Modality-Autoregressive World-Action Models

Adam Hung, Bardienus P. Duisterhof, Deva Ramanan, Jeffrey Ichnowski
Carnegie Mellon University

Project page: https://adamhung60.github.io/ModAR/

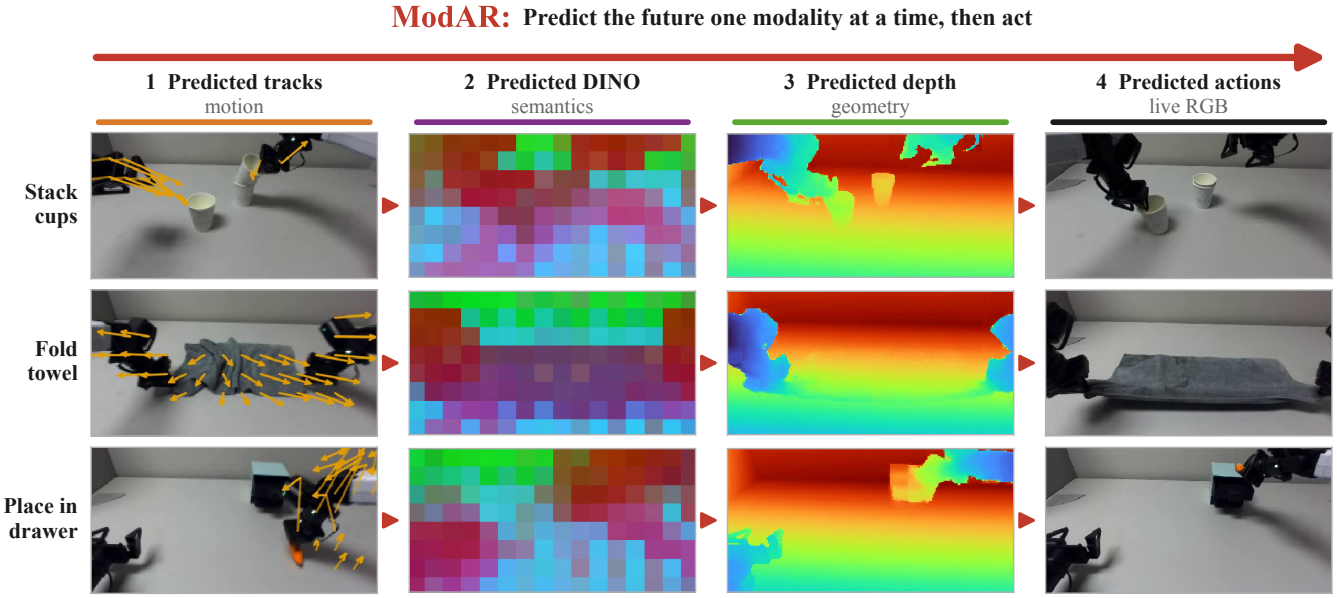


Fig. 1. We present ModAR, a world-action model (WAM) that predicts future observations as a sequence of multiple modalities. ModAR autoregressively denoises one modality at a time before finally denoising robot actions. We train all models from scratch for controlled comparisons across predicted modalities and training data. We find that additional modalities improve performance, and our multimodal formulation outperforms prior WAM formulations. We further validate ModAR on real-world bimanual manipulation using robot demonstrations and human videos.

***Abstract*— World-action models (WAMs) jointly model future observations and actions, typically predicting the future as RGB images. Other visual modalities such as depth, pretrained visual features, and point tracks can more efficiently capture geometric, semantic, and motion features. However, how best to combine these modalities within WAMs remains an open question. We introduce ModAR, the first WAM to autoregressively denoise multiple future modalities before predicting actions. This allows each prediction to condition on previously generated modalities. We train from scratch to systematically study how training-data mixtures, predicted modalities, and WAM formulations affect performance. In our evaluations, WAMs benefit from predicting point tracks, DINO features, and depth maps, while additionally predicting future RGB does not provide a consistent benefit. We also find that ModAR's sequential generation outperforms existing WAM formulations, with the highest average success rate at all evaluated data scales. We also fine-tune the video-model-initialized WAM Flex-π on the same data; ModAR achieves a slightly higher observed average success rate (75% vs. 72%) while using approximately $20\times$ fewer training FLOPs and no pretraining. On three real-world bimanual tasks, ModAR outperforms baselines and improves with human videos.**

I. Introduction

Future-observation prediction is a powerful objective for learning rich representations of the world's dynamics and semantics. World-action models (WAMs) harness this objective for robotics by jointly modeling future observations and corresponding robot actions.

While robot action prediction requires *action-labeled* robot data, future-observation prediction can learn from broader *actionless* data sources or initialize from pretrained video generation models. Supervision on these broader data sources can benefit action prediction both as a training-time auxiliary objective, even when future generation is omitted at deployment [1], and by enabling policies to condition action generation on predicted visual futures [2], [3]. Recent predictive world models have explored other representations of the future that emphasize physical, spatial, or semantic structure. These include depth [4], point tracks [5], [6], and pretrained visual features such as DINO [7], [8]. Here, we use *modality* to denote a representation of the future, including both sensory signals and derived features. These modalities each contain unique inductive biases that capture manipulation-relevant features. In this work, we ask: *How should WAMs combine multiple modalities?*

Our contributions are as follows: **(1)** We introduce ModAR, a WAM that autoregressively denoises multiple future-observation modalities before generating actions. **(2)** We contribute controlled experiments on how WAM formulation,

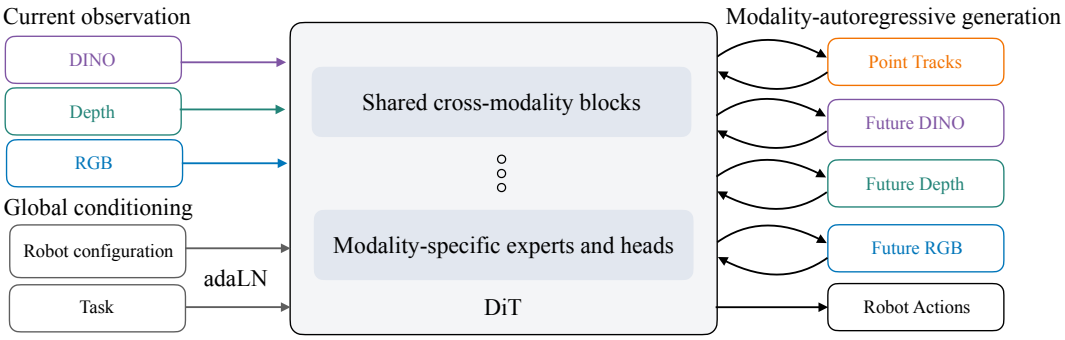


Fig. 2. **ModAR architecture.** We embed the current observation from each modality and pass the resulting tokens into a shared diffusion transformer. The DiT combines cross-modal blocks with modality-specific experts and output heads. Robot configuration and a learned task embedding provide global adaLN conditioning to each DiT layer. ModAR sequentially denoises future modalities, conditioning each on earlier predictions, and denoises actions last.

target modalities, and actionless data scale affect policy performance. We find that modality-autoregressive generation performs best across all evaluated data scales and benefits most from additional actionless data. In our experiments, predicting point tracks, DINO features, and depth provide additive gains, while additionally predicting future RGB provides no consistent benefit. We also fine-tune the 6B-parameter video-pretrained WAM Flex-π [9] on our data. Our 30.1M-parameter ModAR achieves a slightly higher observed average success rate (75% vs. 72%) despite using approximately 20$\times$ fewer training FLOPs and no pretraining. **(3)** We validate the resulting design on real-world bimanual manipulation using a mix of robot demonstrations and actionless human demonstrations.

II. Related work

A. World-action models

World-action models (WAMs) jointly model robot actions and future observations, enabling large-scale training on both action-labeled robot demonstrations and actionless demonstrations. Their designs vary along two primary axes: how predicted futures inform action generation and which representations of the future they predict.

WAM formulations differ in how predicted futures inform action generation (Fig. 3). Joint-generation models like DreamZero [10] co-denoise visual futures and actions simultaneously with cross-stream attention (Fig. 3(b)). Unified World Models [12] and Flex-π [9] do the same, but sample noise levels independently for different streams during training. Fast-WAM [1] instead prevents attention between future and action targets (Fig. 3(c)), using future-observation prediction as a training-time auxiliary objective and omitting it at deployment. UniPi [2] and VERA [3] instead fully denoise visual futures and then infer actions from the resulting frames, rather than co-denoising both streams. Building on this futures-then-actions ordering, ModAR autoregressively denoises multiple future modalities one at a time before denoising robot actions. Our controlled experiments compare each formulation (Fig. 3) for multimodal WAMs and we find ModAR performs best.

WAMs also differ in how they represent predicted futures. Most WAMs predict future RGB as image latents from frozen video VAEs [12], [13], [11], [10], [14], [1]. While this representation provides a convenient interface to pretrained video generators, reconstructing visual appearance does not explicitly prioritize the geometric, physical, and semantic structures most relevant to manipulation tasks. Recent work has therefore explored predicting alternative modalities either in place of RGB [7], [15] or alongside it [5], [8], [4]. Point tracks encode scene motion and correspondence independently of appearance, providing direct supervision for how task-relevant scene elements move through time [16], [17], [6]. DINO features are robust to appearance variation while encoding object semantics and scene structure [18]. Depth makes scene geometry explicit and provides direct supervision for the spatial reasoning required for predicting 3D actions [4]. Concurrent work Flex-π finds that jointly predicting multiple future modalities (RGB, 3D pointmaps, and DINO features) can improve performance over predicting future RGB [9].

Most existing WAMs build on pretrained video-generation models [11], [10], [9]. This initialization is highly effective, but makes it difficult to isolate the effects of WAM formulation, target representations, and actionless-data scale. We instead train all models from scratch and systematically study these factors in a controlled setting.

B. Multimodal generation

Predicting multiple representations of the same underlying signal can provide complementary supervision for representation learning. By co-training a shared backbone across multiple objectives, each objective can enrich and regularize the representations used by the others. Prior work finds that such co-training can improve per-task performance relative to single-task training, and recent systems have scaled this idea to many vision, language, and action tasks with strong results [19], [20], [21], [22]. Following this idea, we jointly predict multiple representations of the future so that action prediction can benefit from their inductive biases.

Beyond the choice of targets, their generation order can influence how much they help one another. Latent Forcing [23] generates image latents before pixels, allowing the latents to serve as a semantic "scratchpad" for generating fine-grained appearance. Similarly, Modality Forcing [24]

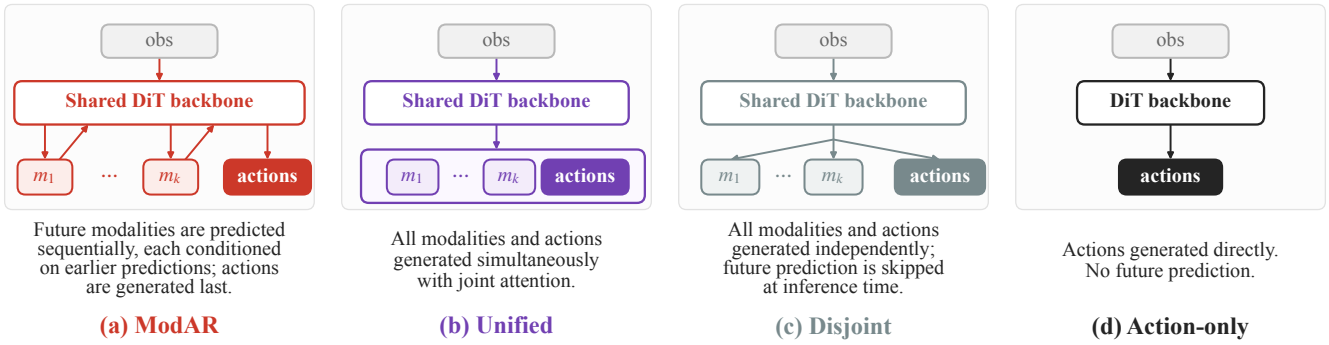


Fig. 3. **WAM formulations.** We draw block diagrams of the evaluated WAM formulations and the Action-only baseline, which differ only in how they couple future prediction with action prediction. ModAR (a) autoregressively denoises future modalities before denoising actions. Unified (b) represents joint-generation WAMs such as DreamZero and Cosmos Policy [10], [11], which denoise future-observation and action streams together. Disjoint (c) represents Fast-WAM [1], which predicts future observations and actions independently and omits future-observation prediction at inference time. Action-only (d) denoises actions directly without predicting future observations. Independent-noise (not drawn) is identical to Unified (b) except during training: rather than applying one shared noise level to all predicted streams, it samples each stream's flow timestep independently.

adapts a pretrained image generation model to jointly generate depth and finds that image models contain valuable priors which improve depth accuracy. These results suggest that *generating easier-to-predict modalities earlier can expose structure that simplifies subsequent predictions.* We extend this principle to multimodal WAMs by generating modalities autoregressively, starting with more structured modalities like point tracks and DINO features followed by increasingly detailed depth and RGB predictions, and finally actions.

III. Method

A. Problem definition

Given demonstrations spanning multiple manipulation tasks, our objective is to jointly model future multimodal observations and robot actions conditioned on the current observation, robot configuration, and task label. We consider the set of predicted future modalities $\mathcal{M} = \{\text{rgb}, \text{depth}, \text{dino}, \text{tracks}\}$. At decision time t, let $\mathbf{o}_t$ denote the multimodal observation. The complete conditioning information is $\mathbf{c}_t = (\mathbf{o}_t, \mathbf{q}_t, g)$, where $\mathbf{q}_t$ is the robot's proprioceptive configuration (joint positions and gripper opening) and g is the learned embedding associated with a discrete task label. For each $m \in \mathcal{M}$, let $\mathbf{Y}_t^m = (\mathbf{y}_{t+\Delta}^m, \dots, \mathbf{y}_{t+J\Delta}^m)$ denote J future targets in modality m, where Δ is the dynamics stride and $H = J\Delta$ is the prediction horizon. We predict actions at every control step: $\mathbf{A}_t = (\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+H-1})$. Thus, the final prediction target corresponds to the next decision point $t + H$, while $\mathbf{A}_t$ contains the H actions executed between t and $t + H$.

B. ModAR: Modality-Autoregressive World Modeling

Given the targets above, ModAR denoises one future modality at a time and uses each completed prediction as context for generating the next. We generate actions last, conditioning the policy on every generated modality. Formally, let $(m_1, \dots, m_K)$ be an ordering of $\mathcal{M}$ (or a subset of $\mathcal{M}$), and write $\mathbf{Y}_t = (\mathbf{Y}_t^{m_1}, \dots, \mathbf{Y}_t^{m_K})$ for the corresponding future targets. We model their joint distribution with actions as

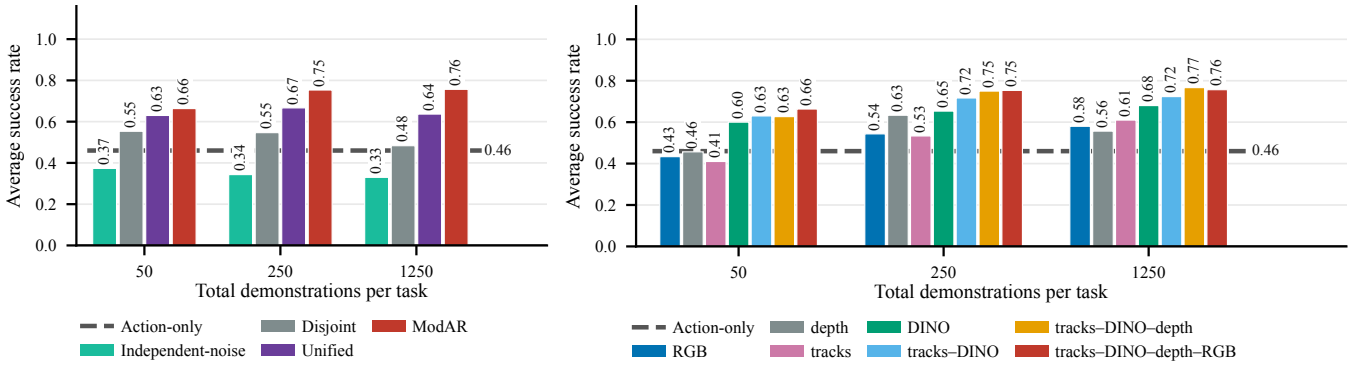
$$p_\theta(\mathbf{Y}_t, \mathbf{A}_t \mid \mathbf{c}_t) = p_\theta(\mathbf{A}_t \mid \mathbf{c}_t, \mathbf{Y}_t) \prod_{k=1}^{K} p_\theta(\mathbf{Y}_t^{m_k} \mid \mathbf{c}_t, \mathbf{Y}_t^{<k}), \tag{1}$$

where $\mathbf{Y}_t^{<k} = (\mathbf{Y}_t^{m_1}, \dots, \mathbf{Y}_t^{m_{k-1}})$ denotes the modalities preceding m_k in the generation order. The final action-prediction step acts as an inverse dynamics model (IDM), mapping the generated future observations to the actions that induce the predicted transitions. For actionless examples, we omit action prediction and supervise only the available future targets.

Modality tokenization. We patchify RGB and depth maps. We extract DINO tokens from the spatial patch tokens of a frozen DINOv2 encoder [18]. For point tracks, we initialize a 2D grid of queries at patch centers and track them with CoTracker3 [25]; we represent each point-time pair with a token that encodes its displacement from its initial position and its visibility. We linearly project each modality into the common token dimension. We add a learned modality embedding and apply axial rotary position embeddings (RoPE) [26] over time and the two spatial dimensions for visual tokens, and over time for action tokens.

Shared world-action backbone. As shown in Fig. 2, we process the resulting tokens from all modalities with a diffusion transformer (DiT) [27]. The DiT first applies several shared transformer blocks that attend across all causally available modality tokens. It then applies a small modality-specific expert stack, whose attention is restricted to that modality's stream, followed by a linear output head. The robot configuration, task embedding, and flow timesteps for each modality condition the transformer through adaLN. Thus, cross-modal information is fused in the shared blocks before within-modality specialization in the expert blocks.

Block-causal modality generation. Equation (1) factorizes the joint multimodal distribution into a sequence of modality-



(a) Formulation comparison (Figure 3)

(b) Predicted modalities

Fig. 4. **RoboTwin simulation results.** Success rates over six tasks with 50 trials per task. We train all models with 50 action-labeled demonstrations per task, then add actionless demonstrations to increase the total demonstration count from 50 to 250 to 1,250. (a) ModAR achieves the highest average success rate among the formulations at each data scale. (b) We find it beneficial to predict future tracks, DINO, and depth, while additionally predicting future RGB provides no consistent gain. RGB (dark blue) matches the typical WAM setting of predicting only future RGB and actions. Independent-noise is identical to Unified except during training: rather than applying one shared noise level to all predicted streams, it samples each stream's flow timestep independently.

conditional distributions. Our autoregressive sequence consists of modality blocks: we generate all tokens of one modality jointly before moving to the next modality. When generating block m_k, the model attends to the current observation and to the completed blocks $m_1, \ldots, m_{k-1}$; blocks later in the sequence are not available. At inference, we therefore denoise one block, re-embed the completed prediction as context, and then denoise the next block. We use the order tracks $\rightarrow$ DINO $\rightarrow$ depth $\rightarrow$ RGB. Intuitively, this orders the targets from compact, structured representations that are easier to predict toward increasingly high-dimensional and detailed representations.

During training, we supervise every stage in one pass by creating a clean context copy and a noisy prediction copy of each target modality. A block-causal mask lets the prediction copy for m_k attend only to the observation and context copies of $m_1, \ldots, m_{k-1}$, preventing target leakage.

Injecting context noise. Modality-autoregressive generation is susceptible to cascading error, as imperfections in earlier generations become inputs to all later predictions. Following Latent Forcing [23], we mitigate this by adding noise to context blocks during training. Specifically, when predicting m_k, we independently sample $\boldsymbol{\epsilon}_j^{\text{ctx}} \sim \mathcal{N}(0, I)$ and $\tau_j^{\text{ctx}} \sim \mathcal{U}(1-\beta, 1)$ for every context block m_j in every training example, and replace its clean target with $\widetilde{\mathbf{Y}}_{t,\text{ctx}}^{m_j} = \tau_j^{\text{ctx}} \mathbf{Y}_t^{m_j} + (1-\tau_j^{\text{ctx}})\boldsymbol{\epsilon}_j^{\text{ctx}}$. We apply this noise only during training, not during inference.

Training. We train every output stream with a JiT-style x-prediction objective [28]. Empirically, we find that replacing x-prediction with velocity (v) prediction is often unstable and can cause training to diverge. Velocity prediction can struggle in high-dimensional spaces; predicting the clean sample is well-conditioned when the data lie on a low-dimensional manifold, as with images, depth, and other visual representations [28], [24]. We use the same parameterization for robot action generation [29], [30]. For each supervised modality m, we independently sample $\boldsymbol{\epsilon}^m \sim \mathcal{N}(0, I)$ and a flow timestep τ_m, and form a linear flow-matching interpolant [31]

$$\widetilde{\mathbf{Y}}_t^m = \tau_m \mathbf{Y}_t^m + (1-\tau_m)\boldsymbol{\epsilon}^m. \tag{2}$$

The corresponding output head directly predicts the clean target $\widehat{\mathbf{Y}}_t^m$. We minimize the loss function:

$$\mathcal{L}_m(\theta) = \mathbb{E}\left[\frac{\|\widehat{\mathbf{Y}}_t^m - \mathbf{Y}_t^m\|_2^2}{\max(1-\tau_m, \delta_m)^2}\right], \tag{3}$$

where δ_m stabilizes the loss near the clean endpoint. We apply the same objective to the action chunk, replacing $(\mathbf{Y}_t^m, \widehat{\mathbf{Y}}_t^m, \tau_m, \delta_m)$ with $(\mathbf{A}_t, \widehat{\mathbf{A}}_t, \tau_{\text{act}}, \delta_{\text{act}})$. The total loss is

$$\mathcal{L}(\theta) = \sum_{k=1}^{K} \mathcal{L}_{m_k} + \mathcal{L}_{\text{act}}, \tag{4}$$

We include the action term only for examples with action labels.

Inference. We generate one modality at a time in the order $m_1, \ldots, m_K$. We initialize each stream from isotropic Gaussian noise and integrate it from $\tau = 0$ to 1 using the velocity implied by its clean prediction, $\mathbf{v}_\theta = (\widehat{\mathbf{Y}} - \widetilde{\mathbf{Y}})/(1-\tau)$; the same equation applies to the action stream with $\mathbf{Y}$ replaced by $\mathbf{A}$. The resulting clean tokens become context for generating the next modality, and we condition the final action chunk on the complete generated future. We cache keys and values for the current observation and all previously generated modalities to avoid recomputing them at every denoising step.

IV. Experiments

In simulation, we evaluate WAM formulations (Figure 3), actionless-data scaling, and predicted modality sets. We also compare with a large video-model-initialized WAM, Flex-π [9]. We then evaluate ModAR on real-world bimanual manipulation and learning from human demonstrations.

TABLE I

PER-TASK SUCCESS RATES BY WAM FORMULATION AT TOTAL-DEMONSTRATION SCALE D: ACTION-ONLY, INDEPENDENT-NOISE, DISJOINT, UNIFIED, AND MODAR.

Task	D	Action-only	Independent-noise	Disjoint	Unified	ModAR
dump bin	50	0.82	0.64	0.90	0.88	**0.92**
	250	—	0.68	0.92	0.92	**0.94**
	1250	—	0.64	0.78	0.94	**0.98**
pick bottles	50	0.30	0.40	0.44	**0.58**	0.48
	250	—	0.38	0.42	**0.60**	**0.60**
	1250	—	0.38	0.38	**0.62**	0.58
place bread	50	0.24	0.20	0.46	0.44	**0.52**
	250	—	0.14	0.48	0.50	**0.74**
	1250	—	0.28	0.22	0.44	**0.72**
put bottles	50	0.32	0.20	0.36	0.52	**0.64**
	250	—	0.20	0.36	0.62	**0.82**
	1250	—	0.18	0.42	0.56	**0.82**
stack bowls	50	0.64	0.36	0.58	**0.76**	**0.76**
	250	—	0.18	0.62	**0.76**	0.68
	1250	—	0.04	0.62	0.68	**0.80**
turn switch	50	0.44	0.44	0.58	0.60	**0.66**
	250	—	0.48	0.48	0.60	**0.74**
	1250	—	0.46	0.48	0.58	**0.64**
overall	50	0.46	0.37	0.55	0.63	**0.66**
	250	—	0.34	0.55	0.67	**0.75**
	1250	—	0.33	0.48	0.64	**0.76**

TABLE II

PER-TASK SUCCESS RATES BY PREDICTED-MODALITY SUBSET AT TOTAL-DEMONSTRATION SCALE D. MODALITIES COMPRISE RGB, DEPTH, POINT TRACKS, AND DINO FEATURES.

Task	D	Action-only	RGB	Depth	Tracks	DINO	Tracks + DINO	Tracks + DINO + Depth	Tracks + DINO + Depth + RGB
dump bin	50	0.82	0.80	0.90	0.90	**0.96**	0.92	0.92	0.92
	250	—	0.94	0.92	0.92	0.94	0.94	**0.96**	0.94
	1250	—	0.98	0.88	0.90	0.96	**1.00**	0.94	0.98
pick bottles	50	0.30	0.18	0.44	0.12	0.44	**0.58**	0.48	0.48
	250	—	0.32	0.48	0.48	**0.62**	**0.62**	0.58	0.60
	1250	—	0.30	0.40	0.42	0.62	**0.70**	0.66	0.58
place bread	50	0.24	0.26	0.38	0.10	0.44	0.52	**0.54**	0.52
	250	—	0.34	0.64	0.26	0.56	0.64	**0.76**	0.74
	1250	—	0.36	0.60	0.38	0.54	0.64	**0.72**	**0.72**
put bottles	50	0.32	0.20	0.12	0.30	**0.64**	0.48	0.48	**0.64**
	250	—	0.38	0.48	0.48	0.60	0.66	0.74	**0.82**
	1250	—	0.54	0.42	0.74	0.64	0.66	**0.90**	0.82
stack bowls	50	0.64	0.58	0.54	0.74	0.66	0.70	0.72	**0.76**
	250	—	0.72	0.74	0.64	0.66	**0.86**	0.78	0.68
	1250	—	0.66	0.64	0.66	**0.84**	0.74	0.80	0.80
turn switch	50	0.44	0.58	0.36	0.30	0.46	0.58	0.62	**0.66**
	250	—	0.56	0.54	0.42	0.54	0.58	0.68	**0.74**
	1250	—	**0.64**	0.40	0.56	0.48	0.60	0.58	**0.64**
overall	50	0.46	0.43	0.46	0.41	0.60	0.63	0.63	**0.66**
	250	—	0.54	0.63	0.53	0.65	0.72	**0.75**	**0.75**
	1250	—	0.58	0.56	0.61	0.68	0.72	**0.77**	0.76

A. Experimental setup

Simulation. We evaluate on six representative RoboTwin [32] tasks: "dump bin", "pick bottles", "place bread", "put bottles", "stack bowls", and "turn switch." Each task uses $D \in \{50, 250, 1250\}$ total training demonstrations. We retain 50 action-labeled demonstrations and use the remaining $D - 50$ as actionless demonstrations. For each method and data scale, we train one multitask model for all six tasks. We evaluate all models on 50 held-out initial conditions per task.

Real world. We evaluate on three challenging tabletop tasks—stacking cups, folding a crumpled towel, and placing an object in a drawer and closing the drawer—using bimanual YAM arms (Fig. 1). For each task, we collect 100 teleoperated robot demonstrations, 200 in-domain actionless human demonstrations, and 1,000 EgoDex demonstrations [33]. We use the corresponding EgoDex categories "stack/unstack cups," "basic fold," and "insert/remove drawer" for cup stacking, towel folding, and drawer placement, respectively. For each method and data mixture, we train one multitask model for all three real-world tasks. We evaluate each reported method and data mixture with 30 rollouts per task, with varied initial object poses.

Baselines. We compare representative WAM formulations (Fig. 3) within one controlled implementation. All methods share the same backbone, action-labeled data, and optimization budget.

Unified (Fig. 3(b)) jointly denoises all future-observation and action streams using one shared flow timestep, representing joint-generation WAMs such as DreamZero and Cosmos Policy [10], [11].

Disjoint (Fig. 3(c)) predicts each target independently, without attention between future-observation and action targets, as in Fast-WAM [1].

Independent-noise samples a separate flow timestep for every stream during training, as in Unified World Models and Flex-π [12], [9]; at inference, it uses the same simultaneous denoising process as Unified (Fig. 3(b)).

Action-only (Fig. 3(d)) predicts actions directly from observations without future-observation prediction, representing a typical flow-matching behavior cloning policy. Because it has no future-observation prediction objective, Action-only trains only on action-labeled data.

B. Simulation results

Figure 4 compares WAM formulations and predicted-modality subsets as the amount of actionless data varies.

Formulation comparison. Figure 4(a) compares formulations trained to predict all four modalities; Table I gives the per-task breakdown. ModAR achieves the highest average success rate at every data scale. We hypothesize that early modalities act as "scratchpads" for later ones: generating coarser or easier targets first provides structured context for more detailed targets [23].

Independent-noise generation performs poorly in our from-scratch experiments. Because independently sampled noise levels rarely match the synchronized test-time denoising schedule, we hypothesize that the model receives insufficient training signal near its inference regime.

Scaling with actionless data. ModAR benefits most from additional actionless data: adding 1,200 actionless demonstrations improves the average success rate from 66% to 76% (10 percentage points), compared with a mere 1% improvement for Unified. Disjoint generation outperforms Action-only prediction on average, but its performance decreases as we add actionless data. One possible explanation is that with more actionless data, the shared representation becomes increasingly shaped by the future-observation prediction objective, causing negative transfer to the action prediction objective.

Modality comparison. Figure 4(b) compares ModAR variants that predict each target modality individually against

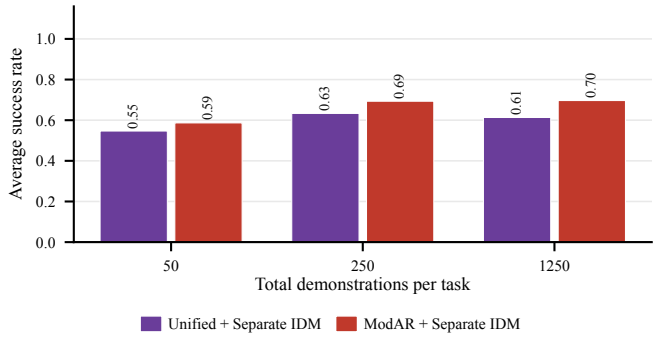


Fig. 5. **Evaluating predicted futures with a separate IDM.** Recall from Sec. III that ModAR's final action-prediction step acts as an inverse-dynamics model (IDM), mapping its future-observation predictions to actions. In contrast, Unified denoises futures and actions jointly, so its action prediction conditions on partially noisy predicted futures. To remove this asymmetry, we discard both models' native action predictions and pass their predicted futures to the same separately trained IDM. ModAR still outperforms Unified, showing that its predicted futures are themselves more useful for predicting actions.

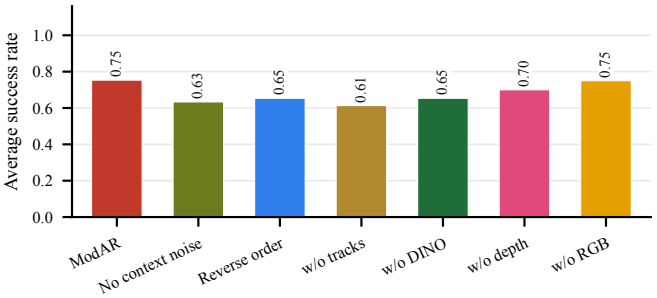


Fig. 6. **ModAR ablations in RoboTwin.** Average success rate over six tasks, with 250 total demonstrations per task and 50 action-labeled demonstrations. We ablate context noise (*No context noise*), generation order (*Reverse order*), and each individual future-observation modality. Results suggest that context noise is critical for robust autoregressive generation. Generating modalities in reverse order also reduces performance, supporting our hypothesis that more structured modalities act as "scratchpads" for more detailed modalities. Removing future RGB prediction does not reduce performance, suggesting that RGB contributes the least of the four future-observation modalities.

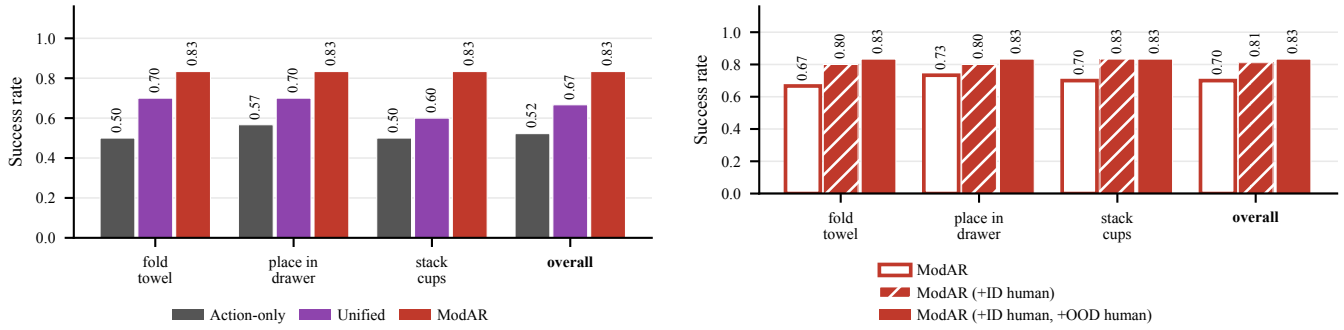
variants that predict progressively larger modality sets, following the generation order point tracks, DINO, depth, and RGB; Table II gives the per-task breakdown. Across data scales, performance generally holds or increases with each added modality, showing that ModAR can effectively combine the benefits of predicting multiple modalities. However, the best modality set naturally varies across tasks because different features define each task, and some modalities represent those features better than others; as a result, additional modalities do not always help. In particular, additionally predicting future RGB on top of the first three modalities provides no consistent gain. We hypothesize that predicting future RGB introduces high-variance appearance details while adding little information beyond the more structured targets in our setting.

Separate inverse-dynamics model. Recall from Sec. III that ModAR's final action-prediction step acts as an inverse-dynamics model (IDM), mapping its fully generated future-observation predictions to actions. Unified instead predicts actions while its predicted futures are still being jointly denoised. Thus, ModAR could outperform Unified simply because its action predictor conditions on more informative predicted futures. To test this, we evaluate both models using the same separately trained IDM in place of their native action predictors. We train this IDM to map ground-truth future observations to actions, and apply context noise to its inputs during training. We evaluate the best-performing ModAR and Unified checkpoints on 50 held-out initial conditions per task using this separate IDM. At each prediction step, each model generates its future-observation predictions and actions normally. We then discard its native action prediction, pass its predicted futures to the separate IDM, and execute the resulting actions (Fig. 5). ModAR still outperforms Unified with the separate IDM, suggesting that its predicted futures are themselves more useful for predicting actions.

Sampling-step comparison. ModAR generates each modality autoregressively, and performs 8 Euler integration steps per modality—40 steps across four future-observation streams and actions—whereas Unified, Independent-noise, and Disjoint use only 8 steps in total. To test whether ModAR's gains arise simply from its larger sampling budget, we re-evaluate each formulation at $D = 250$ with 40 Euler steps, matching ModAR's total number of sampling steps. Additional steps do not close the gap: Unified decreases from 67% to 59%, Disjoint from 55% to 54%, and Independent-noise increases from 34% to 37%, compared to ModAR's 75%.

Comparison to a video-model-initialized WAM. In addition to our controlled from-scratch experiments, we compare against concurrent work Flex-π [9] at $D = 250$. We adapt the official implementation to our single head-camera setting and to an action horizon of $H = 16$, with visual targets at $t + \{4, 8, 12, 16\}$. Following their recipe, we initialize the video backbone from Wan2.2-TI2V-5B [34], [35] and interpolate those weights to a smaller hidden size for the action expert, then full-fine-tune all trainable Flex-π components while keeping the VAE, text encoder, and DINOv3 encoder [36] frozen. On actionless demonstrations we retain every visual objective (RGB latents, pointmaps, and DINO features) and mask the action loss.

We train with global batch size 288 and evaluate on the same 50 held-out initial conditions per task used in our other experiments. At inference we jointly denoise all visual streams and actions, i.e., Flex-π's full multi-stream generation mode. At its final checkpoint after 30,000 steps, Flex-π attains an average success rate of **72%**. Compared to our trained-from-scratch tracks–DINO–depth ModAR variant, which achieves 75% average success, Flex-π has approximately $200\times$ as many parameters (6B vs 30.1M) and uses approximately $20\times$ as many training FLOPs. This comparison shows that, on these in-distribution tasks, a compact WAM trained from scratch can perform as well or better than a much larger video-model-initialized WAM using substantially less training compute. We note that this is a system-level comparison rather than a controlled architectural comparison, since the models



(a) Formulation **(b) Actionless human data**

Fig. 7. **Real-world results.** Success rates on real-world tasks, with 30 trials per model per task. (a) All methods use 100 robot demonstrations per task; ModAR and Unified additionally use 200 in-domain actionless human demonstrations and 1,000 out-of-domain EgoDex demonstrations. ModAR achieves the highest success rate on all three tasks. (b) We train separate ModAR models with just 100 robot demonstrations per task, then adding 200 in-domain actionless human demonstrations (used to supervise future-observation predictions, but not action prediction), and then also adding 1,000 out-of-domain human demonstrations from EgoDex (used similarly). The average success rate progressively improves from 70.0% to 81.1% to 83.3%.

differ in scale, pretraining, target modalities, and training recipe.

Inference latency. On a single NVIDIA GeForce RTX 5090 GPU, end-to-end inference for ModAR takes 147.9 ms (6.76 Hz) when generating all four future-observation modalities and actions.

C. Ablations

Figure 6 ablates the ModAR design at the 250-demonstration scale, where the full model achieves a 75% average success rate. **Context noise.** First, we remove the context noise added to previously generated modalities during training (*No context noise*). The success rate falls to 63%, showing that context noise during training is critical for preventing errors from compounding across successive modalities.

Modality order. We also reverse the future-modality order to RGB $\rightarrow$ depth $\rightarrow$ DINO $\rightarrow$ tracks (actions are still generated last). This lowers the average success rate to 65%, supporting our choice to generate compact, structured representations before increasingly detailed ones. We leave a full systematic comparison of modality orderings to future work.

Ablating modalities. Finally, we measure the contribution of each future-observation modality by removing one modality at a time from the full model. Removing tracks (*w/o tracks*), DINO (*w/o DINO*), or depth (*w/o depth*) lowers the average success rate to 61%, 65%, and 70%, respectively, whereas removing future RGB prediction leaves it unchanged. Thus, predicting RGB contributes the least of the four modalities to success rate in this setting.

D. Real-world experiments

Because adding RGB provided no consistent benefit in simulation while increasing training and generation costs, we train all real-world WAMs to observe and predict only tracks, DINO, and depth.

Formulation comparison. Results in Fig. 7(a) mirror the simulation results: across 30 trials per task, ModAR achieves the highest success rate on all three tasks and an overall success rate of 83.3%, compared with 66.7% for Unified and 52.2% for Action-only.

Effect of adding human demonstration data. We compare ModAR trained on only the 100 teleoperated robot demonstrations per task against adding 200 in-domain human demonstrations and then also adding 1,000 out-of-domain EgoDex demonstrations. These additions improve the average success rate from 70.0% to 81.1% and 83.3%, respectively (Fig. 7(b)), indicating that ModAR can benefit from actionless data collected across embodiments and domains.

E. Implementation details

Inputs and outputs. Models condition on a single-camera 168×224 observation and predict future observations with horizon $H = 16$ and dynamics stride $\Delta = 8$, yielding $J = 2$ sparse targets at $t + 8$ and $t + 16$, together with a dense H-step action chunk; policies replan every H steps. Actions and robot configurations are 14-dimensional absolute dual-arm joint configurations. RGB, depth, and track queries use a 12×16 grid of 14×14 patches. We obtain point tracks by tracking the patch-center queries with CoTracker3 [25], and extract DINO targets with a frozen DINOv2 ViT-S/14 encoder [18].

Architecture. The shared DiT contains six width-384 transformer blocks with six attention heads (ViT-S), followed by two width-384 expert blocks for DINO, depth, and RGB, one width-128 block for tracks, and two width-128 blocks for actions.

Optimization. We use AdamW with learning rate 10^{-4}, betas $(0.9, 0.95)$, weight decay 0.1, gradient clipping at 1.0, and global batch size 48. Training uses bfloat16, an EMA with decay 0.999, and a 48,000-sample linear warmup followed by a constant learning rate. We train all methods for 1.2M optimizer steps. At each optimizer step, we sample equal-sized batches from the action-labeled and actionless pools and weight their losses equally. We evaluate checkpoints every 100,000 optimizer steps and report the best-checkpoint success rate on 50 held-out initial conditions per task.

Flow and sampling. For ModAR, we set the maximum context-noise level to $\beta = 0.5$, set $\delta_m = \delta_{\text{act}} = 0.05$, use

unit loss weights, and sample logit-normal flow timesteps with $(\mu, \sigma) = (-2, 1), (0, 1), (-1, 1.6), (-1, 1), (-1, 1)$ for DINO, tracks, depth, RGB, and actions, respectively. Unified uses one shared flow timestep with $(\mu, \sigma) = (-1, 1)$; Independent-noise samples each stream's modality-specific distribution independently. We integrate each generated stream with eight Euler steps.

Real-world data collection. We collect robot demonstrations by teleoperation with paired teacher arms using the RAIDEN toolkit [37]. A fixed third-person ZED stereo camera provides RGB observations and stereo depth for both robot and actionless human demonstrations.

V. Conclusion

We introduce ModAR, a world-action model that autoregressively denoises multiple future-observation modalities before predicting actions. In a controlled simulation study, this formulation achieves the highest average success rate among representative existing WAM formulations and improves the most as we add actionless data; on three real-world tasks, it achieves the highest success rate among baselines and also improves with human video demonstrations. On average, point tracks, DINO features, and depth provide complementary gains, whereas additionally predicting future RGB provides no consistent benefit in our experiments. These findings highlight modality-autoregressive prediction as a promising alternative to relying solely on future RGB prediction for world-action modeling.

VI. Limitations

Our experiments cover a limited number of tasks and use discrete task labels rather than language instructions, so they do not establish broad generalization across tasks, objects, or scenes. Sequentially generating multiple modalities also increases inference latency relative to simultaneous or action-only generation. Future work should explore larger-scale experiments with broader task diversity, generalization evaluation, and training on heterogeneous internet-scale actionless data. Future work can also explore the best order for generating modalities, which may depend on the task or specific scenario.

VII. Acknowledgment

We thank Narek Harutyunyan for assistance in data collection.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant Nos. DGE2140739 and DGE2631988. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

This work used Bridges-2 at Pittsburgh Supercomputing Center through allocation CIS260202p from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296 [38], [39].

References

[1] T. Yuan, Z. Dong, Y. Liu, and H. Zhao, "Fast-WAM: Do World Action Models Need Test-time Future Imagination?" Mar. 2026, arXiv:2603.16666 [cs.CV].

[2] Y. Du *et al.*, "Learning Universal Policies via Text-Guided Video Generation," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, 2023, pp. 9156–9172.

[3] S. L. Li *et al.*, "Turning Video Models into Generalist Robot Policies," May 2026, arXiv:2605.27817 [cs.RO].

[4] Y. Li *et al.*, "WAM4D: Fast 4D World Action Model via Spatial Register Tokens," Jun. 2026, arXiv:2606.14048 [cs.CV].

[5] J. Guan, W. Zhao, Y. Pei, Z. Chen, A. Solin, and J. Kannala, "Point Tracking Improves World Action Models," May 2026, arXiv:2605.23856 [cs.RO].

[6] A. Hung, B. P. Duisterhof, and J. Ichnowski, "3PoinTr: 3D Point Tracks for Learning Manipulation from Unconstrained Human Videos," Mar. 2026, arXiv:2603.08485 [cs.RO].

[7] G. Zhou, H. Pan, Y. LeCun, and L. Pinto, "DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, ser. Proc. Mach. Learn. Res., vol. 267, 2025, pp. 79 115–79 135.

[8] M. Wang *et al.*, "ST-WAM: Semantic-Temporal World Action Model for Robust Manipulation under Visual Distribution Shifts," Jul. 2026, arXiv:2607.28993 [cs.RO].

[9] G. Yan *et al.*, "Flex-π: A Multi-Stream World-Action Model with Compute Flexibility," Aug. 2026, arXiv:2608.10860 [cs.RO].

[10] S. Ye *et al.*, "World Action Models are Zero-shot Policies," Feb. 2026, arXiv:2602.15922 [cs.RO].

[11] M. J. Kim *et al.*, "Cosmos Policy: Fine-Tuning Video Models for Visuomotor Control and Planning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2026.

[12] C. Zhu, R. Yu, S. Feng, B. Burchfiel, P. Shah, and A. Gupta, "Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets," in *Proc. Robot.: Sci. Syst. (RSS)*, 2025.

[13] J. Pai *et al.*, "mimic-video: Video-Action Models for Generalizable Robot Control Beyond VLAs," in *Proc. Robot.: Sci. Syst. (RSS)*, 2026.

[14] L. Li *et al.*, "Causal World Modeling for Robot Control," in *Proc. Robot.: Sci. Syst. (RSS)*, Sydney, Australia, Jul. 2026.

[15] B. Li, X. Yin, M. Lin, Y. Zhang, and D. Xu, "EgoWAM: World Action Models Beyond Pixels with In-the-Wild Egocentric Human Data," Jul. 2026, arXiv:2607.08436 [cs.RO].

[16] C. Wen *et al.*, "Any-point Trajectory Modeling for Policy Learning," in *Proc. Robot.: Sci. Syst. (RSS)*, 2024.

[17] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani, "Track2Act: Predicting Point Tracks from Internet Videos enables Generalizable Robot Manipulation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2024, pp. 306–324.

[18] M. Oquab *et al.*, "DINOv2: Learning Robust Visual Features without Supervision," *Trans. Mach. Learn. Res.*, 2024.

[19] R. Bachmann, D. Mizrahi, A. Atanov, and A. Zamir, "MultiMAE: Multi-modal Multi-task Masked Autoencoders," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2022, pp. 348–367.

[20] D. Mizrahi *et al.*, "4M: Massively Multimodal Masked Modeling," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 36, 2023, pp. 58 363–58 408.

[21] J. Lu *et al.*, "Unified-IO 2: Scaling Autoregressive Multimodal Models with Vision, Language, Audio, and Action," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024, pp. 26 439–26 455.

[22] NVIDIA *et al.*, "Cosmos 3: Omnimodal World Models for Physical AI," Jun. 2026, arXiv:2606.02800 [cs.CV].

[23] A. Baade *et al.*, "Latent Forcing: Reordering the Diffusion Trajectory for Pixel-Space Image Generation," Feb. 2026, arXiv:2602.11401 [cs.CV].

[24] B. P. Duisterhof, D. Ramanan, J. Ichnowski, J. Johnson, and K. Park, "Modality Forcing for Scalable Spatial Generation," Jun. 2026, arXiv:2606.13676 [cs.CV].

[25] N. Karaev, Y. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht, "CoTracker3: Simpler and Better Point Tracking by Pseudo-Labelling Real Videos," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2025, pp. 6013–6022.

[26] J. Su, M. Ahmed, Y. Lu, S. Pan, B. Wen, and Y. Liu, "RoFormer: Enhanced Transformer with Rotary Position Embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.

[27] W. Peebles and S. Xie, "Scalable Diffusion Models with Transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 4195–4205.

[28] T. Li and K. He, "Back to Basics: Let Denoising Generative Models Denoise," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2026, pp. 36 115–36 125.
[29] Y. Yang *et al.*, "ABot-M0: VLA Foundation Model for Robotic Manipulation with Action Manifold Learning," Apr. 2026, arXiv:2602.11236 [cs.CV].
[30] J. Chen *et al.*, "MV-WAM: Manifold-Aware World Action Model with Value Augmentation," Jun. 2026, arXiv:2606.21088 [cs.RO].
[31] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow Matching for Generative Modeling," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2023.
[32] T. Chen *et al.*, "RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation," Jun. 2025, arXiv:2506.18088 [cs.RO].
[33] R. Hoque, P. Huang, D. J. Yoon, M. Sivapurapu, and J. Zhang, "EgoDex: Learning Dexterous Manipulation from Large-Scale Egocentric Video," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2026.
[34] Team Wan *et al.*, "Wan: Open and Advanced Large-Scale Video Generative Models," Apr. 2025, arXiv:2503.20314 [cs.CV].
[35] Team Wan, "Wan2.2-TI2V-5B," Model release, Jul. 2025.
[36] O. Siméoni *et al.*, "DINOv3," *Trans. Mach. Learn. Res.*, 2026.
[37] S. Iwase, P. Miller, J. Yao, K. M. Jatavallabhula, and S. Zakharov, "RAIDEN: A Toolkit for Policy Learning with YAM Bimanual Robot Arms," 2026.
[38] S. T. Brown, P. Buitrago, E. Hanna, S. Sanielevici, R. Scibek, and N. A. Nystrom, "Bridges-2: A Platform for Rapidly-Evolving and Data Intensive Research," in *Practice and Experience in Advanced Research Computing (PEARC)*, 2021, pp. 1–4.
[39] T. J. Boerner, S. Deems, T. R. Furlani, S. L. Knuth, and J. Towns, "ACCESS: Advancing Innovation: NSF's Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support," in *Practice and Experience in Advanced Research Computing (PEARC '23)*, 2023, pp. 1–4.