HUMAN-IN-THE-WORLD-MODEL FOR SCALABLE ROBOT POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training robot policies through human correction requires repeated physical execution, scene resets, and operator supervision, limiting the scale of corrective data collection. We propose **Human-in-the-World-Model (Hi-WM)**, a robot policy post-training paradigm that collects human corrective supervision inside an action-conditioned world model, without requiring execution on the target robot during collection. A pretrained policy runs in closed loop on model-generated observations, and a human intervenes when the rollout becomes failure-prone. State caching, rollback, and branching allow operators to revisit the same failure state and collect multiple corrective continuations without repeating physical execution. The generated observations and corrective actions are combined with real-world demonstrations to post-train the policy. Across three real-world manipulation tasks involving rigid and deformable objects and two policy backbones, Hi-WM improves success rates by an average of 37.9 percentage points over the base policies and 19.0 percentage points over a baseline that augments training with successful autonomous world-model rollouts. Increasing the amount of virtual corrective data further improves real-world success in the tested range. These results demonstrate that human corrections collected in a learned world model can improve physical robot policies, supporting world models as reusable environments for corrective post-training.

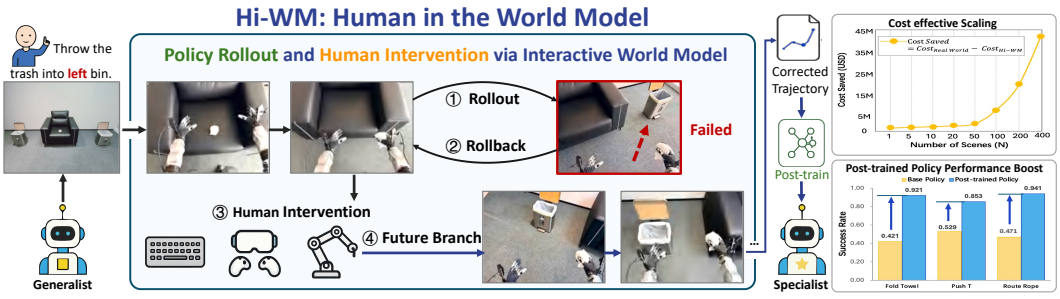

Figure 1: Hi-WM overview. A policy is rolled out in a learned world model. When the rollout becomes failure-prone, a human provides short corrective actions. Cached states support rollback and branching, producing multiple continuations for post-training.

1 INTRODUCTION

Large-scale robot pretraining has produced generalist policies that transfer across tasks, objects, and language instructions. Yet broad pretraining alone is rarely sufficient for deployment. In concrete scenes, policies still fail under contact-rich dynamics, environment-specific variation, and long-horizon error accumulation. In practice, post-training is what turns a generalist policy into a reliably deployable, task- or scene-specialized policy.

The most effective post-training pipelines are still on-policy and corrective: let the current policy act, observe where it breaks, collect targeted supervision on those failure states, and update the policy

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

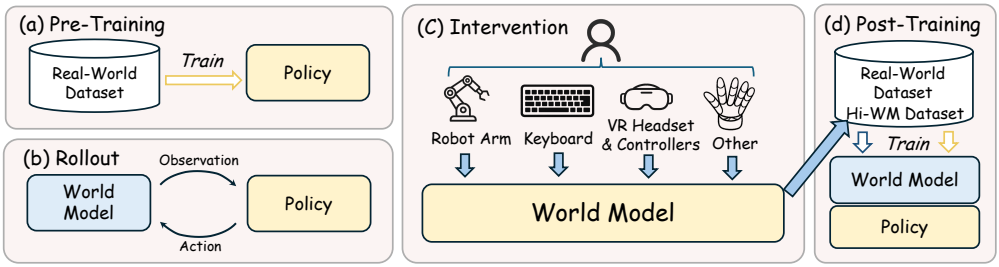

Figure 2: **Human-in-the-World-Model (Hi-WM) pipeline.** (a) In the pre-training stage, the policy network is learned from real-world data. (b) The policy is rolled out in closed-loop inside the world model. (c) When the policy exhibits suboptimal behavior, a human operator intervenes through hardware-agnostic input devices, such as a robot arm, keyboard, or VR controllers, and the corrective actions are executed directly in the world model. (d) The collected intervention segments are then added to the training set for post-training.

accordingly (Ross et al., 2011; Kelly et al., 2019; Liu et al., 2023). However, this loop remains tied to the physical world. Every rollout requires real robots, real scenes, human monitoring, and often manual reset. As coverage expands to more tasks, scenes, checkpoints, and long-tail failure modes, the marginal cost of post-training grows rapidly. For generalist policies, the bottleneck is therefore not only data efficiency, but deployment economics.

World models suggest a different substrate for post-training. Prior work has used learned dynamics primarily for imagination, synthetic data generation, policy optimization, or post-hoc evaluation. In contrast, we study the world model as an interactive corrective workspace for post-training. Our key observation is that the most valuable supervision often lies near the learner's own failure states: not in generic successful rollouts, but in short corrective continuations that recover from imminent failure. If such corrections are collected only in the physical world, each intervention remains tied to hardware time, scene reset, and sequential deployment.

We therefore propose **Human-in-the-World-Model (Hi-WM)**, a post-training framework in which a pretrained policy is rolled out inside an action-conditioned world model, and a human intervenes only when the rollout becomes incorrect, uncertain, or failure-prone. Instead of treating the world model as a generator or evaluator alone, Hi-WM uses it as a reusable corrective substrate: failure states can be cached, rewound, and branched into multiple alternative continuations, yielding dense recovery supervision. Given a trained world model, this corrective collection loop requires no execution on the target robot or physical scene resets. The resulting corrective trajectories are then used for post-training, e.g., with imitation learning or reinforcement learning. In this way, Hi-WM preserves the benefits of targeted human correction while changing the cost structure of post-training from a hardware-bound loop to a reusable interactive process.

We instantiate Hi-WM on three real-world manipulation tasks spanning rigid and deformable object interaction. Across these tasks, we study whether the world model is faithful enough to support intervention, whether world-model-collected corrective trajectories improve real-world policy performance, and whether the benefits scale with additional intervention data that are collected from the world simulator. Our results show that Hi-WM consistently improves real-world success over both the base policy and a world-model closed-loop baseline, suggesting that the main benefit comes not simply from more simulated rollouts, but from failure-targeted human correction inside the learned environment.

Our contributions are three-fold:

- **Human corrective post-training in the world model.** We formulate post-training as targeted human correction inside an action-conditioned world model, rather than collecting corrective supervision only during physical robot execution.
- **Rollback-and-branching corrective data collection.** We introduce a data collection mechanism that caches pre-failure states and reuses them to collect multiple corrective continuations, increasing the density and diversity of recovery supervision around failure-prone states.

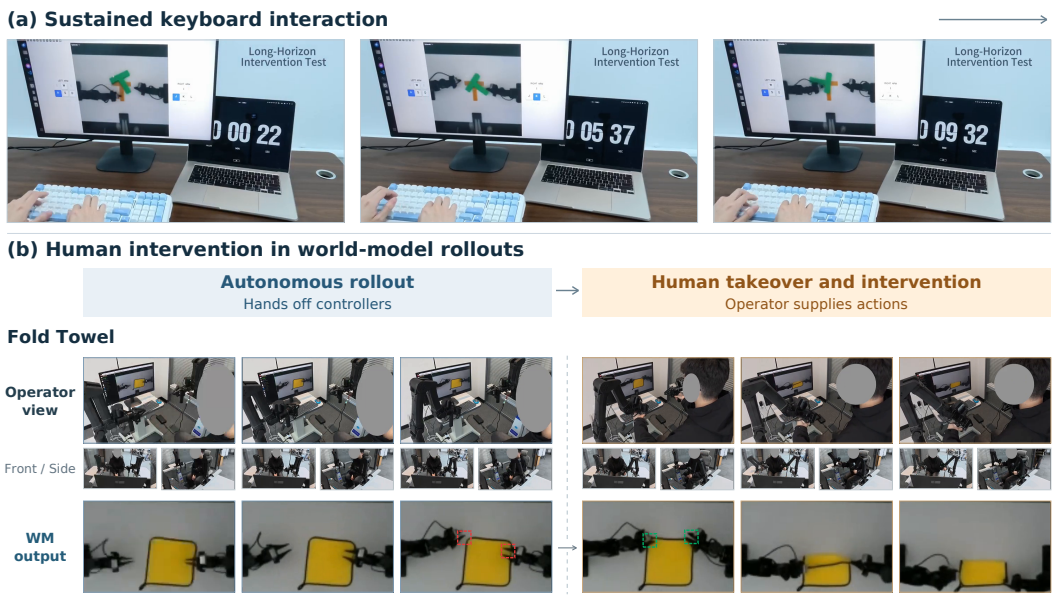

Figure 3: **Human intervention in world-model rollouts.** (a) Sustained keyboard interaction. (b) Fold Towel: red dashed boxes mark missed grasps during autonomous rollout; green dashed boxes mark successful grasps after human takeover. Operator views, smaller front/side views, and world-model outputs are aligned by source-video frame. Additional examples appear in Appendix C.

- **Real-robot evidence for effectiveness and scaling.** We show on three real-world manipulation tasks and two policy backbones that corrective trajectories collected in the world model improve downstream real-world performance, while offering increasingly favorable economics as virtual intervention data scale.

A detailed discussion of related work appears in Appendix A.

2 Methodology

2.1 System Overview

Hi-WM combines an action-conditioned world model, a pretrained robot policy, and a hardware-agnostic human intervention interface. The policy acts on the current observation, and the world model predicts the next observation from the action. A human can take over when the rollout becomes failure-prone and return control to the policy after correction. The generated observations and corrective actions are combined with real-world demonstrations for post-training.

The system requires faithful action responses, including in failure-prone states; a shared continuous action representation for policy and human control; and state caching to revisit failures and collect alternative recovery branches. Virtual interaction removes manual scene resets during corrective collection and supports remote operators without occupying the target robot.

2.2 Interactive World Model for Hi-WM

To support corrective interaction, the world model must follow robot actions and reproduce object interactions. Our implementation follows the encoder–dynamics–decoder framework of Wang et al. (2026): a convolutional encoder maps images to spatial latents, an action-conditioned dynamics network predicts subsequent latents using causal temporal attention, and a latent-conditioned consistency decoder reconstructs images. Encoder–decoder training is followed by latent-dynamics training with the encoder fixed. Appendix E specifies the architecture, training objectives, action encoding, and temporal inference procedure. The data design focuses on the following aspects:

Continuous action conditioning. The robot-side representation contains 14 coordinates: six joint coordinates and one gripper value per arm. The action mapper converts these commands to the configured world-model representation. Full end-effector control uses position, a continuous six-dimensional rotation encoding, and gripper state, giving 20 conditioning channels for two arms. The implementation also supports a four-channel planar representation for pushing. Policy and human commands use the same representation within each configuration; Appendix E details the mappings and normalization.

Fine-grained action discrimination through failure data. To accurately simulate the interaction between objects and robot arms, the world model should follow the commanded action closely, rather than only capturing the overall motion trend (e.g., simply approaching the object). Relying on such approximations often leads to over-optimistic predictions. For instance, if a robotic manipulator moves generally toward a target, the model might bypass exact action execution and hallucinate a successful interaction, even if the specific orientation is incorrect. To prevent the model from relying only on coarse motion trends, we include failure cases and off-task states alongside successful trajectories. Specifically, the dataset contains incomplete tasks, failed grasps, unintended object contacts, and misaligned robot poses. This helps the model distinguish between actions that lead to successful interactions and actions that lead to failure.

Edge-case data coverage. To make the collected data useful for post-training, the world model must stay well aligned with real-world execution. This correspondence ensures that trajectories collected via the model remain physically meaningful when transferred to the real system.

Specifically, once a user performs an action in the world model during human interaction, the action should be faithfully reproduced in the physical world. If the world model differs too much from real execution in robot motion or object interaction, the collected trajectories become unreliable for post-training. To address this, our solution is to collect a diverse dataset of edge cases. These cases include states near the boundary of the workspace and tasks that require precise gripper alignment. By capturing these edge-case trajectories, the model can accurately reflect the geometry of the scene and the outcomes of executed actions. We evaluate this alignment through repeated positioning accuracy in Section 3.2.1 and Figure 6.

With these designs and data strategies, our world model provides a real-time simulation environment that responds faithfully to actions. It serves as the core infrastructure where policy rollout, human intervention, and corrective trajectory collection seamlessly integrate to enable scalable post-training.

2.3 Hardware-Agnostic Intervention Interface

To support practical human intervention within the world model, we develop a hardware-agnostic teleoperation interface. The goal is to decouple the input device from the target robot while keeping control accurate.

The interface supports multiple input devices to suit different task requirements. For simple tasks, a keyboard provides basic, low-dimensional control. For high-precision corrections, operators can use calibrated robot arms to specify end-effector motions, which are then mapped into the policy action space. Alternatively, VR controllers can be used to map tracked spatial poses directly to the robot end-effectors. Regardless of the hardware used, a unified action mapper converts these varied inputs into the standard continuous action representation required by the policy.

This flexibility avoids reliance on a single setup, making human intervention significantly more practical and efficient.

Figure 3 illustrates sustained keyboard interaction and human intervention in a Fold Towel world-model rollout.

2.4 Collecting Corrective Trajectories for Post-Training

Within Hi-WM, post-training data are collected in a closed-loop inside the interactive world model. The pre-trained policy first runs in the world model from the current observation and generates a rollout. When the rollout enters unfamiliar or failure-prone states, a human operator intervenes through the hardware-agnostic interface and provides corrective actions. The world model then continues the rollout from the current state using these human actions. Once the rollout has been guided back toward successful behavior, control can be returned to the policy, as shown in Figure 4.

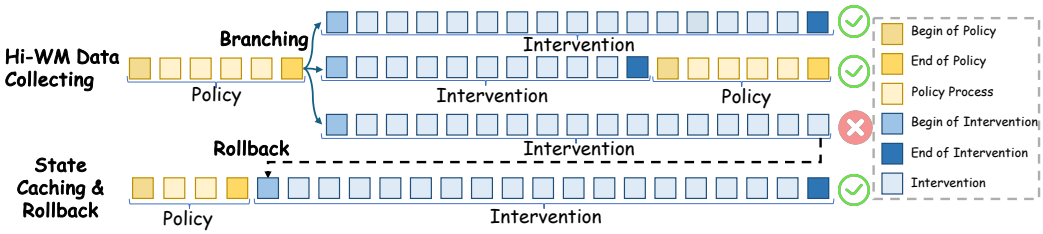

Figure 4: **State caching and rollback mechanism of Hi-WM.** Hi-WM supports state caching and rollback, allowing a rollout to be rewound to an earlier state and reused to collect multiple corrective branches, thereby improving both the efficiency and diversity of post-training data.

A key advantage of operating in the world model is that the system can cache intermediate states and revisit them when needed. Instead of restarting from the beginning after each failure, the operator can rewind the rollout to an earlier timestep and try a different correction from the same state. Consequently, a single simulated failure can be branched into diverse alternative futures, systematically yielding a high-density set of corrective trajectories. This makes data collection more efficient and also increases the diversity of recovery behaviors included in the post-training data.

The collected corrective segments are then merged with the original real-world demonstrations and used to post-train the base policy. Because these segments are collected around failure-prone states, they provide targeted supervision on behaviors that the base policy does not handle well. Since this process takes place in the world model rather than on the physical robot, it reduces repeated deployment, manual resets, and continuous hardware use during intervention. In this way, Hi-WM turns corrective data collection for post-training from a hardware-bound process into a more efficient and scalable pipeline. Appendix B describes the post-training data budgets and trajectory screening, while Appendix D analyzes collection efficiency through shared-prefix reuse.

3 Experiments

We evaluate world-to-real alignment (RQ1), post-training effectiveness across policy backbones (RQ2), cost and data scaling (RQ3), and generalization to held-out scene variations (RQ4).

3.1 Experimental Setup

Real robot tasks. We evaluate three manipulation tasks spanning rigid and deformable objects (Figure 5). In **Fold Towel**, the robot grasps two towel corners and folds the towel into a target configuration. In **Push-T**, it pushes and rotates a T-shaped object to match a target pose. In **Route Rope**, it grasps the rope ends, lifts them, and places them into two designated slots.

Robot setup. We conduct real-world experiments on a dual-arm YAM-Ultra platform. Each arm provides 6-DoF motion control and a gripper state. Observations are captured from a top-view camera, and robot actions are collected and executed at 15 Hz.

World-model rollout initialization. Each world-model rollout starts from a real task-start observation: the top-view image and the initial poses of both arms. The policy receives the image and end-effector poses; its actions are converted to end-effector poses for the world model, which generates subsequent observations in closed loop.

Real-world evaluation protocol. For each method–policy combination in Table 2, we evaluate 38 trials on Fold Towel, 34 trials on Push-T, and 34 trials on Route Rope. Within each task, all compared methods use the same set of initial configurations. Appendix B provides trial counts, result aggregation, and collection details.

Post-training data budget. Each task has approximately 20 hours of original offline trajectories. DP and π_0 collect separate corrections from their own rollouts. Within each task–policy setting, WM-CL and Hi-WM match retained trajectory counts and action steps; shared prefixes are counted once. This matches retained data, not collection time or training compute.

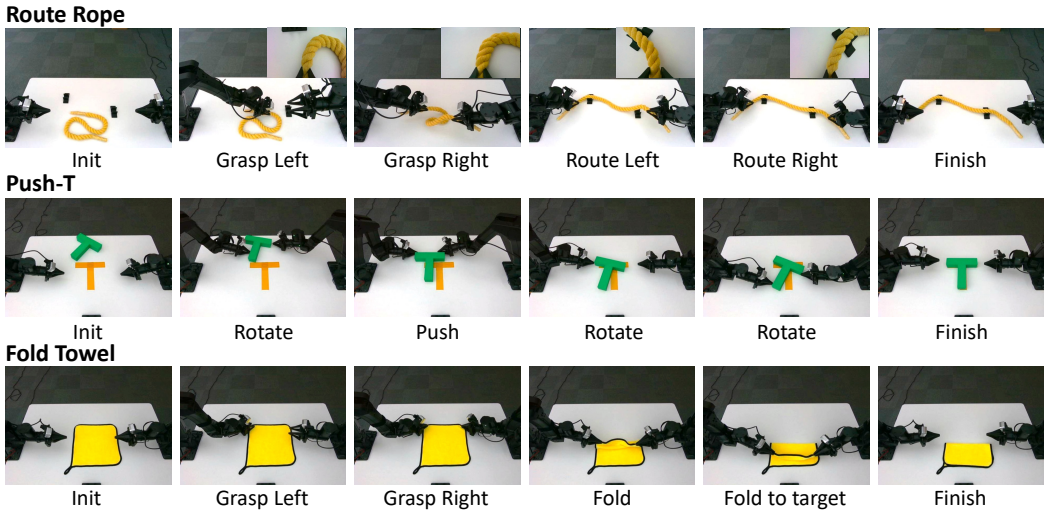

Figure 5: **Evaluated Tasks.** We evaluate our method on three representative manipulation tasks that cover both rigid and deformable object interaction.

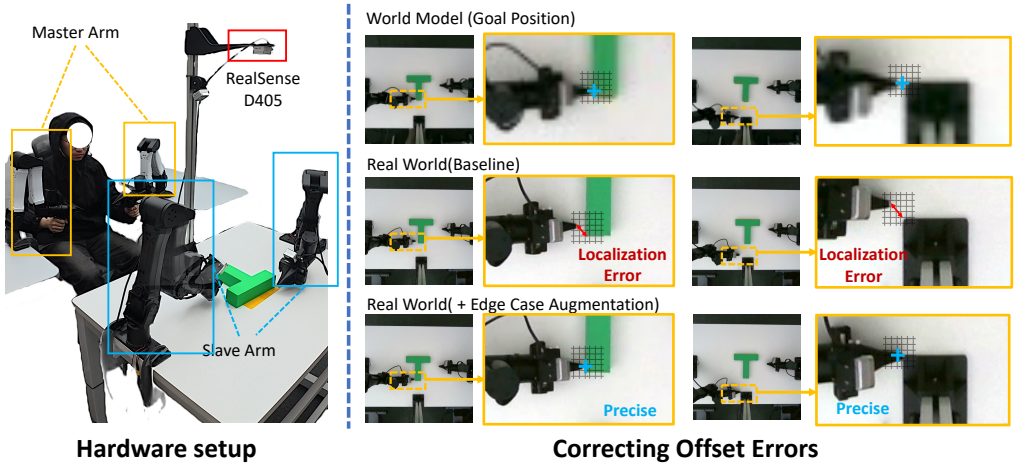

Figure 6: **Hardware setup and positioning alignment. Left:** Master–slave teleoperation setup and overhead RealSense D405 camera. **Right:** World-model targets and real-world execution before and after edge-case augmentation. Measured positioning errors decrease from approximately 25 mm to 1 mm.

Trajectory evaluation and screening. Generated rollouts are assessed automatically, while human operators monitor the current state in real time. In WM-CL, the policy generates all actions and human involvement is limited to monitoring and success screening. In Hi-WM, operators can additionally take over to provide corrective actions. When the automated assessment disagrees with the human judgment, the human judgment takes precedence.

Baselines. To rigorously evaluate our approach, we compare it against the following baselines:

- **Base Policy**: A policy trained solely on the initial offline dataset without any human intervention.
- **World Model Closed-Loop**: The policy is rolled out autonomously within the world model, and only trajectories selected as successful through automated assessment and human screening are retained for fine-tuning. No human corrective actions or takeovers are used in this baseline.

Table 1: **Impact of edge-case data scale on visual fidelity.** We measure image prediction quality with PSNR (↑), SSIM (↑), and LPIPS (↓). Percentages indicate the fraction of edge-case trajectories added for augmentation relative to the complete edge-case dataset. Increasing the amount of edge-case data consistently yields higher PSNR and SSIM and lower LPIPS. The full augmentation setting achieves the strongest overall fidelity.

Ratio (%)	PSNR ↑	SSIM ↑	LPIPS ↓
0 (Base)	18.50	0.815	0.152
20	19.82	0.853	0.124
50	21.15	0.901	0.098
100 (Full)	**22.53**	**0.942**	**0.055**

Table 2: **Real-world success rates (%).** Hi-WM uses Iteration 2 for both backbones. Each method–policy pair uses 38 Fold Towel trials, 34 Push-T trials, and 34 Route Rope trials. WM-CL and Hi-WM match retained additional trajectory counts and action steps. Best results are bold. Appendix B details evaluation and aggregation.

Method	Policy	Success Rate (%)		
		Fold Towel	Push T	Route Rope
Base	DP	42.1	52.9	47.1
	Pi0	55.3	76.5	64.7
WM-CL	DP	76.3	64.7	70.6
	Pi0	78.9	79.4	82.4
Ours	DP	**92.1**	**85.3**	**94.1**
	Pi0	**97.4**	**97.1**	**100.0**

3.2 Real-World Alignment of Hi-WM

To make human intervention effective in the real world, and to ensure the quality of post-training data, an interactive world model must stay well aligned with physical execution. Visual plausibility alone is not enough. The model must also produce outcomes that can be realized by the real robot. We study this alignment from three perspectives. First, we test whether target positions specified in the world model can be reproduced by the real robot arm. Second, we test whether the model can interpolate to unseen interior points within the data-covered workspace, while remaining stable over long interaction sequences. Third, we test whether task performance in the world model matches task performance in the real world.

3.2.1 Repeated Positioning Accuracy

We first test repeated target realization, namely whether a target specified in the world model can be reproduced by the physical robot with high precision. In practice, positioning errors are more likely to occur near the boundary of the workspace and close to the robot's motion limits, so we introduce edge-case data to improve coverage of these rare regions.

Figure 6 compares world-model target positions with physical execution. Without edge-case data, the measured positioning errors are approximately 25 mm; with augmentation, they decrease to approximately 1 mm. These measurements support improved world-to-real positioning alignment. Separately, Table 1 reports visual prediction fidelity through PSNR, SSIM, and LPIPS, which improve with edge-case coverage. Adding edge-case data at approximately 30% of the regular-scene data volume yields clear gains.

3.2.2 Interpolation within the Covered Workspace

Accurate reproduction of a small set of observed points is not enough. A useful interactive world model should also support stable interpolation within the covered workspace. In other words, once the training data covers the operation region at a sufficient level, the model should not only memorize a sparse set of sampled points. It should also produce reasonable and stable interactions at interior points and states that were not explicitly collected.

We evaluate this property with a continuous ten-minute interaction sequence on Push-T, which is showcased in our demonstration. Across the sequence, the rollout traverses many intermediate states not explicitly collected in training, yet the generated interaction remains smooth, with consistent robot and object motion throughout. This provides qualitative evidence that the model supports interpolation within the covered workspace rather than merely memorizing a sparse set of anchor states, while also remaining stable over long interaction horizons.

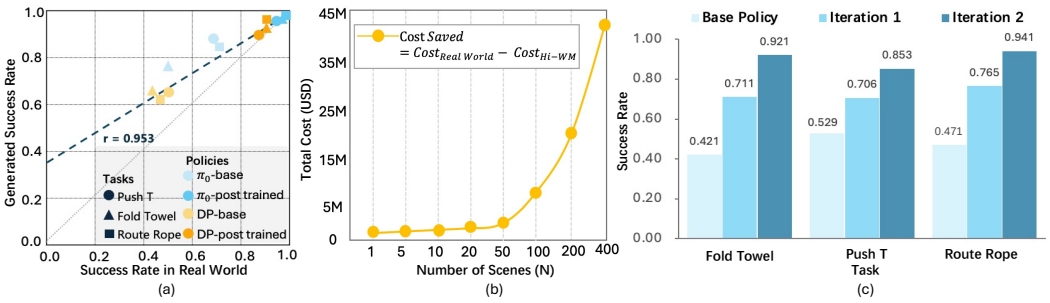

Figure 7: **Evaluation correlation, illustrative cost projection, and data scaling.** **(a):** World-model versus real-world success rates. **(b):** Projected cost difference, $\mathrm{Cost}_{\mathrm{real}} - \mathrm{Cost}_{\mathrm{Hi-WM}}$, as scene coverage increases under the assumed collection scenario; this is an illustrative projection, not measured end-to-end savings. **(c):** Real-world success rates with increasing amounts of virtual corrective data.

3.2.3 Task Level Evaluation Consistency

Finally, we test whether task performance in the world model correlates with task performance in the real world. As shown in Figure 7(a), the two success rates are strongly correlated, with a Pearson correlation coefficient of $r = 0.953$.

This result shows that the world model is not only aligned with real world at the execution level, but also predictive at the task level. It can therefore serve as a reliable proxy for policy evaluation. This gives human operators a useful signal for deciding when and where intervention is needed, and it supports the generation of high-quality intervention data for post-training.

3.3 Policy Performance via Hi-WM

To answer **RQ2**, we evaluate whether the resulting corrective trajectories improve downstream policy performance. Table 2 reports real-world success rates on the three tasks for two base policies, DP (Chi et al., 2023) and π_0 (Black et al., 2024). Hi-WM achieves the best result in all six task–policy settings, showing that its benefit is not tied to a particular policy architecture. Taking an unweighted average across the six task–policy settings, our method improves success by 37.9 percentage points over the base policy and by 19.0 points over the world-model closed-loop baseline. The closed-loop baseline compares successful autonomous rollout augmentation with the complete Hi-WM corrective collection procedure. This comparison assesses the combined value of human corrective supervision, failure targeting, and branching relative to autonomous successful-rollout augmentation. Although this baseline improves over the base policy in all six settings, it remains consistently below Hi-WM.

We attribute this gap to the type of supervision collected by the two methods. In the closed-loop baseline, the additional data mainly reinforce behaviors already available to the current policy. In contrast, Hi-WM introduces human correction when the rollout begins to fail, so the collected segments are concentrated around failure-prone states that the base policy does not handle well. Moreover, the interactive world model supports rollback and branching from a pre-failure state, allowing multiple corrective continuations to be collected efficiently from the same rollout. This likely provides richer supervision for recovery behavior, which in turn leads to higher real-world success after post-training.

3.4 Efficiency and Scaling of Hi-WM

Illustrative cost projection. Figure 7(b) illustrates a possible economic benefit of replacing repeated physical corrective collection with virtual interaction. The projected difference is $\mathrm{Cost}_{\mathrm{real}} - \mathrm{Cost}_{\mathrm{Hi-WM}}$, including fixed setup and operating costs under the assumed collection scenario. This projection should not be interpreted as measured end-to-end savings. The realized advantage depends on the amount of corrective collection and the extent to which a trained world model can be reused. An end-to-end comparison must also account for the real data required to build the world

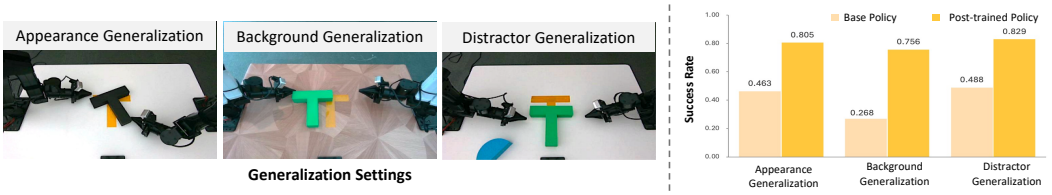

Figure 8: **Real-world policy generalization to unseen scenes. Left:** Held-out Push-T scenes with appearance, background, and distractor variations. **Right:** Real-world success rates of the Base Policy and the policy post-trained with corrective trajectories collected within Hi-WM. The post-trained policy achieves higher success rates across all three generalization settings.

model, model training and maintenance, inference, operator time, and validation of generated trajectories.

Collection efficiency through shared-prefix reuse. Rollback reduces the repeated work needed to reach a failure state. For K corrective continuations from the same branch point, the preceding rollout is generated once rather than K times. Let P denote the time required to execute this shared prefix, and let C_{cache} and C_{restore} denote the caching and per-restoration overheads. Holding continuation costs fixed, the analytical collection-time saving is $(K-1)(P-C_{\text{restore}})-C_{\text{cache}}$. When these overheads are small relative to prefix execution, the saving approaches $(K-1)P$. This reuse avoids repeated policy inference, world-model generation, and monitoring along the prefix, concentrating collection on alternative recovery behaviors. It is particularly useful when short corrective continuations follow long prefixes, since the cost of reaching a failure state can be shared across multiple recovery attempts. Appendix D provides the derivation.

Scaling with virtual intervention data. For each task and policy backbone, Iteration 1 adds approximately 4 hours of corrective trajectories (20% of the task's 20-hour offline dataset), and Iteration 2 increases the cumulative addition to approximately 7 hours (35%). Shared prefixes count once, so the second stage adds approximately 3 hours of distinct trajectory duration. These budgets describe retained training data, not operator time. Corrections are collected separately from each backbone's rollouts, and the policy is post-trained after each stage. Figure 7(c) shows DP performance improving across both stages on all three tasks. The Hi-WM results for both backbones in Table 2 use Iteration 2.

3.5 Policy Generalization via Hi-WM

To answer **RQ4**, we evaluate the base and post-trained policies on previously unseen real-world Push-T scenes with appearance, background, and distractor variations. These evaluation scenes are held out from policy training, including corrective post-training in Hi-WM. Figure 8 shows the scene variations and real-world success rates. The post-trained policy outperforms the Base Policy in all three settings, showing that gains from virtual corrective supervision extend to unseen real-world scenes.

4 Conclusion

We introduced Human-in-the-World-Model (Hi-WM), a framework that moves human corrective data collection into an interactive world model. State caching, rollback, and branching allow failure states to be reused without repeated physical execution or scene resets. Corrective trajectories improve real-world policy success across rigid and deformable manipulation tasks, with further gains as virtual intervention data increase within the tested range. Hi-WM provides a reusable workspace for policy evaluation and correction, reducing dependence on physical deployment during corrective collection while retaining real-world validation.

AI USE STATEMENT

We used generative AI tools to assist with manuscript writing and language polishing, including refining wording, correcting grammar, and improving clarity and readability. AI assistance also supported formatting and compilation checks. The authors take full responsibility for the final content of this paper, including its scientific claims, results, references, and AI-assisted text.

REFERENCES

1X Technologies. 1X World Model — 1x.tech. https://www.1x.tech/discover/1x-world-model, 2024. [Accessed 16-05-2025].

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Maren Awiszus, Frederik Schubert, and Bodo Rosenhahn. World-gan: a generative model for minecraft worlds. In *2021 IEEE Conference on Games (CoG)*, pp. 1–8. IEEE, 2021.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. *arXiv preprint arXiv:2412.03572*, 2024.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

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, 2024. URL https://arxiv.org/abs/2410.24164.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

Xiaoyu Chen, Junliang Guo, Tianyu He, Chuheng Zhang, Pushi Zhang, Derek Cathera Yang, Li Zhao, and Jiang Bian. Igor: Image-goal representations are the atomic control units for foundation models in embodied ai. *arXiv preprint arXiv:2411.00785*, 2024.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Xiaowei Chi, Peidong Jia, Chun-Kai Fan, Xiaozhu Ju, Weishi Mi, Kevin Zhang, Zhiyuan Qin, Wanxin Tian, Kuangzhi Ge, Hao Li, et al. Wow: Towards a world omniscient world model through embodied interaction. *arXiv preprint arXiv:2509.22642*, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

Jensen Gao, Suneel Belkhale, Sudeep Dasari, Ashwin Balakrishna, Dhruv Shah, and Dorsa Sadigh. A taxonomy for evaluating generalist robot manipulation policies. *arXiv preprint arXiv:2503.01238*, 2025.

Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with action: Visual policy learning via joint denoising process. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. *arXiv preprint arXiv:2510.10125*, 2025.

Yanjiang Guo, Tony Lee, Lucy Xiaoyang Shi, Jianyu Chen, Percy Liang, and Chelsea Finn. Vlaw: Iterative co-improvement of vision-language-action policy and world model. *arXiv preprint arXiv:2602.12063*, 2026.

D HO, J MONAS, JT REN, and C YU. 1x world model: evaluating bits, not atoms, 2025.

Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S. Brown, and Ken Goldberg. ThriftyDAgger: Budget-aware novelty and risk gating for interactive imitation learning. In *Proceedings of the 5th Conference on Robot Learning*, pp. 598–608, 2022.

Ryan Hoque, Lawrence Yunliang Chen, Satvik Sharma, Karthik Dharmarajan, Brijen Thananjeyan, Pieter Abbeel, and Ken Goldberg. Fleet-DAgger: Interactive robot fleet learning with scalable human supervision. In *Proceedings of The 6th Conference on Robot Learning*, pp. 368–380, 2023.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

Siyuan Huang, Liliang Chen, Pengfei Zhou, Shengcong Chen, Zhengkai Jiang, Yue Hu, Yue Liao, Peng Gao, Hongsheng Li, Maoqing Yao, et al. Enerverse: Envisioning embodied future space for robotics manipulation. *arXiv preprint arXiv:2501.01895*, 2025.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loïc Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. DreamGen: Unlocking generalization in robot learning through video world models. In *Proceedings of The 9th Conference on Robot Learning*, pp. 5170–5194, 2025.

Snehal Jauhri, Carlos Celemin, and Jens Kober. Interactive imitation learning in state-space. In *Proceedings of the 2020 Conference on Robot Learning*, pp. 682–692, 2021.

Yueru Jia, Jiaming Liu, Shengbang Liu, Rui Zhou, Wanhe Yu, Yuyang Yan, Xiaowei Chi, Yandong Guo, Boxin Shi, and Shanghang Zhang. Video2act: A dual-system video diffusion policy with robotic spatio-motional modeling. *arXiv preprint arXiv:2512.03044*, 2025.

Yunfan Jiang, Chen Wang, Ruohan Zhang, Jiajun Wu, and Fei-Fei Li. TRANSIC: Sim-to-real policy transfer by learning from online correction. In *Proceedings of The 8th Conference on Robot Learning*, pp. 1691–1729, 2025a.

Yuxin Jiang, Shengcong Chen, Siyuan Huang, Liliang Chen, Pengfei Zhou, Yue Liao, Xindong He, Chiming Liu, Hongsheng Li, Maoqing Yao, et al. Enerverse-ac: Envisioning embodied environments with action condition. *arXiv preprint arXiv:2505.09723*, 2025b.

Zhennan Jiang, Kai Liu, Yuxin Qin, Shuai Tian, Yupeng Zheng, Mingcai Zhou, Chao Yu, Haoran Li, and Dongbin Zhao. World4rl: Diffusion world models for policy refinement with reinforcement learning for robotic manipulation. *arXiv preprint arXiv:2509.19080*, 2025c.

Zhennan Jiang, Shangqing Zhou, Yutong Jiang, Zefang Huang, Mingjie Wei, Yuhui Chen, Tianxing Zhou, Zhen Guo, Hao Lin, Quanlu Zhang, et al. Wovr: World models as reliable simulators for post-training vla policies with rl. *arXiv preprint arXiv:2602.13977*, 2026.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083, 2019.

Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Pathdreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14738–14748, 2021.

Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg. DART: Noise injection for robust imitation learning. In *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 143–156, 2017.

Chenhao Li, Andreas Krause, and Marco Hutter. Robotic world model: A neural network simulator for robust policy optimization in robotics. *arXiv preprint arXiv:2501.10100*, 2025a.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Proceedings of The 8th Conference on Robot Learning*, pp. 3705–3728, 2025b.

Yaxuan Li, Yichen Zhu, Junjie Wen, Chaomin Shen, and Yi Xu. Worldeval: World model as real-world robot policies evaluator. *arXiv preprint arXiv:2505.19017*, 2025c.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. In *Robotics: Science and Systems (RSS)*, 2023.

Xiaokang Liu, Zechen Bai, Hai Ci, Kevin Yuchen Ma, and Mike Zheng Shou. World-vla-loop: Closed-loop learning of video world model and vla policy. *arXiv preprint arXiv:2602.06508*, 2026.

Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning, 2024a.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning, 2024b.

Ajay Mandlekar, Caelan Garrett, Danfei Xu, and Dieter Fox. Human-in-the-loop task and motion planning for imitation learning. In *Conference on Robot Learning (CoRL)*, volume 229 of *PMLR*, 2023.

Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. In *Robotics: Science and Systems (RSS)*, 2023.

Xiaowen Qiu, Yian Wang, Jiting Cai, Zhehuan Chen, Chunru Lin, Tsun-Hsuan Wang, and Chuang Gan. LuciBot: Automated robot policy learning from generated videos. *arXiv preprint arXiv:2503.09871*, 2025.

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. WorldGym: World model as an environment for policy evaluation. *arXiv preprint arXiv:2506.00613*, 2025.

Pengzhen Ren, Kaidong Zhang, Hetao Zheng, Zixuan Li, Yuhang Wen, Fengda Zhu, Mas Ma, and Xiaodan Liang. Surfer: Progressive reasoning with world models for robotic manipulation. *arXiv preprint arXiv:2306.11335*, 2023.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 627–635, 2011.

Younggyo Seo, Junsu Kim, Stephen James, Kimin Lee, Jinwoo Shin, and Pieter Abbeel. Multi-view masked world models for visual robotic manipulation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 30613–30632. PMLR, 2023. URL `https://proceedings.mlr.press/v202/seo23a.html`.

Ansh Kumar Sharma, Yixiang Sun, Ninghao Lu, Yunzhe Zhang, Jiarao Liu, and Sherry Yang. World-gymnast: Training robots with reinforcement learning in a world model. *arXiv preprint arXiv:2602.02454*, 2026.

Jonathan Spencer, Sanjiban Choudhury, Matt Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Siddhartha Srinivasa. Learning from interventions: Human-robot interaction as both explicit and implicit feedback. In *Proceedings of Robotics: Science and Systems*, 2020. doi: 10.15607/RSS.2020.XVI.055.

Gemini Robotics Team, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Fangchen Liu, Anirudha Majumdar, et al. Evaluating gemini robotics policies in a veo world simulator. *arXiv preprint arXiv:2512.10675*, 2025.

Yixuan Wang et al. Interactive world simulator for robot policy training and evaluation. *arXiv preprint arXiv:2603.08546*, 2026.

Edgar Welte and Rania Rayyes. Interactive imitation learning for dexterous robotic manipulation: Challenges and perspectives—a survey. *Frontiers in Robotics and AI*, 12:1682437, 2025. doi: 10.3389/frobt.2025.1682437.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023.

Philipp Wu, Yide Shentu, Qiayuan Liao, Ding Jin, Menglong Guo, Koushil Sreenath, Xingyu Lin, and Pieter Abbeel. RoboCopilot: Human-in-the-loop interactive imitation learning for robot manipulation. *arXiv preprint arXiv:2503.07771*, 2025.

Junjin Xiao, Yandan Yang, Xinyuan Chang, Ronghan Chen, Feng Xiong, Mu Xu, Wei-Shi Zheng, and Qing Zhang. World-env: Leveraging world model as a virtual environment for vla post-training. *arXiv preprint arXiv:2509.24948*, 2025.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

Kaidong Zhang, Pengzhen Ren, Bingqian Lin, Junfan Lin, Shikui Ma, Hang Xu, and Xiaodan Liang. Pivot-r: Primitive-driven waypoint-aware world model for robotic manipulation. *arXiv preprint arXiv:2410.10394*, 2024.

Zhongru Zhang, Chenghan Yang, Qingzhou Lu, Yanjiang Guo, Jianke Zhang, Yucheng Hu, and Jianyu Chen. Veo-act: How far can frontier video models advance generalizable robot manipulation? *arXiv preprint arXiv:2604.04502*, 2026.

Pengfei Zhou, Liliang Chen, Shengcong Chen, Di Chen, Wenzhi Zhao, Rongjun Jin, Guanghui Ren, and Jianlan Luo. Act2goal: From world model to general goal-conditioned policy. *arXiv preprint arXiv:2512.23541*, 2025a.

Zhiyuan Zhou, Pranav Atreya, You Liang Tan, Karl Pertsch, and Sergey Levine. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world. *arXiv preprint arXiv:2503.24278*, 2025b.

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Mani-WM: An interactive world model for real-robot manipulation, 2024. URL `https://openreview.net/forum?id=aVyJwS1fqQ`.

A Related Work

Post-training with human corrective supervision. Human-in-the-loop robot learning has long studied how to improve policies under the learner-induced state distribution by querying expert correction during deployment or interactive training (Liu et al., 2023; Jiang et al., 2025a; Wu et al., 2025; Mandlekar et al., 2023; Welte & Rayyes, 2025). More recent work extends this idea to generalist robot policies and post-deployment refinement, where human interventions, preferences, or corrective demonstrations are used to adapt a pretrained policy to downstream tasks (Ross et al., 2011; Laskey et al., 2017; Kelly et al., 2019; Spencer et al., 2020; Jauhri et al., 2021; Hoque et al., 2022; 2023; Luo et al., 2024b;a). These approaches show the value of targeted supervision near policy failure. However, the corrective loop remains predominantly tied to physical execution: each intervention consumes robot time, scene setup, reset effort, and operator attention in the real world. Our work addresses this bottleneck by moving corrective interaction from the robot to a learned environment.

World models for data generation and policy improvement. A separate line of work uses world models (Wu et al., 2023; Seo et al., 2023; Mendonca et al., 2023) as scalable sources of training signal, including imagination for policy learning, synthetic trajectory generation (Koh et al., 2021; Bruce et al., 2024; Bar et al., 2024; Awiszus et al., 2021; Zhu et al., 2024; Seo et al., 2023; Ren et al., 2023; Zhang et al., 2024; Li et al., 2025a; Huang et al., 2025; Jiang et al., 2025b; Wang et al., 2026; Ali et al., 2025; Sharma et al., 2026; Guo et al., 2026; Xiao et al., 2025; Jiang et al., 2026; Chi et al., 2025; Liu et al., 2026; Guo et al., 2025; Jia et al., 2025), and world-model-based policy optimization (Bharadhwaj et al., 2024; Qiu et al., 2025; Jang et al., 2025; Hu et al., 2024; Chen et al., 2024; Ye et al., 2024; Guo et al., 2024; Du et al., 2023; Black et al., 2023; Guo et al., 2026; Sharma et al., 2026; Zhou et al., 2025a; Ali et al., 2025; Chi et al., 2025; Guo et al., 2025; Zhang et al., 2026). These methods leverage learned dynamics to expand coverage beyond manually collected demonstrations and reduce dependence on expensive real-world interaction. Despite their differences, the world model is typically used as an **autonomous generator, simulator, or optimizer**. By contrast, our focus is not autonomous rollout alone, but **human-guided corrective interaction inside the world model**, with the resulting trajectories used specifically for post-training around failure-prone states.

World model for policy evaluator. World models are also increasingly studied as evaluation (Li et al., 2025b; Zhou et al., 2025b; Gao et al., 2025) substrates for robot policies, where the central question is whether model rollouts preserve policy ranking, correlate with real-world success, and support tasks such as checkpoint selection or pre-deployment screening (Li et al., 2025c; Quevedo et al., 2025; Guo et al., 2025; 1X Technologies, 2024; Wang et al., 2026; HO et al., 2025; Huang et al., 2025; Jiang et al., 2025b; Ali et al., 2025; Guo et al., 2026; Sharma et al., 2026; Jiang et al., 2025c; Liao et al., 2025; Team et al., 2025). This literature highlights that useful evaluation environments require not only visual realism, but also action faithfulness and consistency with real-world outcomes. Our work is complementary: rather than using the world model only as a post-hoc evaluator, we use it as an interactive corrective workspace in which evaluation, intervention, and corrective data collection are coupled within the same learned environment.

In summary, prior work has explored human corrective learning in the physical world, and world models as generators, evaluators, or optimization environments. Hi-WM lies at the intersection of these directions but addresses a distinct problem setting: **how to use a world model as a reusable workspace for human-in-the-loop post-training**. The defining feature of our formulation is that failure states inside the learned environment can be cached, rewound, and branched into multiple corrective continuations, converting one-shot physical correction into reusable recovery supervision for post-training.

B Evaluation and Post-training Details

B.1 Evaluation Counts and Initial Configurations

For each method–policy combination in Table 2, Fold Towel, Push-T, and Route Rope use 38, 34, and 34 evaluation trials, respectively. Across the three methods and two policy backbones, these counts sum to 636 task trials. The counts refer to evaluation episodes, not independent training runs. Within each task, all compared methods are evaluated using the same set of initial configurations.

For the Push-T generalization evaluation in Figure 8, each policy is evaluated on 41 trials in each of the appearance, background, and distractor settings. Success rates are computed separately for each policy and setting as the number of successful trials divided by 41.

B.2 Aggregation of Main Results

The average improvement is computed as the unweighted mean of the percentage-point differences over the six task–policy settings. Using the displayed success rates in Table 2, Hi-WM improves over Base and WM-CL by 37.9 and 19.0 percentage points, respectively, after rounding to one decimal place. These are averages across settings, rather than success-rate differences pooled over all trials.

B.3 Post-training Data Budget

Each of the three tasks has approximately 20 hours of original offline trajectories. All durations below are per task, rather than totals across tasks, and measure trajectory duration rather than operator collection time. DP and π_0 collect separate corrective datasets from their respective rollouts; corrective trajectories are not shared between the two backbones.

Within each task–policy setting, WM-CL and Hi-WM match both the number of additional retained trajectories and their total action steps. A shared prefix contributes once to the retained-data budget, regardless of how many branches reuse it; duplicated prefixes contribute no additional duration. This accounting convention does not specify how often samples are drawn during optimization. Matching retained data does not establish equal collection cost or training compute.

Scaling experiment and main-table checkpoint. For each task–policy setting, approximately 20 hours of original offline trajectories are supplemented with cumulative corrective additions of approximately 4 hours (20%) at Iteration 1 and 7 hours (35%) at Iteration 2. The second stage therefore adds approximately 3 hours of distinct trajectories, with shared prefixes counted once. Both DP and π_0 Hi-WM results in Table 2 are from Iteration 2; Figure 7(c) shows the DP scaling results. The approximate hours are consistent with the reported percentages; they do not replace exact trajectory and action-step counts from dataset records.

B.4 Automated Assessment and Human Oversight

Generated rollouts are evaluated automatically, while a human operator monitors the current state in real time. In WM-CL, all rollout actions are generated by the policy; the operator only monitors and screens trajectories for success, without providing corrective actions or taking over. In Hi-WM, the operator can additionally take over to correct failure-prone behavior. When the automated assessment and human judgment disagree, the human judgment takes precedence. WM-CL therefore uses autonomous action generation with human screening, whereas Hi-WM also incorporates human corrective control.

C Additional Human Intervention Examples

Figure 9 complements the Fold Towel example in Figure 3 with intervention sequences for Push-T and Route Rope.

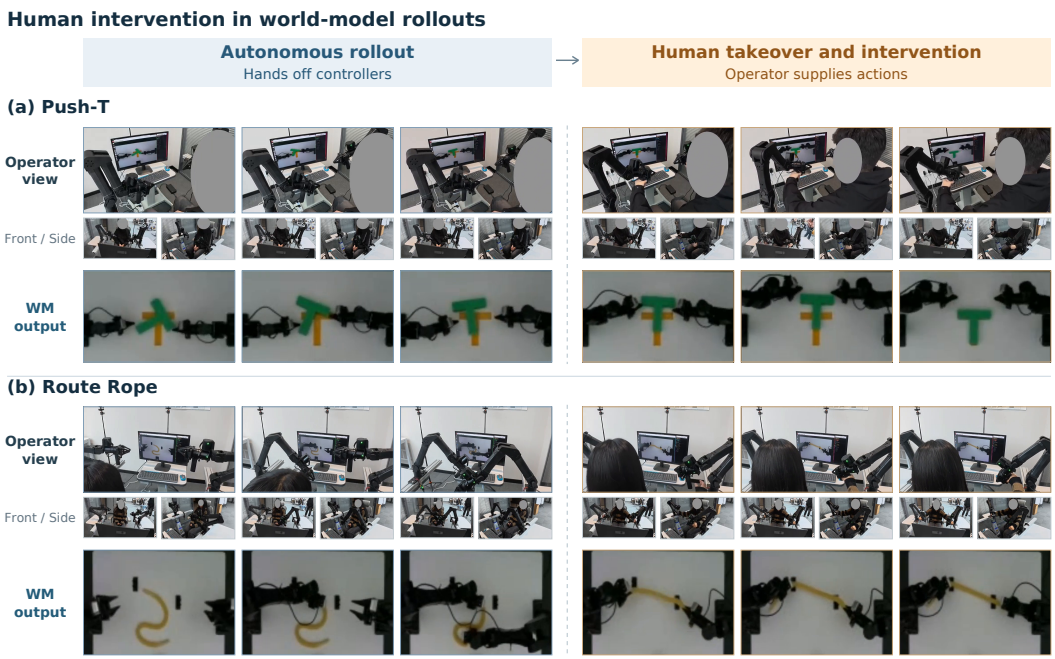

Figure 9: **Additional human-intervention examples in the world model.** (a) Push-T. (b) Route Rope. Each sequence contains three autonomous-rollout moments followed by three human-intervention moments. Operator views, smaller front (left) and side (right) views, and world-model outputs in each column come from the same source-video frame. Columns follow chronological order; their spacing does not represent elapsed time.

D COLLECTION EFFICIENCY THROUGH SHARED-PREFIX REUSE

Reusing the trajectory before intervention. The rollback mechanism in Section 2.4 and Figure 4 improves corrective data collection by preserving the shared trajectory prefix. Consider collecting several corrective continuations from the same pre-failure state. All continuations share the observations and actions that led to this branch point. With state caching, this prefix is generated once and retained. After exploring one continuation, the operator restores the cached branch point and collects another correction, reusing the preceding rollout. Each additional branch therefore requires a state restoration and a new continuation, avoiding repeated policy execution and world-model generation along the shared prefix, together with any repeated operator monitoring of that prefix.

Collection-cost analysis. For a fixed set of K corrective continuations from one branch point, let P denote the time required to execute the shared prefix, including policy inference, world-model generation, and any required monitoring. Let B_i denote the time to collect continuation i, including human correction and subsequent rollout. We compare restarting and re-executing the prefix for each continuation with restoring a cached branch point, holding the continuation costs B_i fixed. Let C_{cache} be the one-time caching overhead and C_{restore} the overhead of each restoration. The respective collection times are

$$T_{\text{restart}} = KP + \sum_{i=1}^{K} B_i, \tag{1}$$

$$T_{\text{rollback}} = P + \sum_{i=1}^{K} B_i + C_{\text{cache}} + (K-1)C_{\text{restore}}. \tag{2}$$

Thus, the time saved by reusing the prefix is

$$\Delta T = T_{\text{restart}} - T_{\text{rollback}} = (K-1)(P - C_{\text{restore}}) - C_{\text{cache}}. \tag{3}$$

When caching and restoration overheads are small relative to re-executing the prefix, the saving approaches $(K-1)P$: the shared prefix is executed once instead of K times. More generally, rollback reduces collection time whenever $(K-1)(P - C_{\text{restore}}) > C_{\text{cache}}$. This makes prefix reuse especially useful when short corrective continuations follow a long rollout, and allows the cost of reaching a failure-prone state to be amortized across multiple corrections.

Role in Hi-WM. Human intervention supplies corrective actions at failure-prone states, while rollback enables multiple such corrections to reuse the same preceding interaction. These components form a single collection workflow: once a useful branch point is reached, further collection can focus on alternative recoveries from that state. The WM-CL comparison evaluates the complete Hi-WM workflow against successful autonomous-rollout collection; Equation 3 characterizes the specific collection-time saving obtained by sharing the prefix across corrective branches.

E Implementation and Training Details

E.1 Architecture and Action Conditioning

The implementation uses a convolutional image encoder, a latent dynamics network, and a consistency decoder following Wang et al. (2026). Table 3 lists the default architecture parameters in the implementation; the action input width is determined by the selected control representation.

Table 3: **Default world-model architecture configuration.** Spatial latent shape is shown for one RGB view. These are code-level architecture defaults; task-specific training and inference configurations can override them.

Component	Configuration
Image resolution	128×128
Encoder latent	$4 \times 32 \times 32$; two spatial downsampling blocks
Action embedding MLP	$d_a \to 64 \to 128 \to 128 \to 512$
Dynamics width / multipliers	64 / $(1, 2)$
Dynamics attention	4 heads; 128 channels per head; causal in time
Decoder width / multipliers	64 / $(1, 2, 3)$
Decoder residual blocks / dropout	2 per level / 0.1

Robot coordinates and model inputs. The YAM interface represents each arm by six joint coordinates and one gripper value. For end-effector conditioning, forward kinematics and the calibrated base-to-world transform convert these joint coordinates to poses in a common world frame. Full pose conditioning uses three position coordinates, a six-dimensional rotation representation, and one gripper value per arm. Table 4 distinguishes the robot-side coordinates from the supported model encodings. Actions are normalized using per-coordinate range statistics. The policy interface accepts absolute commands; keyboard increments update the current normalized command, which is clipped to $[-1, 1]$ before model execution.

Table 4: **Supported YAM action representations.** The robot-side coordinate count and the encoded world-model input dimension describe different interfaces.

Representation	Coordinates per arm	Total dimension
Joint-space control	6 joint coordinates + 1 gripper	14
Full end-effector pose	3 position + 6 rotation + 1 gripper	20
Planar end-effector control	x, y position	4

E.2 Training Objectives and Module Updates

The training implementation separates image reconstruction from latent dynamics learning. In the first stage, the encoder and decoder are optimized jointly on images. In the second stage, the trained encoder produces latent targets under a no-gradient context and optimization updates the dynamics network. Both the decoder and dynamics objectives use a noise-level-weighted mean-squared error between the prediction and a target at the sampled target noise level. The shared configuration selects fused signal-to-noise-ratio weighting. The decoder is conditioned on encoded images, whereas the dynamics network is conditioned on the action sequence. When the corresponding module uses more than one denoising step, an additional weighted term reconstructs the clean image or latent from the target-noise sample. Targets are detached from the gradient computation.

Optimization uses AdamW; the shared algorithm configuration specifies weight decay 10^{-4} and momentum coefficients $(0.9, 0.999)$. The linear warm-up option increases the learning-rate multiplier from 10^{-4} to 1 over the configured warm-up interval. Learning rates, batch sizes, training duration, and initialization checkpoints are stage-specific run settings.

E.3 Preprocessing and Autoregressive Interaction

The dataset loader aligns image and action sequences by timestep and resizes RGB observations to the configured resolution. Image values are mapped from $[0, 1]$ to $[-1, 1]$, and action normalization uses the corresponding coordinate statistics. At initialization, the interaction server encodes the starting observation. It then appends each new command to the action history, predicts one new latent, and decodes that latent into the next observation for the policy or human operator.

The server retains at most ten latent frames and the aligned action history. The dynamics network additionally bounds its input window to ten temporal positions, including the latent being generated; a full window therefore contains up to nine preceding latents. This temporal window is distinct from the number of iterative denoising steps, which is separately configured for the dynamics network and decoder. The default algorithm configuration specifies one dynamics step and three decoder steps. Generated latents and their aligned actions form the temporal context for subsequent continuations.