WAM-OPD: On-Policy Distillation for World Action Models

Liuhaichen Yang[1], **Zhuang Jiang**[1], **Chenchao Sheng**[2] **and Zezhi Tang**[1,*]

[1]Department of Computer Science, University College London
[2]Department of Mechanical Engineering, University College London
[*]Corresponding author: Zezhi Tang. E-mail: zezhi.tang@ucl.ac.uk

Abstract

World action models (WAMs) couple visual future prediction with robot action generation, but accelerated students can lose task capabilities during distillation and later encounter states that are poorly represented by offline data. We study whether on-policy distillation (OPD) can repair such a student without requiring sparse-reward reinforcement learning. We introduce WAM-OPD, a deployment-consistent post-training recipe for a video-first WAM. The student acts in the environment and therefore determines the history distribution. A frozen teacher labels those student histories with coherent video and action targets, while the student action branch is trained under its own generated video plan, as it is at deployment. Joint video and action losses update lightweight adapters in the shared backbone, together with an action flow-matching regularizer. In preliminary RoboTwin 2.0 studies on two tasks, the released one-video/one-action-step Flash-WAM improves from 0.0% to 58.3% success on HANDOVER MIC, and from 16.7% to 33.3% on PUT OBJECT CABINET. These task-specific results are an initial capability proof rather than evidence of broad or uniform generalization. They nevertheless suggest that dense teacher supervision on student-induced histories is a promising post-training interface for video-first WAMs.

Keywords *world action models, on-policy distillation, robot learning, flow matching, post-training*

1. Introduction

Large-scale robot policies are increasingly trained as general-purpose models rather than task-specific controllers. RT-1 demonstrated that Transformer policies can absorb diverse real-robot experience, while RT-2 connected robot actions with Internet-scale vision–language pretraining [4, 5]. Cross-embodiment datasets and open generalist policies have subsequently broadened the range of robots, tasks, and observation/action spaces that can share a policy initialization [11, 19, 20]. In parallel, generative action decoders have become an important alternative to direct regression: Diffusion Policy models multimodal action distributions through iterative denoising, and π_0 uses flow matching to generate continuous action chunks [3, 7]. Together, these developments have established vision–language–action (VLA) models as a practical foundation for downstream robot adaptation.

Most VLAs nevertheless treat future physical evolution only implicitly: a policy maps the current history to actions without requiring an explicit visual account of what those actions should cause. A growing family of predictive robot models makes this structure explicit. RoboDreamer uses generated video as a compositional plan, Prediction with Action learns visual prediction and control through a joint denoising process, and recent unified video–action and world action models couple the two modalities inside one generative model [9, 16, 26, 27]. LingBot-VA advances this direction with an autoregressive video–action world model: it predicts future video latents from the causal history and then decodes an action chunk

conditioned on both that visual future and the history [14]. Video and action are therefore factorized at the output level but coupled within an interleaved causal model. This design offers a useful inductive bias: visual prediction represents intended state change, while inverse dynamics grounds that prediction in control. We use *World Action Model* (WAM) for this broader family throughout the paper.

The generative formulation also creates a deployment bottleneck. Diffusion and flow policies ordinarily require repeated network evaluations to integrate a trajectory from noise to a sample [7, 17]. Progressive distillation and consistency models reduce this cost by learning few-step or one-step maps that preserve a pretrained generative trajectory [18, 23, 24]. Flash-WAM adapts consistency distillation to the asymmetric noise regimes of video and action streams, compressing the released LingBot-VA pipeline to few-step inference without changing its base model architecture [2]. This is denoising-step distillation on offline training samples, rather than behavioral OPD on Student-controlled environment histories. Yet an accelerated Student need not preserve every capability of its slower Teacher uniformly. Even when aggregate benchmark performance is strong, a downstream task can expose a local Teacher–Student gap that the original offline distillation data did not resolve.

Closing such a gap after release is not merely a matter of running supervised fine-tuning for longer. Offline imitation and distillation optimize a fixed data distribution, whereas deployment histories are generated by the Student's own closed-loop decisions. Small errors can therefore alter the states at which later predictions are made—the classical covariate-shift motivation for interactive imitation learning [22]. Online reinforcement learning (RL) restores Student-controlled interaction and can improve task success, as shown for VLAs by SimpleVLA-RL and for WAMs by WAM-RL [13, 21]. However, binary robot success is sparse, and credit assignment over long trajectories can be costly. On-policy distillation (OPD) offers a complementary post-training interface: let the Student determine the occupancy distribution, but query a stronger frozen Teacher for dense supervision on what the Student actually visits. Here, "on-policy" describes where supervision is evaluated; it does not require policy-gradient RL or a sparse reward. This principle appears in Generalized Knowledge Distillation for autoregressive models and is brought to robot action tokens by VLA-OPD [1, 25]; DiffusionOPD and Flow-OPD extend related reasoning to continuous diffusion transitions and flow states [8, 15].

Applying OPD to an accelerated WAM is technically different from applying it to an action-only policy. A video-first WAM factorizes the deployed policy as

$$p_\theta(z_t, a_t \mid h_t) = p_\theta(z_t \mid h_t)\, p_\theta(a_t \mid h_t, z_t), \tag{1}$$

where h_t is the closed-loop history, z_t is a generated video plan, and a_t is an action chunk. This introduces two coupled distribution shifts. At the environment level, supervision must cover Student-induced histories $h_t \sim d^{\pi_S}$. At the model level, the deployed action branch is conditioned on the Student plan z_S, not the Teacher plan z_T. Training actions only under z_T therefore creates a conditional-interface mismatch, even when the Teacher action itself is strong. Moreover, video and action paths share Transformer blocks, so updating one modality can change the representation used by the other. A WAM-oriented OPD method must decide both *whose histories to label* and *which video-to-action computation to train*.

We introduce WAM-OPD, a deployment-consistent post-training recipe for this setting. The released Student acts in the environment and supplies the history distribution. A frozen LingBot-VA Teacher labels those Student histories with a coherent video target z_T and action target a_T. During optimization, however, the trainable Student follows its deployment graph: it first produces z_S, then predicts a_S from $\mathrm{sg}(z_S)$. Video and action losses, together with an action flow-matching auxiliary, update rank-8 JointLoRA adapters across all 30 shared blocks. This

makes the Student-side conditional path exact at training and deployment, while using video alignment to reduce the remaining Teacher-plan/Student-plan target gap.

We evaluate this design as a deliberately narrow capability proof on HANDOVER MIC and PUT OBJECT CABINET in RoboTwin 2.0 [6]. For each task, we start from released one-video/one-action-step Flash-WAM and use eight Student trajectories, 160 Teacher-labeled contexts, and 120 optimizer steps. Each selected checkpoint is evaluated on six held-out scene seeds under two fixed noise banks. Exact-paired success changes from 0.0% to 58.3% on HANDOVER MIC and from 16.7% to 33.3% on PUT OBJECT CABINET. Because each pair of noise-bank runs reuses the same scene seed, and because only two clean tasks are tested, these results support repeatability beyond a single task but not broad or uniform generalization.

Our contributions are:

- We identify a conditioning mismatch specific to video-first WAM OPD: deployed actions consume a video plan generated by the Student, not the Teacher.
- We define a joint video–action loss and a LoRA update for a shared, flow-based WAM. We delimit it from exact reverse-KL and full pathwise transition matching.
- We provide exact-paired RoboTwin evaluations on two tasks. The same recipe improves held-out success in both settings, while their different effect sizes expose the evidence still required for a general claim.

2. Related Work

Generalist robot policies and generative action models. Scaling robot data and model capacity has produced policies that transfer across tasks, embodiments, and language instructions. RT-1 studies large-scale real-robot policy training, RT-2 integrates vision–language pretraining with action tokens, and Open X-Embodiment standardizes multi-institution robot data for cross-embodiment learning [4, 5, 20]. Octo and OpenVLA provide open generalist policy initializations, while OpenVLA-OFT shows that action representation, chunking, and decoding choices materially affect downstream adaptation [11, 12, 19]. Alongside token-based VLAs, Diffusion Policy and π_0 generate continuous action sequences through diffusion or flow matching [3, 7]. These works establish the policy and adaptation substrate for our study. Our question is narrower: how to post-train an already accelerated WAM when the action generator consumes an internal video prediction.

Predictive robot policies and world action models. World models have long used learned dynamics for control, but recent video generators make high-dimensional future observations available as explicit robot plans. RoboDreamer factorizes video imagination for compositional goals; Prediction with Action and Video Prediction Policy connect visual prediction with robot control [9, 10, 26]. Unified Video Action Model and Unified World Models couple video and action generation, rather than treating video solely as an external planner [16, 27]. LingBot-VA instead uses a causal video-first factorization: predicted future latents condition inverse-dynamics action generation, while video and action tokens remain coupled through an interleaved attention architecture [14]. More broadly, WAM policies can either generate explicit visual futures at test time or use world-model representations without decoding future video during every action query. We study the former, specifically the released LingBot-VA/Flash-WAM pipeline. These architectures motivate joint supervision, but joint generation alone does not specify how a frozen Teacher should label histories visited by a faster Student. WAM-OPD focuses on that post-release Teacher–Student interface.

Few-step diffusion and WAM acceleration. The iterative inference cost of diffusion and flow models has motivated progressive distillation, consistency models, and latent consistency models [18, 23, 24]. Their common goal is to approximate a many-step generative trajectory with a small number of function evaluations. Flash-WAM preserves the LingBot-VA model architecture and specializes this idea to its WAM solver: a frozen Teacher advances a denoising state, while the online Student and an EMA target learn a shared clean endpoint along that trajectory. Because video and action occupy different noise regimes, Flash-WAM uses modality-aware consistency parameterizations and optimizes their losses jointly [2]. Our method does not replace that native acceleration recipe. It initializes from the released accelerated checkpoint and changes a different distribution: the closed-loop robot histories on which the Student receives supervision.

Interactive and reinforcement-learning post-training. DAgger addresses imitation-learning covariate shift by repeatedly querying an expert on learner-induced states [22]. Modern robot post-training also uses online RL: SimpleVLA-RL optimizes VLA trajectories from binary task outcomes, while WAM-RL studies actor-only and joint world/action optimization [13, 21]. The shared occupancy principle is that the current policy, rather than a static demonstration set, determines which states receive learning signal. The supervision differs: RL obtains scalar environment returns, whereas our pilot uses dense frozen-Teacher video/action labels. We do not currently provide a compute- or interaction-matched RL comparison, so we position OPD as a complementary training signal rather than a superior alternative.

On-policy distillation in discrete and continuous generators. GKD trains autoregressive Students on self-generated prefixes and supports a family of distributional divergences against the Teacher [1]. VLA-OPD moves this idea to environment occupancy: the VLA Student controls robot rollouts, while a frozen Teacher provides action-token distributions on the visited states and the Student minimizes reverse KL [25]. DiffusionOPD and Flow-OPD instead define occupancy inside a continuous generative process. They query Teacher transitions or vector fields at states visited by the Student sampler [8, 15]. These forms of OPD share Student-induced support and dense Teacher queries, but differ in whether "state" denotes an environment history, a token prefix, a diffusion state, or a flow state. They do not, however, define the video-first WAM case in which an action distribution is conditioned on a separately generated Student video plan and both modalities share trainable blocks. Our endpoint pseudo-Huber losses plus flow-matching auxiliary should therefore be understood as a WAM-specific, deployment-consistent distillation objective—not as VLA-OPD's token reverse-KL or a reproduction of full DiffusionOPD/Flow-OPD path matching.

3. Preliminaries

World Action Model policies. A World Action Model couples predictive world modeling with action generation. Existing policies differ in how explicitly prediction enters control. Some use world-model features while producing actions directly; others generate a future visual trajectory at test time and condition control on that imagined future. Within the latter family, causal models incorporate new observations between action chunks, whereas chunk-wise models generate a bounded future from the current context. This taxonomy is organizational rather than universal. We study the explicit-imagination, causal policy instantiated by the released LingBot-VA/Flash-WAM checkpoint [2, 14].

Let $h_t = (o_{\leq t}, a_{<t}, \ell)$ contain the observation–action history and language instruction at control

time t. The policy first samples a future video-plan latent z_t, then an action chunk a_t:

$$p_\theta(z_t, a_t \mid h_t) = p_\theta(z_t \mid h_t)\, p_\theta(a_t \mid h_t, z_t). \tag{2}$$

This output factorization is hierarchical, but it does not imply two independent networks. The released LingBot-VA Teacher and Flash-WAM Student use the same model class and configuration: modality-specific video/action input, time, and output modules surround one shared 30-block Transformer. Flash-WAM initializes its Student from LingBot-VA and distills the generative solver without changing this backbone architecture [2, 14]. At deployment, the shared model first generates and caches the video plan, then generates actions from that cache. The plan is therefore part of the deployed action condition, and environment histories follow the closed-loop Student occupancy $h \sim d^{\pi_S}$.

Flow matching. A time-dependent vector field $v_\sigma : \mathbb{R}^d \to \mathbb{R}^d$ defines a flow through the ordinary differential equation

$$\frac{d}{d\sigma}\phi_\sigma(x) = v_\sigma(\phi_\sigma(x))\,. \tag{3}$$

Flow Matching learns a neural field v_θ by regressing to the velocity that generates a prescribed probability path [17]. Using the convention of Flash-WAM, a straight conditional path between clean data x_0 and noise ϵ is

$$x_\sigma = (1-\sigma)x_0 + \sigma\epsilon, \qquad u_\sigma(x_\sigma \mid x_0) = \epsilon - x_0, \tag{4}$$

where generation integrates from the noise boundary $\sigma = 1$ toward $\sigma = 0$. Conditional Flow Matching minimizes

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{\sigma, x_0, \epsilon}\left[\left\| v_\theta(x_\sigma, \sigma, c) - (\epsilon - x_0)\right\|_2^2\right], \tag{5}$$

with context c containing language and robot history. Video and action can use different noise schedules and parameterizations even when their representations interact in the same WAM.

Numerical integration ordinarily requires multiple network evaluations. Consistency distillation instead trains a map that sends different points on a Teacher trajectory to a common clean endpoint [2, 24]. Flash-WAM uses different consistency parameterizations for high-noise video and low-noise action streams, then optimizes the two modality losses jointly. This accelerates the generative solver; it does not address which closed-loop robot histories appear after the accelerated Student is deployed.

On-policy distillation. On-policy distillation changes the support on which the Teacher supervises the Student. In its general form, the Student generates contexts, a frozen Teacher is evaluated on the same contexts, and the Student minimizes a dense discrepancy:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{h \sim d^{\pi_S}}\left[\mathcal{D}(q_\theta(\cdot \mid h), q_T(\cdot \mid h))\right]. \tag{6}$$

The sampled trajectory is normally treated as data rather than differentiated through. The discrepancy $\mathcal{D}$ depends on the generator: GKD supports token-distribution divergences on Student prefixes; VLA-OPD uses reverse KL on action tokens at Student-visited environment states; DiffusionOPD and Flow-OPD match Teacher transitions or vector fields at Student sampler states [1, 8, 15, 25]. Thus "on-policy" names the Student-induced occupancy, not a requirement for sparse reward or policy-gradient RL. Our setting uses environment histories generated by the deployed WAM Student and supervises both its video plan and its conditioned action output; it does not claim full generative-path matching.

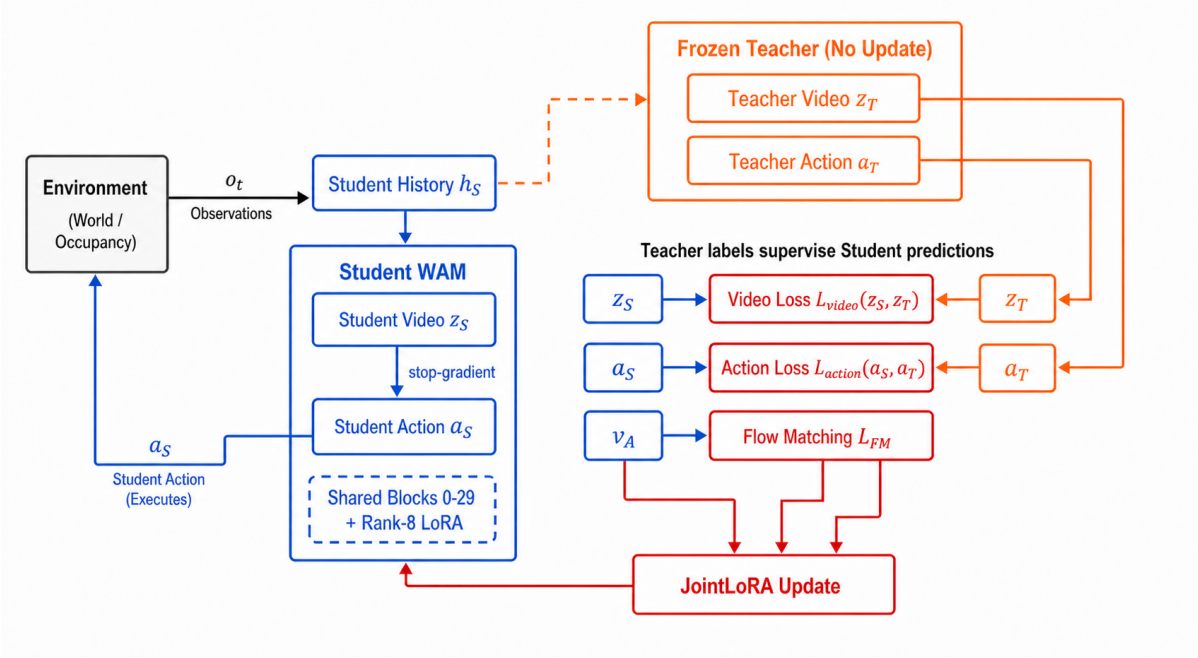

Figure 1. Overview of WAM-OPD. The student alone acts in the environment and determines the history distribution. A frozen teacher labels each student history with coherent video and action targets. The student action branch consumes its own stop-gradient video plan, matching deployment. Video, action, and flow-matching losses update only JointLoRA parameters in the shared student blocks. The precise objective is given in Eq. (9).

4. Method

4.1. Deployment-consistent WAM distillation

For each history h_S collected from a released Flash-WAM student, we run the trainable student in the same order used at inference:

$$z_S = f_\theta^{\text{vid}}(h_S), \tag{7}$$

$$a_S = f_\theta^{\text{act}}(h_S, \text{sg}(z_S)). \tag{8}$$

The stop-gradient prevents the action loss from changing the video solve through the plan tensor. It does not freeze the shared backbone during the action forward pass; both modality losses can update the same adapters.

A frozen, slower LingBot-VA teacher labels the same history with a coherent pair (z_T, a_T). In the current implementation, a_T is generated with the teacher plan z_T, rather than by querying the teacher under the exact student plan z_S. The video loss therefore plays two roles: it supervises visual prediction and reduces the conditional gap between the plan used for the teacher action target and the plan presented to the deployed student action branch. This design is deployment-consistent on the student side, but it is not exact-condition teacher matching.

4.2. Joint objective

We use the pseudo-Huber penalty $\rho_\delta(e) = \delta^2(\sqrt{1 + (e/\delta)^2} - 1)$ for robust endpoint regression. The training objective is

$$\begin{aligned}\mathcal{L}(\theta) = \lambda_z \rho_\delta(z_S - z_T) + \lambda_a \rho_\delta(a_S - a_T) \\ + \lambda_{\text{FM}} \left\| v_\theta^{\text{act}}(x^a_{\sigma=1}, 1, c) - (\epsilon_a - a_T) \right\|_2^2 .\end{aligned} \tag{9}$$

Our pilot sets $(\lambda_z, \lambda_a, \lambda_{\text{FM}}) = (1, 1, 0.2)$. The first two terms directly align the generated video and action endpoints. The third retains an action flow-matching learning signal at the high-noise boundary. We do not interpret this finite objective as an exact reverse-KL or as full denoising-trajectory matching.

4.3. Parameter-efficient update

The video and action paths have separate input, time, and output modules but share 30 Transformer blocks. We insert rank-8 JointLoRA adapters across shared blocks 0–29 and freeze the released weights. This scope lets both modalities modify the representation used by the video-first policy while keeping the update tractable. Joint training does not guarantee non-conflicting gradients; it simply aligns the trainable scope with the shared computation used by both outputs. Measuring per-modality gradient interaction is left to a controlled ablation.

4.4. Training protocol

The current proof of concept uses a fixed package of trajectories generated by the released student. Each trajectory is labeled by the frozen teacher, then reused for a bounded three-epoch update. This is on-policy with respect to the checkpoint that collected the package, but it becomes progressively stale as the student changes. A complete iterative version of WAM-OPD would alternate fresh student collection, teacher labeling, and bounded optimization; the present experiment tests one such post-training package only.

5. Experiments

5.1. Pilot question and tasks

We ask a deliberately narrow question: can joint deployment-consistent distillation turn dense teacher supervision into closed-loop task success for an accelerated WAM student? We evaluate two RoboTwin 2.0 tasks in the clean setting: HANDOVER MIC and PUT OBJECT CABINET [6]. In the former, the policy must transfer a microphone between two robot hands. The task exposes visual-plan and action coordination errors: before transfer the hands must approach and coordinate, while dropping the microphone can move the episode outside useful teacher support. In the latter, the policy must place and release an object in the designated cabinet drawer. We use RoboTwin's native success predicate; closing the drawer is not required and is not counted as a separate outcome.

5.2. Data, optimization, and checkpoint selection

For each task, we initialize from the released Flash-WAM one-video/one-action-step checkpoint. We use eight Student trajectories, 160 labeled macro-step contexts, and four disjoint calibration trajectories. The same update configuration is used for both tasks: rank-8 JointLoRA over all 30 shared blocks and AdamW with learning rate 2×10^{-5}. The batch size is 4; video, action, and action flow-matching losses have weights 1, 1, and 0.2. Each run contains three epochs and 120 optimizer steps.

We screen checkpoints on a disjoint split and select them using a predeclared ordering of task success, semantic progress, and calibration loss. This rule selects epoch 3 for both tasks. Screening data are excluded from the held-out results.

5.3. Exact-paired held-out evaluation

For each task, the held-out split contains six scene seeds and two fixed noise banks, producing 12 exact-paired evaluation units. Within each pair, Released and WAM-OPD share the instruction, initial simulator snapshot, scene seed, and noise bank. The primary outcome is RoboTwin's latched `eval_success`. The two noise-bank runs for a scene share the same scene seed, so 12 units are not 12 independent scene samples. We report tasks separately rather than pooling the 24 task–seed–bank units.

Table 1. Preliminary exact-paired performance on two `demo_clean` tasks. Each task comprises six held-out scene seeds under two noise banks. Percentage-point changes are computed within task.

Task	Released	WAM-OPD	Improvement
HANDOVER MIC	0.0%	**58.3%**	+58.3 pp
PUT OBJECT CABINET	16.7%	**33.3%**	+16.7 pp

As shown in Table 1, the same WAM-OPD recipe improves held-out success on both tasks. The larger gain on HANDOVER MIC and the smaller gain on PUT OBJECT CABINET suggest that the benefit depends on the task and evaluation distribution. These results support the core mechanism, but the samples remain too small for a general performance claim.

5.4. Planned evaluation

The next version will replace the placeholders in Table 2 with a multi-task study, matched baselines, and controlled ablations. The key tests are whether gains repeat across horizons and randomized settings, whether joint video supervision is necessary, and whether fresh on-policy recollection outperforms reuse of a fixed package at matched compute.

Table 2. Planned evidence matrix. Entries marked "TBD" are not experimental results.

Evaluation	Released	WAM-OPD	Status
Broader RoboTwin task suite	TBD	TBD	planned
Domain-randomized evaluation	TBD	TBD	planned
Action-only vs. joint update	TBD	TBD	planned
Fixed-package vs. refreshed OPD	TBD	TBD	planned
Cross-task retention	TBD	TBD	planned

6. Discussion and Limitations

The two task studies support a limited but useful conclusion: joint video–action supervision on Student-induced histories can recover task capability in an accelerated WAM. The result is consistent with our central hypothesis that WAM post-training should respect both the Student's deployment distribution and its video-to-action computation. The different improvement magnitudes also show that the current evidence is task-dependent.

Several limitations determine the next experiments. First, the evidence covers only two clean simulation tasks and a small held-out set. It provides neither a representative multi-task average nor evidence of real-robot transfer. Second, the fixed trajectory package is only on-policy for the released collector; iterative recollection is required to retain the formal on-policy property after updates. Third, the teacher action target is conditioned on the teacher video

plan. Video alignment can reduce, but does not eliminate, the teacher-plan/student-plan mismatch. Fourth, shared JointLoRA may produce helpful or harmful video/action gradient interaction, which is not measured here. Fifth, there is no compute-matched comparison to SFT, RL, action-only distillation, or full pathwise flow supervision.

Finally, RoboTwin success is taken from the official latched `eval_success`. Snapshot replay preserves pose and success-latch state but does not preserve the simulator contact manifold; consequently, auxiliary contact diagnostics recorded from restored snapshots are not reliable evidence of sustained contact. We therefore claim official task success and paired semantic progress, not verified stable contact mechanics.

7. Conclusion

We presented WAM-OPD, a preliminary post-training framework that applies on-policy distillation to video-first World Action Models. Its central design choice is to train the action branch under the Student's own video plan while a frozen Teacher supplies coherent labels on Student-induced histories. A parameter-efficient joint update raises exact-paired success from 0.0% to 58.3% on HANDOVER MIC and from 16.7% to 33.3% on PUT OBJECT CABINET. These results establish a promising two-task vertical slice, not a finished general method. The next stage is to broaden the task suite, refresh Student occupancy between updates, and separate the contributions of video supervision, action supervision, flow matching, and shared adaptation.

References

[1] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2024.

[2] Arman Akbari, Ci Zhang, Arash Akbari, Lin Zhao, Yixiao Chen, Weiwei Chen, Xuan Zhang, Geng Yuan, and Yanzhi Wang. Flash-WAM: Modality-aware distillation for world action models. *arXiv preprint arXiv:2606.05254*, 2026.

[3] Kevin Black, Noah Brown, Danny Driess, et al. π_0: A vision-language-action flow model for general robot control. *Robotics: Science and Systems*, 2025.

[4] Anthony Brohan et al. RT-1: Robotics transformer for real-world control at scale. *Robotics: Science and Systems*, 2023.

[5] Anthony Brohan et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *Conference on Robot Learning*, 2023.

[6] Tianxing Chen, Zanxin Chen, Baijun Chen, 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.

[7] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Robotics: Science and Systems*, 2023.

[8] Zhen Fang, Wenxuan Huang, Yu Zeng, Yiming Zhao, Shuang Chen, Kaituo Feng, Yunlong Lin, Lin Chen, Zehui Chen, Shaosheng Cao, and Feng Zhao. Flow-OPD: On-policy distillation for flow matching models. *arXiv preprint arXiv:2605.08063*, 2026.

[9] 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 *Advances in Neural Information Processing Systems*, 2024.

[10] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *International Conference on Machine Learning*, 2025.

[11] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, et al. OpenVLA: An open-source vision-language-action model. *Conference on Robot Learning*, 2024.

[12] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *Robotics: Science and Systems*, 2025.

[13] Haozhan Li et al. SimpleVLA-RL: Scaling VLA training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025.

[14] Lin Li, Qihang Zhang, Mingrui Yu, Zelin Gao, Yiming Luo, Nan Xue, Shuai Yang, Xing Zhu, Ruilin Wang, Yujun Shen, Fei Han, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[15] Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. DiffusionOPD: A unified perspective of on-policy distillation in diffusion models. *arXiv preprint arXiv:2605.15055*, 2026.

[16] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. In *Robotics: Science and Systems*, 2025.

[17] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

[18] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

[19] Octo Model Team et al. Octo: An open-source generalist robot policy. *Robotics: Science and Systems*, 2024.

[20] Open X-Embodiment Collaboration et al. Open X-embodiment: Robotic learning datasets and RT-X models. *IEEE International Conference on Robotics and Automation*, 2024.

[21] Zezhong Qian, Xiaowei Chi, Yu Qi, Haozhan Li, Zhi Yang Chen, and Shanghang Zhang. WAM-RL: World-action model reinforcement learning with reconstruction rewards and online video SFT. *arXiv preprint arXiv:2606.17906*, 2026.

[22] Stéphane Ross, Geoffrey Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

[23] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[24] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023.

[25] Zhide Zhong, Haodong Yan, Junfeng Li, Junjie He, Tianran Zhang, and Haoang Li. VLA-OPD: Bridging offline SFT and online RL for vision-language-action models via on-policy distillation. *arXiv preprint arXiv:2603.26666*, 2026.

[26] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

[27] Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Robotics: Science and Systems*, 2025.