success, from 82% to 77%, suggests that teacher-forced action conditioning is important for turning future prediction into stronger action generation. When failure rollouts are added but their action imitation loss is not masked, success drops to 63%, confirming that failed actions should supervise consequences rather than action generation.

Failure data reduces future hallucination.

We compare a success-only checkpoint with one co-trained on success demonstrations and real failure rollouts. Figure 5 shows that, under the same bad-action condition, the success-only model still predicts a successful grasp, while failure-aware co-training predicts the observed failed outcome. Table 4 quantifies this effect: failure-aware co-training substantially improves prediction quality on failure-rollout futures while leaving successful-demonstration futures nearly unchanged, indicating that failure data reduces success-biased future hallucination without degrading normal future prediction. The same trend holds under SSIM, which we report in Appendix F.

Table 4: **Future prediction quality (PSNR $\uparrow$).** Future-image prediction on 512 held-out samples, split evenly between success demonstrations and failure rollouts.

	Ours	Ours w/ failure
All	22.82	26.00
Success-rollout	26.12	26.08
Failure-rollout	19.51	25.92

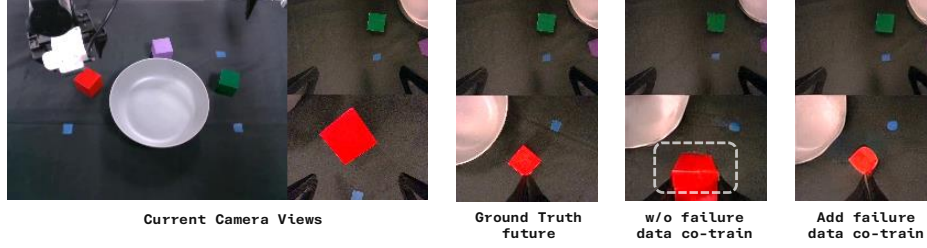


Figure 5: **Effect of failure data on future prediction.** Under the same bad-action condition, the success-only model hallucinates a successful grasp (marked by a white dotted box), while failure-aware co-training predicts the observed failure outcome.

Failure-data scaling. We further probe how the amount of failure data shapes policy learning. On three RoboTwin *clean* tasks, we train FACT with $p \in \{0\%, 50\%, 100\%\}$ of the available failure rollouts mixed into training; at $p=100\%$, failure data accounts for about 45% of the total training set. As shown in Figure 6, average success improves monotonically from 32.7% to 57.3%, indicating that FACT continues to benefit as more failure rollouts are added rather than saturating early.

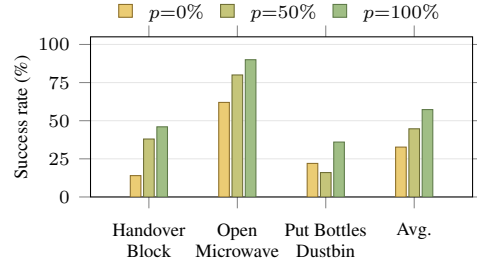


Figure 6: **Failure-data scaling on RoboTwin.** Success rate vs. failure-rollout fraction p .

Value-guided candidate scoring. The action-conditioned task-progress value provides an optional ranking signal after it has been trained on failed outcomes. We sweep the number of candidates on a long-horizon grasping task using the same checkpoint, measuring both relative task completion and inference latency. As shown in Figure 7, task completion improves clearly from $N = 1$ to $N = 4$, while latency increases with the number of candidates. Larger candidate sets provide a smaller gain relative to their additional cost, so optional scoring uses $N = 4$ in our real-world experiments.

Value traces reflect action outcomes. Figure 8 visualizes a Pick Cubes rollout with a missed grasp followed by recovery. The predicted value rises as the task progresses, drops when the grasp fails, and increases again after the policy adjusts and re-grasps. Because the value is predicted after conditioning on the executed action, it can decrease at a later timestep when that action leads to a poor outcome. Appendix G shows additional value traces, and Appendix H further probes this behavior by scoring candidate placements in a controlled Stack Cubes grid.

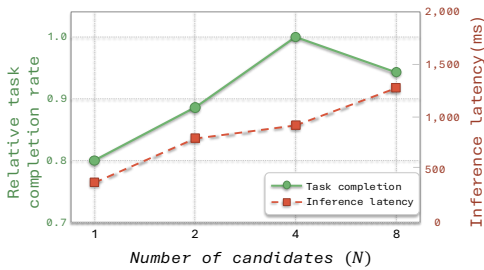


Figure 7: **Choices of different N .** Relative task completion and inference latency as the candidate count N varies on a long-horizon grasping task; $N=4$ balances the two.

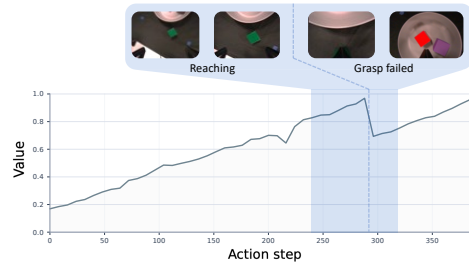


Figure 8: **Task-progress value over a rollout.** Value drops at a missed grasp and recovers after re-grasping, supporting its use as an action-conditioned ranking signal.

5 Limitations

While this work focuses on failure-aware consequence modeling, scaling the same causal training order to broader robot and human-interaction data may further improve the physical plausibility of future prediction and the discriminative power of the value head. Also, future work could replace our value head or augment it with learned progress estimators while keeping the same action-conditioned value interface. Finally, best-of- N selection trades computation for reliability. FACT can run in action-only mode when latency is critical, but value-guided selection requires an additional scoring pass for each candidate batch.

6 Conclusion

We presented FACT, a causal World-Action Model that reverses the usual WAM order by generating actions before predicting future video and task-progress value. A teacher-forcing mask makes the clean executed action the condition for all world-side predictions, allowing failure rollouts to supervise future and value prediction while their action imitation loss is disabled. Across simulation and real-world bimanual manipulation benchmarks, this failure-aware consequence modeling improves policy learning and reduces success-biased future hallucination under bad actions. The learned progress predictor also provides an optional interface for candidate scoring, offering a compute-performance tradeoff at deployment. This action-conditioned view of world modeling provides a natural interface for future training regimes that include online rollouts, DAgger-style corrections, and reinforcement learning from negative experience.

References

- [1] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.
- [2] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [4] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.
- [5] L. Li, Q. Zhang, Y. Luo, S. Yang, R. Wang, F. Han, M. Yu, Z. Gao, N. Xue, X. Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.
- [6] M. J. Kim, Y. Gao, T.-Y. Lin, Y.-C. Lin, Y. Ge, G. Lam, P. Liang, S. Song, M.-Y. Liu, C. Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.
- [7] H. Bi, H. Tan, S. Xie, Z. Wang, S. Huang, H. Liu, R. Zhao, Y. Feng, C. Xiang, Y. Rong, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.
- [8] T. Yuan, Z. Dong, Y. Liu, and H. Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.
- [9] A. Ye, B. Wang, C. Ni, G. Huang, G. Zhao, H. Li, H. Li, J. Li, J. Lv, J. Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.
- [10] Y. Hu, J. Zhang, Y. Luo, Y. Guo, X. Chen, X. Sun, K. Feng, Q. Lu, S. Chen, Y. Zhang, et al. Bagelvla: Enhancing long-horizon manipulation via interleaved vision-language-action generation. *arXiv preprint arXiv:2602.09849*, 2026.
- [11] S. Ye, Y. Ge, K. Zheng, S. Gao, S. Yu, G. Kurian, S. Indupuru, Y. L. Tan, C. Zhu, J. Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.
- [12] P. Zhou, S. Chen, D. Chen, J. Wang, R. Jin, B. Zhu, Y. Pan, S. Gu, K. Wang, S. Nan, et al. τ_0 -wm: A unified video-action world model for robotic manipulation. *arXiv preprint arXiv:2606.01027*, 2026.
- [13] Z. Bai, P. Wang, T. Xiao, T. He, Z. Han, Z. Zhang, and M. Z. Shou. Hallucination of multimodal large language models: A survey. *arXiv preprint arXiv:2404.18930*, 2024.
- [14] V. Rawte, A. Sheth, and A. Das. A survey of hallucination in large foundation models. *arXiv preprint arXiv:2309.05922*, 2023.
- [15] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.
- [16] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

- [17] J. Zheng, J. Li, Z. Wang, D. Liu, X. Kang, Y. Feng, Y. Zheng, J. Zou, Y. Chen, J. Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.
- [18] W. Wu, F. Lu, Y. Wang, S. Yang, S. Liu, F. Wang, Q. Zhu, H. Sun, Y. Wang, S. Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.
- [19] W. Wu, F. Wang, F. Lu, H. Sun, S. Liu, Y. Wang, Y. Yan, Y. Wang, S. Ma, X. Wang, Y. Liu, S. Yang, T. Zhou, K. Zhang, L. Zhou, C. Su, N. Xue, B. Tan, H. Zhang, Y. Zhang, F. Liao, X. Zhu, Y. Shen, and K. Zheng. From foundation to application: Improving vla models in practice. *arXiv preprint arXiv:2607.06403*, 2026.
- [20] S. Gao, W. Liang, K. Zheng, A. Malik, S. Ye, S. Yu, W.-C. Tseng, Y. Dong, K. Mo, C.-H. Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.
- [21] J. Lyu, K. Liu, X. Zhang, H. Liao, Y. Feng, W. Zhu, T. Shen, J. Chen, J. Zhang, Y. Dong, et al. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion. *arXiv preprint arXiv:2602.12215*, 2026.
- [22] G. Jiang, Y. Liang, J. Ye, J.-Y. Huang, C. Jing, R. Duan, P. Abbeel, X. Wang, and X. Zou. Cross-hand latent representation for vision-language-action models. *arXiv preprint arXiv:2603.10158*, 2026.
- [23] I. Liu, A.-C. Cheng, R. Yan, G. Chen, R.-Z. Qiu, X. Zou, S. Yi, H. Yin, X. Wang, and S. Liu. Long-horizon manipulation via trace-conditioned vla planning. *arXiv preprint arXiv:2604.21924*, 2026.
- [24] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.
- [25] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.
- [26] A. O’Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.
- [27] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.
- [28] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
- [29] Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.
- [30] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.
- [31] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song. Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation. *arXiv preprint arXiv:2505.21864*, 2025.

- [32] T. Cheng, K. Chen, L. Chen, L. Zhang, Y. Zhang, Y. Ling, M. Hamad, Z. Bing, F. Wu, K. Sharma, et al. Tacumi: A multi-modal universal manipulation interface for contact-rich tasks. *arXiv preprint arXiv:2601.14550*, 2026.
- [33] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, X. Cheng, R.-Z. Qiu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.
- [34] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu, et al. Egoscale: Scaling dexterous manipulation with diverse egocentric human data. *arXiv preprint arXiv:2602.16710*, 2026.
- [35] R.-Z. Qiu, S. Yang, X. Cheng, C. Chawla, J. Li, T. He, G. Yan, D. J. Yoon, R. Hoque, L. Paulsen, et al. Humanoid policy~ human policy. *arXiv preprint arXiv:2503.13441*, 2025.
- [36] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.
- [37] S. Nasiriany, S. Nasiriany, A. Maddukuri, and Y. Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. *arXiv preprint arXiv:2603.04356*, 2026.
- [38] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023.
- [39] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.
- [40] O. Mees, L. Hermann, E. Rosete-Beas, and W. 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.
- [41] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.
- [42] H. Li, P. Ding, R. Suo, Y. Wang, Z. Ge, D. Zang, K. Yu, M. Sun, H. Zhang, D. Wang, et al. Vla-rft: Vision-language-action reinforcement fine-tuning with verified rewards in world simulators. *arXiv preprint arXiv:2510.00406*, 2025.
- [43] J. Lv, H. Li, J. Li, Y. Nie, F. Kong, Y. Wang, X. Wang, Z. Zhu, C. Ni, Q. Deng, et al. Viva: A video-generative value model for robot reinforcement learning. *arXiv preprint arXiv:2604.08168*, 2026.
- [44] J. Yang, K. Lin, J. Li, W. Zhang, T. Lin, L. Wu, Z. Su, H. Zhao, Y.-Q. Zhang, L. Chen, et al. Rise: Self-improving robot policy with compositional world model. *arXiv preprint arXiv:2602.11075*, 2026.
- [45] Z. Liu, A. Bahety, and S. Song. Reflect: Summarizing robot experiences for failure explanation and correction. *arXiv preprint arXiv:2306.15724*, 2023.
- [46] D. H. Grollman and A. G. Billard. Robot learning from failed demonstrations. *International Journal of Social Robotics*, 4(4):331–342, 2012.
- [47] X. Wang, S. Cheng, S. Wang, X. Zhang, D. Zhang, and W. Zhang. Learning from demonstration with failure awareness for safe robot navigation. *arXiv preprint arXiv:2604.23360*, 2026.

- [48] B. Li, M. Mishra, Y. I. Liu, S. Tao, N. Kumar, A. G. Gray, R. Wickramarachchi, J. Francis, S. Scherer, and T. Silver. Recover, discover, plan: Learning skills and concepts from robot failures. *arXiv preprint arXiv:2606.18328*, 2026.
- [49] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.
- [50] H. Qi, H. Yin, A. Zhu, Y. Du, and H. Yang. Inference-time enhancement of generative robot policies via predictive world modeling. *IEEE Robotics and Automation Letters*, 2026.
- [51] T. Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.
- [52] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [53] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12156–12163. IEEE, 2024.

Appendix

A Training and Inference Algorithms

Algorithms 1–3 summarize the procedure used by FACT. The notation follows Sec. 3.3: successful demonstrations are denoted by $\mathcal{D}_s$, rollout failures by $\mathcal{D}_f$, and the action-conditioned task-progress target is larger for actions that are expected to complete the task.

Algorithm 1 Failure-aware co-training of FACT

Require: successful demonstrations $\mathcal{D}_s$, failure rollouts $\mathcal{D}_f$, model θ

```

1: for each training step do
2:   Sample a minibatch  $B_s \subset \mathcal{D}_s$  and  $B_f \subset \mathcal{D}_f$ 
3:   for each trajectory window  $(o_t, \ell, a_{t:t+H}, o_{t:t+K}) \in B_s \cup B_f$  do
4:     Compute progress target  $v_t(a_{t:t+H})$  using Eq. (6)
5:     Pack tokens  $[P, A, G, V, I]$  as in Eq. (5)
6:     Corrupt predicted action, value, and future-video targets with flow-matching noise
7:     if the window comes from  $\mathcal{D}_f$  then
8:       Set action imitation mask  $m_a \leftarrow 0$ 
9:     else
10:      Set action imitation mask  $m_a \leftarrow 1$ 
11:    end if
12:  end for
13:  Apply the teacher-forcing attention mask from Fig. 3
14:  Update  $\theta$  with  $m_a w_a \mathcal{L}_a + w_v \mathcal{L}_v + w_I \mathcal{L}_I$ 
15: end for
```

Algorithm 2 Rollout failures for co-training

Require: $\mathcal{D}_s$, initial policy π_{θ_0} , rollout budget M

```

1: Initialize  $\mathcal{D}_f \leftarrow \emptyset$ 
2: Train  $\pi_{\theta_0}$  on  $\mathcal{D}_s$ 
3: for task  $\ell$  and rollout  $m = 1, \dots, M$  do
4:    $\tau_m = \{(o_t, a_t)\}_{t=1}^{T_m} \sim \pi_{\theta_0}(\cdot \mid o_t, \ell)$ 
5:   if success( $\tau_m$ ) = 0 then
6:     Annotate failure onset  $t_f$  when available
7:      $\mathcal{D}_f \leftarrow \mathcal{D}_f \cup \{(\tau_m, \ell, t_f)\}$ 
8:   end if
9: end for
10: Continue training on  $\mathcal{D}_s \cup \mathcal{D}_f$  using Algorithm 1
```

Algorithm 3 Two-stage inference with optional candidate scoring

Require: observation o_t , instruction ℓ , number of candidates N

```

1: Encode the observation prefix  $P = (o_t, \ell)$ 
2: Sample  $N$  noisy action chunks  $\{A^{(k)}\}_{k=1}^N$ 
3: for  $k = 1, \dots, N$  in parallel do
4:   Stage 1: denoise  $A^{(k)}$  conditioned on  $P$  to obtain  $\hat{a}_{t:t+H}^{(k)}$ 
5:   if candidate scoring is disabled then
6:     return  $\hat{a}_{t:t+H}^{(1)}$ 
7:   end if
8:   Stage 2: place  $\hat{a}_{t:t+H}^{(k)}$  in the clean action-conditioning slot
9:   Denoise the value token to obtain  $\hat{v}^{(k)} = V_\theta(o_t, \ell, \hat{a}_{t:t+H}^{(k)})$ 
10:  Optionally denoise future-video tokens for consequence visualization
11: end for
12: Select  $k^* = \arg \max_k \hat{v}^{(k)}$ 
13: return  $\hat{a}_{t:t+H}^{(k^*)}$ 
```

B Task Overview

Figure 9 and Figure 10 show the real-world tasks evaluated in Sec. 4.2. The seen set contains five manipulation tasks used for real-world training and evaluation, and the unseen set contains held-out variants with changed object colors, shapes, and instructions.

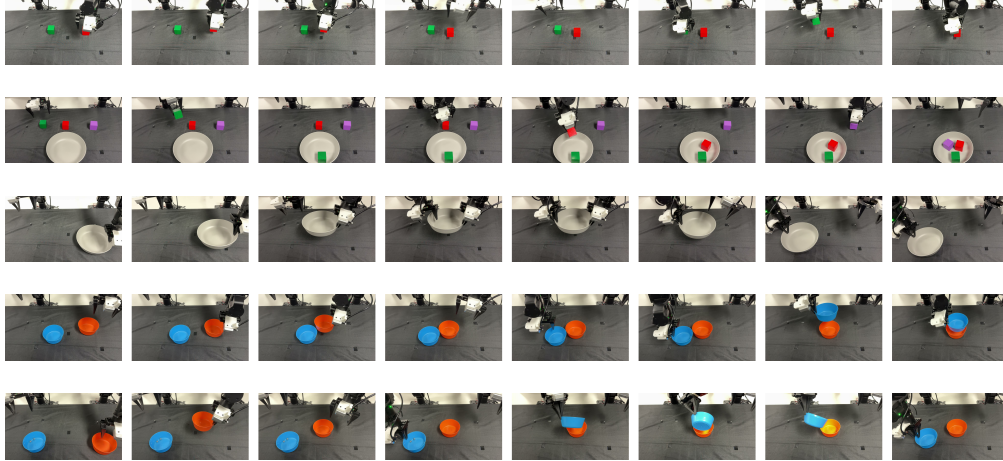


Figure 9: **Seen real-world tasks.** Rollout image sequences for the five seen tasks reported in Table 2, ordered from top to bottom as Stack Cubes, Pick Cubes, Hand-over, Stack Bowls, and Pour.

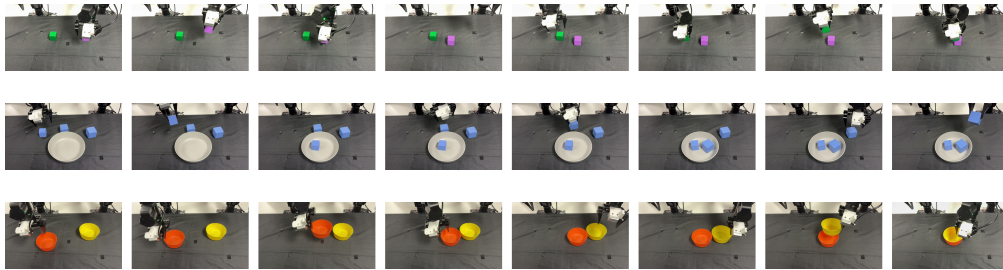


Figure 10: **Unseen real-world tasks.** Held-out task variants used for Table 3, ordered from top to bottom as Stack Cubes, Pick Cubes, and Stack Bowls.

C Real-World Setup and Task Prompts

Figure 11 shows the real-world platform used for the experiments in Sec. 4.2. The setup contains two YAM robot arms, one Intel RealSense D435 camera for the main view, two D405 wrist cameras, and a red GELLO [53] teleoperation device. For multi-view observations, we pack the three camera streams into a single video canvas, preserving the policy observation while keeping the input compatible with the shared video backbone. We also pre-compute the VAE latent for each camera view to accelerate training. The right side of the figure shows the manipulation objects used in the real-world tasks.

All baselines in Tables 2 and 3 are fine-tuned on the same expert success demonstrations as FACT before evaluation. Table 5 lists the language prompts used for the five seen real-world tasks. These prompts correspond to the task names in Table 2.

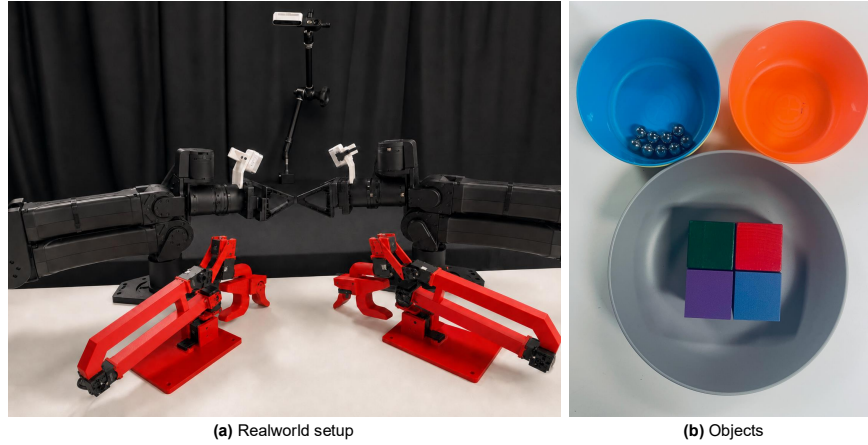


Figure 11: **Real-world hardware setup.** As shown in (a), we use two YAM arms, three RGB camera views, and a GELLO teleoperation device. The objects we used in real-world experiments are shown in (b).

Real-world Task Prompts
Stack Cubes Grasp the red cube and place it at the center of the table, then grasp the green cube and stack it on top of the red cube. Pick Cubes Pick up the red cube, green cube, and purple cube on the table and put them on the gray plate one by one. Hand-over Use the left arm to pick up the gray plate on the table, and hand it over to the right arm and place it on the table. Stack Bowls Grasp the orange bowl and place it at the center of the table, and then grasp the blue bowl and stack it on the orange bowl. Pour Pick the orange bowl and put it at the center of the table, then pick the blue bowl and pour the little silver balls into the orange bowl.

Table 5: **Real-world task prompts.** For real-world experiments, we use the prompts listed above and compute the T5 embeddings for cross-attention.

D Detailed RoboTwin Results

For RoboTwin training, we use a mixture of clean and domain-randomized demonstrations across the 50 tasks. The clean split contains 2,500 demonstrations in total, with 50 demonstrations per task, while the randomized split contains 25,000 demonstrations, with 500 demonstrations per task. The randomized scenes vary visual backgrounds, lighting, etc., providing a robustness test under distribution shift.

E Success Rate and Inference Time

Table 7 compares success rates with measured deployment latency on an RTX PRO 6000. Although Motus has the highest average success in simulation, its video-first inference is substantially slower. FACT keeps the action-first path lightweight: the action-only deployment used by the main policy runs faster than recent WAM baselines while retaining strong simulation and real-world success.

F Future Prediction Metrics

Table 8 reports the full future-image prediction metrics for the ablation in Sec. 4.3, including both SSIM and PSNR. We evaluate on 512 held-out samples, split evenly between successful-demonstration windows and failure-rollout windows, and both models use 20 denoising steps. Consistent with the PSNR results reported in Table 4, failure-aware co-training substantially improves prediction quality on failure-rollout futures under both metrics while leaving successful-demonstration futures nearly unchanged.

G Additional Action-Conditioned Value Traces

Figure 12 provides additional value-trace visualizations for the Stack Bowls and Stack Cubes tasks. Together with Figure 8, these rollouts show that the predicted task-progress value changes with the action-conditioned task outcome across different manipulation skills.

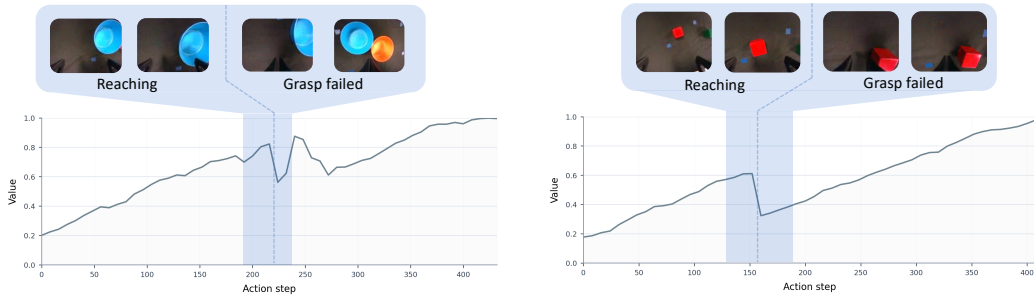


Figure 12: **Additional action-conditioned value traces.** Predicted task-progress values over roll-outs on Stack Bowls (left) and Stack Cubes (right).

H Action-Conditioned Value Heat Map

Figure 13 provides an additional diagnostic for the value head discussed in Sec. 4.3. On the Stack Cubes task, we evaluate the same state with candidate placements over a 3×3 grid. Only the center placement completes the task successfully, shown in (a). The predicted task-progress values are unclipped in this visualization, so failed action candidates can receive negative scores. The model assigns the highest value to the successful center placement and lower values to placements that move the cube to failure positions, supporting the action-conditioned target in Eq. (6).

Simulation Task	X-VLA		Motus		FACT w/o video co-train		FACT		FACT w/ failure	
	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.	Clean	Rand.
<i>Adjust Bottle</i>	100	99	89	93	96	97	99	98	99	98
<i>Beat Block Hammer</i>	92	88	95	88	84	67	75	80	81	80
<i>Blocks Ranking Rgb</i>	83	83	99	97	92	90	96	92	94	93
<i>Blocks Ranking Size</i>	67	74	75	63	52	51	60	50	64	57
<i>Click Alarmclock</i>	99	99	100	100	85	87	89	92	79	89
<i>Click Bell</i>	100	100	100	100	68	89	93	91	80	90
<i>Dump Bin Bigbin</i>	79	77	95	91	93	91	97	95	96	100
<i>Grab Roller</i>	100	100	100	100	100	100	100	100	100	100
<i>Handover Block</i>	73	37	86	73	68	55	65	62	84	72
<i>Handover Mic</i>	0	0	78	63	96	96	99	100	100	99
<i>Hanging Mug</i>	23	27	38	38	35	38	36	45	36	49
<i>Lift Pot</i>	99	100	96	99	97	99	99	100	100	99
<i>Move Can Pot</i>	89	86	34	74	82	82	80	86	80	90
<i>Move Pillbottle Pad</i>	73	71	93	96	94	91	94	91	97	97
<i>Move Playingcard Away</i>	93	98	100	96	100	95	100	97	100	100
<i>Move Stapler Pad</i>	78	73	83	85	40	41	54	50	62	50
<i>Open Laptop</i>	93	100	95	91	97	99	98	97	98	100
<i>Open Microwave</i>	79	71	95	91	87	78	91	95	98	98
<i>Pick Diverse Bottles</i>	58	36	90	91	71	66	65	58	79	67
<i>Pick Dual Bottles</i>	47	36	96	90	88	80	87	74	97	83
<i>Place A2b Left</i>	48	49	88	79	89	89	92	89	95	89
<i>Place A2b Right</i>	36	36	91	87	86	90	90	90	93	93
<i>Place Bread Basket</i>	81	71	91	94	86	76	84	80	85	74
<i>Place Bread Skillet</i>	77	67	86	83	85	79	86	81	84	81
<i>Place Burger Fries</i>	94	94	98	98	95	99	98	97	98	95
<i>Place Can Basket</i>	49	52	81	76	75	67	85	69	87	71
<i>Place Cans Plasticbox</i>	97	98	98	94	99	95	99	97	99	100
<i>Place Container Plate</i>	97	95	98	99	98	96	96	98	100	98
<i>Place Dual Shoes</i>	79	88	93	87	62	65	83	85	85	80
<i>Place Empty Cup</i>	100	98	99	98	97	100	99	100	100	100
<i>Place Fan</i>	80	75	91	87	87	75	87	88	95	87
<i>Place Mouse Pad</i>	70	70	66	68	52	65	69	77	86	82
<i>Place Object Basket</i>	44	39	81	87	76	79	90	81	88	86
<i>Place Object Scale</i>	52	74	88	85	83	74	81	80	84	84
<i>Place Object Stand</i>	86	88	98	97	91	93	96	94	98	94
<i>Place Phone Stand</i>	88	87	87	86	89	92	89	95	90	91
<i>Place Shoe</i>	96	95	99	97	98	95	99	99	98	98
<i>Press Stapler</i>	92	98	93	98	76	74	84	73	82	79
<i>Put Bottles Dustbin</i>	74	77	81	79	63	75	73	81	83	88
<i>Put Object Cabinet</i>	46	48	88	71	79	78	94	85	89	82
<i>Rotate Qrcode</i>	34	33	89	73	79	79	79	82	81	84
<i>Scan Object</i>	14	36	67	66	78	72	87	77	86	80
<i>Shake Bottle Horizontally</i>	100	100	100	98	100	98	100	99	100	98
<i>Shake Bottle</i>	99	100	100	97	99	95	100	99	100	97
<i>Stack Blocks Three</i>	6	10	91	95	83	79	88	91	96	94
<i>Stack Blocks Two</i>	92	87	100	98	97	96	95	97	100	95
<i>Stack Bowls Three</i>	76	86	79	87	74	70	82	75	79	77
<i>Stack Bowls Two</i>	96	93	98	98	97	95	97	93	94	93
<i>Stamp Seal</i>	76	82	93	92	60	69	72	80	78	91
<i>Turn Switch</i>	40	61	84	78	69	48	66	61	61	56
Average	72.88	72.84	88.66	87.02	82.54	80.98	86.34	84.92	<u>88.36</u>	<u>86.56</u>

Table 6: **Detailed RoboTwin results.** Per-task success rates on 50 RoboTwin tasks under clean and randomized evaluation. Each task is evaluated for 100 trials in each split.

Table 7: **Success rate and inference time.** Success rates are averages from Sec. 4; latency is measured on an RTX PRO 6000.

Model (inference time)	Sim	Real seen	Real unseen
$\pi_{0.5}$ (47 ms)	79.8	<u>88</u>	85
π_0 (45 ms)	62.2	48	57
Motus (1220 ms)	87.8	64	62
Cosmos (620 ms)	–	25	8
Ours w/ failure (380 ms)	<u>87.5</u>	89	<u>77</u>

Table 8: **Future prediction quality.** SSIM and PSNR on 512 held-out future-prediction samples, split evenly between successful-demonstration and failure-rollout windows. Both models use 20 denoising steps.

Subset	SSIM $\uparrow$		PSNR $\uparrow$	
	Ours	Ours w/ failure	Ours	Ours w/ failure
All	0.7873	0.8288	22.82	26.00
Success-rollout	0.8285	0.8286	26.12	26.08
Failure-rollout	0.7461	0.8290	19.51	25.92

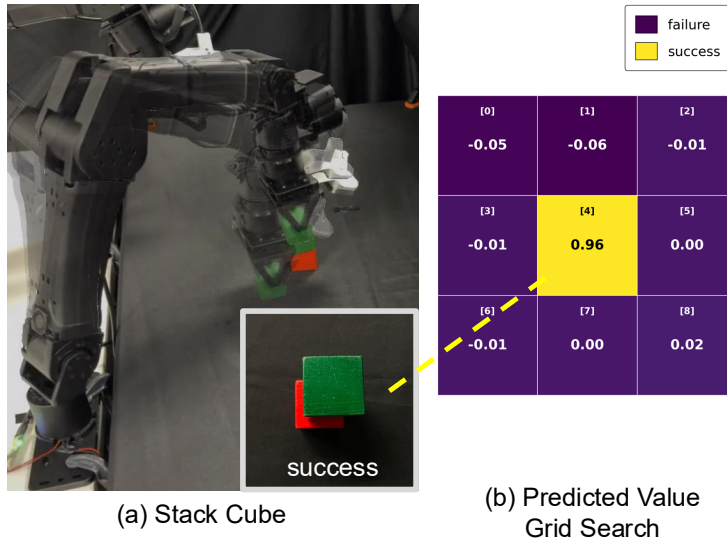


Figure 13: **Action-conditioned value heat map.** On Stack Cubes, the learned value head scores the successful center placement higher than failed placements around it.