FutureWorlds: LEARNING ROBOTIC WORLD MODELS FROM ALTERNATIVE FUTURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Robotic world models predict action-conditioned future scenes, providing a foundation for understanding action outcomes. However, turning alternative predictions into useful learning signals remains challenging: similar candidates limit informative quality comparisons, while diverging trajectories require persistent maintenance of their individual histories. We introduce `FutureWorlds`, a framework that unifies candidate construction, history maintenance, and learning from relative quality. Built on a multimodal discrete autoregressive model, `FutureWorlds` uses diverse beam search during reinforcement learning to construct candidate futures that balance confidence and diversity. Candidate-specific bounded memory preserves scene states and ensures that generation and policy scoring use matching histories. We further propose `MemSPO` (Memory-Conditioned Search-Guided Policy Optimization), which converts video trajectory rewards into group-relative advantages to optimize the world model. On RT-1, BridgeV2, and RoboCasa, `FutureWorlds` reduces LPIPS for 32-frame predictions by 14.78%, 20.84%, and 9.12%, respectively, relative to the strongest baseline on each dataset. Under fixed evaluation configurations, only 200 `MemSPO` updates further improve generation quality and support continued prediction beyond the training horizon. Memory ablations, decoding sensitivity analysis, and optical-flow evaluation show that these gains extend beyond visual quality to more accurate motion prediction and more consistent object states. Project page and code: https://anonymous.4open.science/w/FutureWorlds-BAEC/.

1 INTRODUCTION

Robotic world models predict how scenes evolve in response to actions, providing a foundation for action-outcome prediction (Finn et al., 2016), visual planning (Ebert et al., 2018; Hafner et al., 2019), behavior learning (Hafner et al., 2020), and policy evaluation (Yin et al., 2026; GigaWorld Team et al., 2026). Given the same observations, action sequence, and instruction, a model can propose multiple predictions of the future. These *alternative futures* represent different hypotheses about the scene's evolution, with varying robot motions, object states, and prediction quality. Their differences offer a learning opportunity: by constructing useful alternatives and comparing their outcomes against recorded observations, a world model can learn from the predictions it generates.

Most autoregressive world models learn from recorded trajectories through supervised training. Their reliance on ground-truth histories during training differs from the self-generated histories available at inference, allowing errors to accumulate along a rollout (Ross et al., 2011; Bengio et al., 2015). Token-level likelihood also does not directly optimize complete generated sequences (Ranzato et al., 2016; Rennie et al., 2017). Figure 1 illustrates this challenge: supervised fine-tuning (SFT) can leave predictions with insufficient robot motion or inaccurate poses. Reinforcement learning (RL) connects generated outputs to sequence-level rewards (Black et al., 2024). In world modeling, RLVR-World uses decoded prediction metrics (Wu et al., 2025a), while RLIR introduces inverse-dynamics rewards for action following (Ye et al., 2025). These advances enable learning from trajectory feedback, but the resulting learning signal also depends on *which futures the model compares*. More candidates alone do not guarantee useful comparative feedback. Effective post-training therefore requires informative candidate differences and history-consistent comparisons that turn alternative futures into reliable learning signals.

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.

Learning from alternative futures requires coordinated candidate construction and history management. Candidate quality and diversity depend on the generation strategy (Holtzman et al., 2020): independent sampling does not explicitly coordinate these properties across candidates, while conventional beam search can concentrate on nearly identical sequences (Vijayakumar et al., 2018). A useful candidate set therefore needs to balance model confidence with distinct predictions. As these predictions unfold, each branch develops its own robot and object states. Retaining the corresponding history becomes especially important when prediction extends beyond the pretraining horizon, motivating bounded memory for sustained generation (Dai et al., 2019; Wu et al., 2025b; Zheng et al., 2026). Learning introduces a further requirement: policy scoring must condition on the same retained history that produces each candidate. These considerations connect three roles—*constructing* informative alternatives, *maintaining* their individual histories, and *learning* from their relative quality.



Figure 1: ***Learning from alternative futures.*** RT-1 comparison at +32; metrics use full images. `FutureWorlds` constructs diverse futures, maintains their histories, and learns through `MemSPO` by comparing trajectory rewards.

How can we construct informative alternative futures, maintain their individual histories, and learn from their relative quality to improve robotic world models?

We introduce `FutureWorlds`, a framework that learns robotic world models by constructing, maintaining, and comparing alternative futures (Figure 2). We instantiate the framework with a multimodal discrete autoregressive model (van den Oord et al., 2017; Wu et al., 2024), whose token probabilities support candidate expansion and trajectory scoring under visual, action, and language conditioning. *To construct alternatives*, diverse beam search combines model likelihood with cross-candidate differences during RL training. *To maintain their histories*, candidate-specific memory retains scene and state anchors alongside rolling context, supporting longer predictions within a bounded memory budget. Search and memory operate together as each candidate evolves, and generation and policy scoring use matching histories. *To compare and learn*, `MemSPO` (Memory-Conditioned Search-Guided Policy Optimization) converts video trajectory rewards into group-relative advantages and updates the model through a search-guided surrogate based on GRPO (Shao et al., 2024) and clipped policy optimization (Schulman et al., 2017). Candidate quality differences thus become feedback for improving the model that generates them.

Our evaluation covers RT-1 (Brohan et al., 2023), BridgeData V2 (Walke et al., 2023), and RoboCasa (Nasiriany et al., 2026). On 32-frame predictions, `FutureWorlds` reduces LPIPS by 14.78%, 20.84%, and 9.12% relative to the strongest baseline on each dataset. Under fixed evaluation configurations, only 200 `MemSPO` updates improve generation and support prediction beyond the training horizon. Memory ablations, decoding sensitivity analysis, and motion evaluation further assess historical context, robustness, and dynamic consistency. Our contributions are summarized as follows:

- ❶ *Learning from Alternative Futures.* We introduce `FutureWorlds`, a framework that organizes model-generated futures into training experiences through coordinated candidate construction, history maintenance, and relative comparison.
- ❷ *Search-Guided, History-Consistent Learning.* We propose `MemSPO`, a GRPO-based post-training algorithm that combines diverse candidate search with candidate-specific memory and matches the histories used for generation and policy scoring.

❸ *Experimental Evaluation.* We evaluate generation quality and long-horizon prediction, analyze memory and decoding robustness, and assess motion consistency and inference resource requirements. LPIPS decreases by 9.12–20.84% versus the strongest baseline on each dataset.

2 Related Work

Robotic World Models. Robotic world models predict action consequences for planning and policy learning. Early work uses action-conditioned video prediction for visual control (Finn et al., 2016; Ebert et al., 2018); PlaNet (Hafner et al., 2019) and Dreamer (Hafner et al., 2020) learn latent dynamics. UniSim (Yang et al., 2024) combines language and low-level controls for visual simulation, while Vid2World (Huang et al., 2026) adapts video diffusion for interactive prediction. VideoGPT (Yan et al., 2021) and iVideoGPT (Wu et al., 2024) develop discrete autoregressive models, with iVideoGPT integrating observations, actions, and rewards through compressive tokenization. `FutureWorlds` uses discrete autoregression to construct and compare alternative futures under shared visual, action, and language conditions.

Memory for Long-Horizon World Modeling. Transformer-XL (Dai et al., 2019) propagates segment-level history, while StreamingLLM (Xiao et al., 2024) retains initial attention sinks and recent context. StreamingT2V (Henschel et al., 2025) combines short- and long-term memory; FramePack (Zhang & Agrawala, 2025) packs historical frames into a fixed context budget. Wu et al. (2025b) introduce geometry-grounded memory for scene revisits. Ctrl-World (Guo et al., 2026) and Mem-World (Zheng et al., 2026) retrieve robotic observations through pose conditioning and spatiotemporal surfel indexing, respectively. `FutureWorlds` maintains bounded, candidate-specific histories for long-horizon prediction and uses matching contexts for generation and policy scoring, preserving distinct conditioning as candidate futures diverge.

Search-Guided Reinforcement Learning for World Models. MIXER (Ranzato et al., 2016) and SCST (Rennie et al., 2017) optimize sequence metrics through policy gradients; DDPO (Black et al., 2024) extends reward optimization to diffusion models. RLVR-World (Wu et al., 2025a) rewards decoded prediction quality, while RLIR (Ye et al., 2025) evaluates action following through inverse dynamics. Candidate construction determines which predictions provide this feedback. Diverse Beam Search (Vijayakumar et al., 2018) reduces redundancy, Expert Iteration (Anthony et al., 2017) alternates search and policy learning, and BGRPO (Lee et al., 2025) combines beam-generated candidates with rank-aware rewards for algebraic tasks. QWM (Dong et al., 2026) searches predicted futures for robotic action selection and learns policies and values from real transitions. `FutureWorlds` instead searches futures under a fixed action sequence. `MemSPO` updates the video predictor from their relative quality using history-consistent scoring and a search-guided surrogate based on GRPO's group-relative advantages and clipped updates (Shao et al., 2024).

3 Method

Figure 2 presents `FutureWorlds`, which learns from alternative futures through coordinated candidate construction, history maintenance, and relative comparison. A multimodal autoregressive world model first learns conditional video prediction through supervised training. During post-training, diverse search constructs candidate futures under shared visual, action, and language conditions. Each candidate retains its own history, with bounded persistent memory supporting continuation beyond the pretraining horizon. `MemSPO` compares decoded trajectories through video rewards and converts their relative quality into world-model updates. Generation and policy scoring use matching candidate histories, while true future frames enter only reward computation during post-training.

3.1 Problem Formulation

Robotic world modeling predicts how a scene evolves under actions. We use a GPT-style autoregressive model to represent alternative future trajectories conditioned on the same visual observations, action sequence, and optional language instruction. Let $\mathbf{z}_{1:H}$ denote the discrete visual sequence for H future frames, with N dynamic tokens per frame. Given initial visual conditions $\mathbf{c}_v$, robot actions $\mathbf{a}_{0:H-1}$, and an optional language instruction ℓ, the world model factorizes the conditional

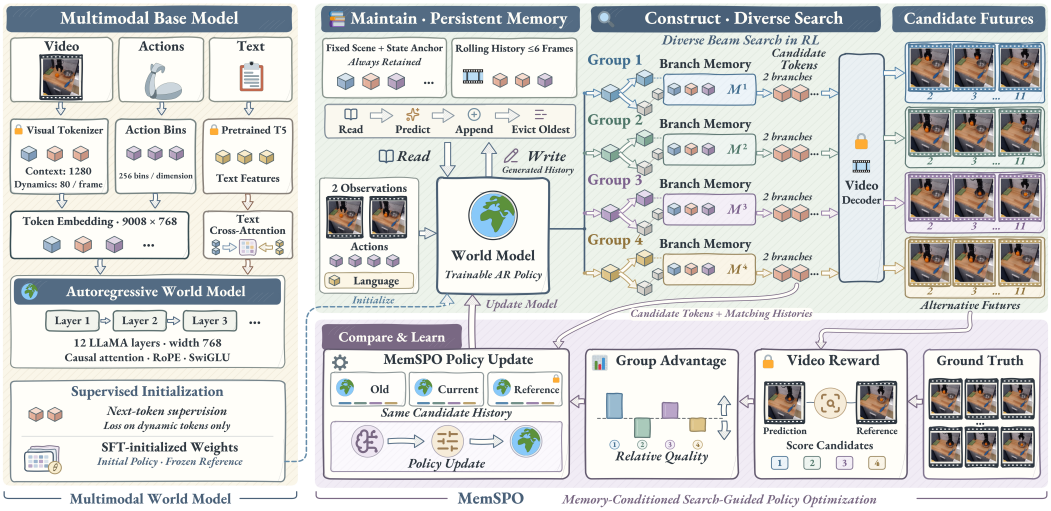

Figure 2: ***Overview of*** `FutureWorlds`**.** Supervised initialization trains the multimodal world model. `MemSPO` searches alternative futures, maintains separate histories, and learns from relative trajectory quality. Generation and policy scoring use matching histories.

distribution as

$$\pi_\theta(\mathbf{z}_{1:H} \mid \mathbf{c}_v, \mathbf{a}_{0:H-1}, \ell) = \prod_{t=1}^{H} \prod_{j=1}^{N} \pi_\theta(z_{t,j} \mid \mathbf{c}_v, \mathbf{z}_{<t}, \mathbf{z}_{t,<j}, \mathbf{a}_{<t}, \ell). \tag{1}$$

Here, $\mathbf{z}_{<t}$ contains tokens from preceding frames, $\mathbf{z}_{t,<j}$ is the within-frame prefix, and $\mathbf{a}_{<t} = \mathbf{a}_{0:t-1}$ contains actions available before frame t. In particular, a_{t-1} conditions the transition to that frame. A visual decoder maps predicted tokens back to video. This factorization supports both candidate expansion and trajectory scoring. Our goal is to construct alternative predictions under the same conditions, maintain their histories within a bounded context, and learn from their relative quality.

3.2 Multimodal Autoregressive World Model

The multimodal model provides the representation for constructing and scoring candidate futures. Following iVideoGPT's conditional video compression (Wu et al., 2024), we use more context tokens for the initial scene and fewer dynamic tokens for subsequent changes, reducing repeated encoding of static information. Context and conditional dynamics encoders produce discrete representations through Finite Scalar Quantization (FSQ; Mentzer et al., 2024), and a visual decoder reconstructs frames:

$$C = Q_c\left(E^c_\phi(o_{\text{ref}})\right), \qquad \mathbf{z}_t = Q_d\left(E^d_\phi(o_t \mid o_{\text{ref}})\right), \qquad \tilde{o}_t = D_\phi(C, \mathbf{z}_t). \tag{2}$$

Here, o_{ref} is the initial reference frame, ϕ denotes tokenizer parameters, and Q_c and Q_d map continuous features to discrete tokens with straight-through gradients.

The tokenizer jointly optimizes pixel, perceptual, and adversarial losses:

$$\mathcal{L}_{\text{tok}}(\phi) = \lambda_{\text{rec}}\mathcal{L}_1 + \lambda_{\text{perc}}\mathcal{L}_{\text{LPIPS}} + \lambda_{\text{adv}}\mathcal{L}_{\text{GAN}}. \tag{3}$$

Pixel and perceptual losses supervise both reference and dynamic frames; the adversarial term further constrains visual quality. The tokenizer remains frozen throughout world-model SFT and RL. Dataset-specific initialization and adaptation details are given in Appendix B.1.

As Figure 2 (left) shows, actions undergo per-dimension normalization and discretization, then form a temporally ordered causal sequence with visual tokens. A frozen T5 encoder (Raffel et al., 2020) supplies language features through cross-attention. Supervised fine-tuning conditions on ground-truth histories and applies negative log-likelihood only to target dynamic tokens:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{\mathcal{D}}\left[\frac{1}{HN}\sum_{t=1}^{H}\sum_{j=1}^{N}\log \pi_\theta(z_{t,j} \mid \mathbf{c}_v, \mathbf{z}_{<t}, \mathbf{z}_{t,<j}, \mathbf{a}_{<t}, \ell)\right]. \tag{4}$$

This objective jointly updates the autoregressive backbone, action embeddings, and text cross-attention modules. The resulting model initializes the post-training policy, with a frozen copy serving as the reference policy.

3.3 Candidate-Specific Persistent Memory

As candidate futures diverge, each accumulates its own prediction history. Persistent memory supports long-horizon continuation within a bounded context by retaining the initial scene, a state anchor, and recent predictions for each candidate. Let $\mathbf{c}_v = (C, \mathbf{z}_{\text{anc}})$ and let $\tilde{\mathbf{a}}_s$ denote discrete action tokens. Before predicting frame t, candidate i has memory

$$M_t^{(i)} = \left[C, \mathbf{z}_{\text{anc}}, \tilde{\mathbf{a}}_0, \left\{ (\hat{\mathbf{z}}_s^{(i)}, \tilde{\mathbf{a}}_s) \right\}_{s \in \mathcal{I}_t} \right], \quad \mathcal{I}_t = \{\max(1, t-W), \ldots, t-1\}. \tag{5}$$

The rolling history is temporally ordered and initially empty; its window length W satisfies the token budget $|M_t^{(i)}| \le B$. Scene and state anchors persist throughout prediction. After each frame, memory appends its dynamic tokens and corresponding action, excluding older history from the model context once the window is full. Language remains available independently of this history.

During search, branches inherit their parents' memories, which are reordered together with candidate selection. To learn from each resulting future under the conditions that generate it, policy scoring uses the same memory and within-frame prefix:

$$\log \pi_\theta(\hat{\mathbf{z}}_{1:H}^{(i)} \mid \mathbf{c}_v, \mathbf{a}_{0:H-1}, \ell) = \sum_{t=1}^{H} \sum_{j=1}^{N} \log \pi_\theta(\hat{z}_{t,j}^{(i)} \mid M_t^{(i)}, \hat{\mathbf{z}}_{t,<j}^{(i)}, \ell). \tag{6}$$

The old, current, and reference policies share these conditions, preserving the link between each candidate and its history during optimization. Supervised training uses ground-truth histories; RL and free rollouts use predicted histories, with true future frames reserved for reward computation.

3.4 Memory-Conditioned Search-Guided Policy Optimization

`MemSPO` turns alternative futures into post-training feedback: it constructs candidates through diverse search, maintains their histories as described above, and learns from their relative trajectory quality. The update uses GRPO-style group-relative advantages and a clipped policy surrogate (Shao et al., 2024; Schulman et al., 2017).

Constructing alternative futures. We maintain G search groups with b beams each. For prefix y and next token v, group g uses the diverse search score (Vijayakumar et al., 2018):

$$S_g(y \oplus v) = S_g(y) + \log \pi_{\text{old}}(v \mid u(y)) - \lambda_{\text{div}} d_g(v). \tag{7}$$

Here, $u(y)$ contains the branch's memory and multimodal conditions, and $d_g(v)$ measures the normalized frequency of v selected by earlier groups at the same position. Each group retains its b highest-scoring branches and returns one final trajectory, producing G videos under the same input conditions. The penalty encourages token-level differences across groups and affects selection only; policy scoring uses unmodified probabilities.

Comparing trajectory quality. The frozen visual decoder converts candidate tokens into videos. Following RLVR-World (Wu et al., 2025a), our base reward combines pixel error and perceptual distance (Zhang et al., 2018):

$$R_i = -\frac{1}{H} \sum_{t=1}^{H} \left[\text{MAE}(\hat{o}_t^{(i)}, o_t) + \text{LPIPS}(\hat{o}_t^{(i)}, o_t) \right]. \tag{8}$$

For either reward design (Appendix B.3), we normalize scores within each input:

$$A_i = \frac{R_i - \bar{R}}{\text{Std}(R_1, \ldots, R_G) + \varepsilon}, \qquad \bar{R} = \frac{1}{G} \sum_{i=1}^{G} R_i. \tag{9}$$

All dynamic tokens in a trajectory share A_i. True future frames enter reward computation only, never candidate memory. Video decoding and reward computation carry no gradients.

Learning under matching histories. The current, old, and frozen reference policies share each candidate's generation condition $u_{t,j}^{(i)} = (M_t^{(i)}, \hat{\mathbf{z}}_{t,<j}^{(i)}, \ell)$. The token probability ratio is

$$\rho_{t,j}^{(i)}(\theta) = \frac{\pi_\theta(\hat{z}_{t,j}^{(i)} \mid u_{t,j}^{(i)})}{\pi_{\text{old}}(\hat{z}_{t,j}^{(i)} \mid u_{t,j}^{(i)})}. \tag{10}$$

`MemSPO` maximizes the surrogate objective

$$\mathcal{J}_{\text{MemSPO}}(\theta) = \frac{1}{GHN} \sum_{i=1}^{G} \sum_{t=1}^{H} \sum_{j=1}^{N} \left[\ell_{\text{clip}}\left(\rho_{t,j}^{(i)}(\theta), A_i\right) - \beta \kappa_{t,j}^{(i)}(\theta) \right]. \tag{11}$$

Here, ℓ_{clip} is the clipped policy term, including dual clipping for negative advantages. The reference regularizer is $\kappa = q - \log q - 1$, where $q = \pi_{\text{ref}}/\pi_\theta$ evaluates the same token under the same history.

Each iteration constructs fresh candidates and performs one world-model update, transferring their relative-quality feedback into the model parameters. The visual tokenizer, text encoder, and memory rules remain fixed. Beam selection changes the candidate distribution, so this is a search-guided surrogate without an unbiased on-policy gradient guarantee. We analyze how candidate construction shapes the group-relative reward gradient and bound its magnitude under candidate collapse (Appendix A, Proposition A.1 and Corollary A.1).

4 Experiments

Our experiments address four research questions: ❶ ***RQ1*** How well does FutureWorlds predict future observations? ❷ ***RQ2*** Can MemSPO improve generation with few post-training updates? ❸ ***RQ3*** Do generation gains translate into more accurate motion and dynamics? ❹ ***RQ4*** Does memory maintenance improve prediction beyond the training horizon?

4.1 Experimental Setup

Baselines. We compare four baseline categories: ✦ video generation models: FitVid (Babaeizadeh et al., 2021) and iVideoGPT (Wu et al., 2024); ✦ action-conditioned world models: IRASim (Zhu et al., 2025), Vid2World (Huang et al., 2026), Ctrl-World (Guo et al., 2026), DriftWorld (Lu et al., 2026), and WEAVER (Jain et al., 2026); ✦ RL post-training methods: RLVR-World (Wu et al., 2025a) and PersistWorld (Bardhan et al., 2026); and ✦ adapted video foundation models: Wan2.2-TI2V-5B (Team Wan et al., 2025), Cosmos-Predict2.5-2B (NVIDIA, 2025), and DreamDojo-2B (Gao et al., 2026). Adaptation and evaluation details appear in Appendix B.4.

Datasets and Evaluation. We evaluate action-conditioned video generation on RT-1 (Brohan et al., 2023), BridgeV2 (Walke et al., 2023), and RoboCasa (Nasiriany et al., 2026), covering real and simulated manipulation. Table 1 compares 32-frame predictions on 128 fixed held-out trajectories per dataset using PSNR, SSIM, and LPIPS (Zhang et al., 2018). Methods share trajectories and target frames, with predictions resized to the reference resolution. Metric definitions and evaluation details appear in Appendix B.4.

Implementation Details. We train `FutureWorlds` in PyTorch on 16 NVIDIA H20 GPUs across two servers. Starting from SFT, `MemSPO` uses AdamW with learning rate 10^{-6} and freezes the visual tokenizer and text encoder. Diverse beam search uses four groups of two beams, retaining one trajectory per group for reward comparison and updates. Main evaluations use a beam width of four and persistent memory. Pretrained initialization, SFT, and RL settings appear in Appendix B.1.

4.2 Video Generation Quality (RQ1)

In Table 1, `FutureWorlds` leads all three metrics on all datasets. Relative to the strongest baseline per metric on BridgeV2, it gains 1.71 dB PSNR and 2.17% SSIM and reduces LPIPS by 20.84%. LPIPS also decreases by 14.78% on RT-1 and 9.12% on RoboCasa relative to the strongest baseline per dataset. Figure 3 shows closer object locations and arm poses in RT-1, fewer object substitutions and local distortions in BridgeV2, and more accurate arm and gripper poses in RoboCasa.

Table 1: Quantitative comparison of 32-frame action-conditioned video prediction (128 trajectories per dataset). Metrics: PSNR↑, SSIM↑, and LPIPS↓. 1st, 2nd, and 3rd mark the top three scores; the last row reports relative gains over the runner-up.

Method	RT-1			BridgeV2			RoboCasa		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Video Generation Models									
FitVid	18.71	0.5916	0.4750	17.09	0.6072	0.5235	13.50	0.5422	0.5655
iVideoGPT	15.96	0.5571	0.3833	20.06	0.7791	0.1899	16.67	0.7187	0.2366
Action-Conditioned World Models									
IRASim	16.83	0.5936	0.3474	17.39	0.7094	0.2524	13.55	0.5834	0.3706
Vid2World	19.03	0.7210	0.1960	18.92	0.7110	0.1880	15.40	0.7000	0.2287
Ctrl-World	19.11	0.7110	0.2522	18.38	0.6932	0.2621	15.55	0.7035	0.2686
DriftWorld	20.54	0.7410	0.2595	20.32	0.7429	0.2379	16.50	0.7200	0.2891
WEAVER	19.74	0.7617	0.2321	18.74	0.7447	0.2476	14.62	0.7136	0.2726
RL Post-Training									
RLVR-World	19.47	0.7360	0.1820	19.06	0.7210	0.1780	15.80	0.7100	0.2017
PersistWorld	18.77	0.6995	0.2717	19.07	0.7154	0.2408	15.61	0.7112	0.2639
Adapted Video Foundation Models									
Wan2.2-TI2V-5B	18.42	0.7058	0.2478	16.35	0.6889	0.2568	14.37	0.6906	0.2711
Cosmos-Predict2.5-2B	17.77	0.6765	0.2889	18.98	0.7799	0.1870	14.30	0.6802	0.2916
DreamDojo-2B	18.47	0.7152	0.2660	16.95	0.7461	0.2563	13.92	0.6349	0.3278
FutureWorlds (ours)	**22.47**	**0.8026**	**0.1551**	**22.03**	**0.7968**	**0.1409**	**18.47**	**0.7659**	**0.1833**
Gain over runner-up	+9.40%	+5.37%	+14.78%	+8.42%	+2.17%	+20.84%	+10.80%	+6.38%	+9.12%

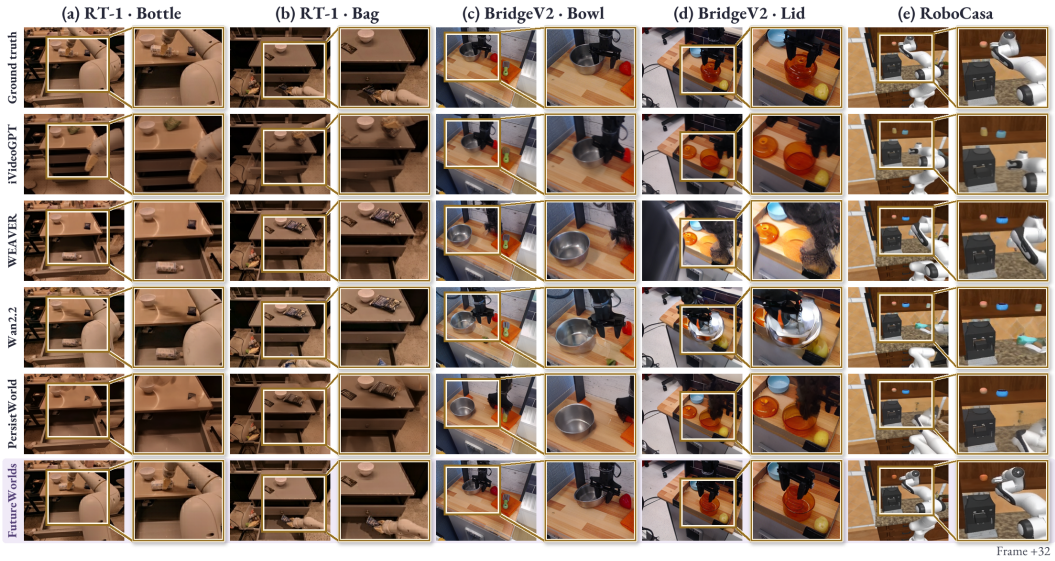

Figure 3: ***Qualitative comparison at frame 32.*** Five selected cases on RT-1, BridgeV2, and RoboCasa comparing four baselines and our method, with full images and matched detail views.

These comparisons support improved object consistency and action-conditioned prediction. The matched crops expose incorrect object positions relative to the gripper despite plausible textures, distinguishing faithful action-conditioned prediction from visually plausible but inconsistent manipulation states. Further quantitative and qualitative results appear in the appendix. A qualitative case in Appendix D illustrates how predicted futures provide feedback for action selection toward closed-loop control.

Table 2: Post-training comparison with 200 updates. Metrics: PSNR↑, SSIM↑, and LPIPS↓. Gain rows show relative improvements over GRPO (%).

Method	10 frames			20 frames			32 frames		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
RT-1									
SFT (50k)	23.76	0.8298	0.1308	22.49	0.8075	0.1527	21.37	0.7860	0.1734
GRPO (200)	23.97	0.8319	0.1289	22.60	0.8089	0.1517	21.39	0.7858	0.1739
MemSPO (200)	**24.33**	**0.8357**	**0.1240**	**23.42**	**0.8192**	**0.1397**	**22.45**	**0.8016**	**0.1565**
Gain over GRPO	+1.49%	+0.45%	+3.80%	+3.61%	+1.28%	+7.92%	+4.97%	+2.01%	+9.98%
BridgeV2									
SFT (50k)	24.07	0.8300	0.1096	22.74	0.8084	0.1270	21.73	0.7940	0.1450
GRPO (200)	24.12	0.8301	0.1089	22.80	0.8088	0.1263	21.80	0.7945	0.1441
MemSPO (200)	**24.31**	**0.8320**	**0.1073**	**23.00**	**0.8108**	**0.1244**	**21.97**	**0.7962**	**0.1418**
Gain over GRPO	+0.83%	+0.23%	+1.44%	+0.88%	+0.25%	+1.53%	+0.78%	+0.22%	+1.58%
RoboCasa									
SFT (50k)	19.50	0.7957	0.1513	18.22	0.7689	0.1787	17.50	0.7519	0.1971
GRPO (200)	19.68	0.7965	0.1514	18.64	0.7729	0.1758	18.03	0.7581	0.1919
MemSPO (200)	**19.92**	**0.7984**	**0.1476**	**18.88**	**0.7743**	**0.1718**	**18.29**	**0.7600**	**0.1877**
Gain over GRPO	+1.21%	+0.24%	+2.47%	+1.31%	+0.19%	+2.28%	+1.45%	+0.25%	+2.17%

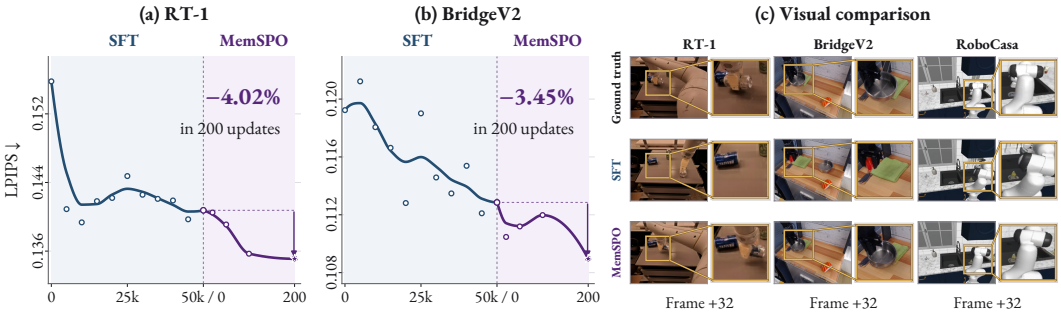

Figure 4: ***Post-training gains and qualitative improvements.*** (a,b) LPIPS during SFT and MemSPO training. (c) Frame-32 predictions with matched detail views on three datasets.

4.3 Post-Training Effectiveness and Efficiency (RQ2)

On 128 matched trajectories per dataset with decoding and memory fixed, MemSPO leads all three metrics at 10, 20, and 32 frames after 200 RL updates (Table 2). At 32 frames, LPIPS drops by 9.98%, 1.58%, and 2.17% versus GRPO on RT-1, BridgeV2, and RoboCasa, showing gains beyond ten-frame training.

On 32 monitoring clips, 200 updates reduce LPIPS by 4.02%/3.45% on RT-1/BridgeV2 versus SFT (Figure 4a,b), with improved object consistency and motion (c). Gains persist across five decoding settings (Appendix Table 3).

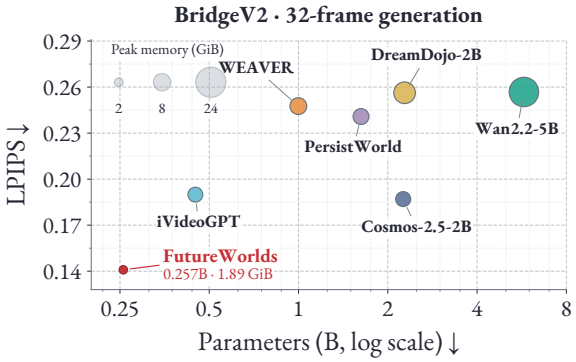

Figure 5: ***Model size and quality.*** Bubble area denotes peak GPU memory on one H20.

With checkpoints and beam width fixed, the LPIPS reduction remains 3.03–4.32% on RT-1 and 2.60–3.84% on BridgeV2 across the tested temperature and top-p settings. A matched ordinary-beam post-training comparison also favors MemSPO in PSNR and LPIPS across all three datasets and horizons under identical evaluation decoding and memory settings (Appendix E.3, Table 5).

(a) Motion prediction quality

32 frames · 128 trajectories per dataset

Method	RT-1			BridgeV2			RoboCasa		
	Flow ↓	Gap ↓	DINO ↑	Flow ↓	Gap ↓	DINO ↑	Flow ↓	Gap ↓	DINO ↑
SFT	1.29	0.74	0.94	0.90	0.33	0.94	1.28	0.81	0.92
WEAVER	1.77	1.19	0.87	1.57	0.59	0.85	1.89	1.27	0.88
Cosmos-Predict2.5-2B	2.03	0.89	0.81	1.72	0.53	0.88	1.99	1.04	0.77
DreamDojo-2B	2.34	0.78	0.85	1.92	0.59	0.83	2.94	1.48	0.85
FutureWorlds (ours)	**1.13**	**0.43**	**0.96**	**0.86**	**0.30**	**0.94**	**1.07**	**0.53**	**0.92**

(b) Optical flow

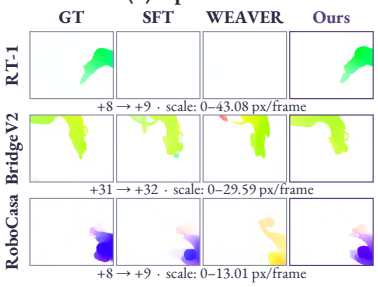

Figure 6: ***Motion quality and optical-flow comparisons.*** (a) Full-cohort results at 32 frames. (b) Selected examples with a shared flow-color scale within each dataset.

Resource footprint. `FutureWorlds` leads seven models in BridgeV2 LPIPS with 0.257B inference parameters (visual codec included) and 1.89 GiB peak memory (Figure 5). Costs use three 32-frame rollouts at batch size one, with native configurations and cached language features. Beam decoding takes 26.76 s versus iVideoGPT's 4.14 s per rollout. Smaller models do not ensure faster inference; Appendix G details all measurements and settings.

4.4 Motion and Dynamic Prediction (RQ3)

Using WorldArena's flow and feature components (Shang et al., 2026), we evaluate 128 trajectories per dataset against SFT, WEAVER, Cosmos, and DreamDojo (Figure 6a). Flow error measures vector disagreement, magnitude gap measures motion-strength mismatch, and DINO similarity measures frame-level feature agreement (Appendix B.7). `FutureWorlds` leads all three diagnostics, reducing flow error by 12.40%, 4.57%, and 16.68% versus SFT on RT-1, BridgeV2, and RoboCasa. Improvements in both flow error and magnitude gap support closer motion agreement rather than merely larger movements. The shared-scale examples (b) complement these cohort-level results.

4.5 Memory Maintenance (RQ4)

We fix weights and decoding and vary only the retained history: the latest state, six recent states, or an initial anchor plus six recent states. Full memory leads PSNR, SSIM, and MSE at 32 frames across all datasets (Figure 7), reducing MSE by 40.83%, 57.69%, and 29.22% versus removing historical memory on RT-1, BridgeV2, and RoboCasa. Recent history supplies most gains; the anchor adds smaller benefits. As no variant is retrained, this comparison isolates the contribution of retained context at inference. Appendix F reports settings and results across horizons.

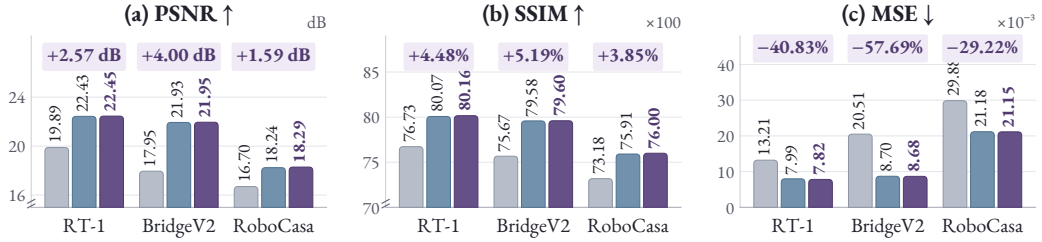

Figure 7: ***Memory ablation at 32 frames.*** w/o Historical Memory, w/o Initial Anchor, and Full Model. Gains are versus w/o Historical Memory; PSNR and SSIM axes are truncated.

5 Conclusion and Future Work

We present `FutureWorlds`, which learns from alternative futures through coordinated candidate construction, history maintenance, and quality comparison. `MemSPO` combines diverse search with history-consistent optimization, improving generation within hundreds of updates and beyond the training horizon. Three-dataset evaluations support gains in visual quality, object consistency, and motion prediction. BridgeV2 comparisons also show better quality with fewer parameters and lower measured memory usage, although autoregressive beam search incurs latency. Future work focuses on faster decoding and validating these benefits in closed-loop planning and robot control.

AI USE STATEMENT

We use generative AI tools to assist with manuscript drafting, translation, and language editing; implementation and debugging of research and evaluation code; refinement of experimental protocols; qualitative case inspection and interpretation of results; review of mathematical derivations; and preparation of figures and tables. Reported numerical results come from experimental outputs rather than language-model-generated estimates. The authors take full responsibility for verifying AI-assisted work and for the accuracy and integrity of the final text, references, claims, code, and other artifacts.

REPRODUCIBILITY STATEMENT

Section 3 presents the model and optimization procedure. Appendix B documents initialization, supervised fine-tuning, post-training, reward designs, baseline adaptation, and evaluation protocols. Appendices E.3, F, and G provide the settings and full results for candidate-construction, memory, and resource comparisons. The anonymous project link in the abstract provides the code and reproduction instructions; we plan to release the trained checkpoints.

REFERENCES

Thomas Anthony, Zheng Tian, and David Barber. Thinking Fast and Slow with Deep Learning and Tree Search. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1705.08439.

Mohammad Babaeizadeh, Mohammad Taghi Saffar, Suraj Nair, Sergey Levine, Chelsea Finn, and Dumitru Erhan. FitVid: Overfitting in Pixel-Level Video Prediction. *arXiv preprint arXiv:2106.13195*, 2021. URL https://arxiv.org/abs/2106.13195.

Jai Bardhan, Patrik Drozdík, Josef Šivic, and Vladimír Petrík. PersistWorld: Stabilizing Multi-step Robot World Model Rollouts via Reinforcement Learning. *arXiv preprint*, 2026. URL https://jaibardhan.com/persistworld/.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In *Advances in Neural Information Processing Systems*, 2015. URL https://arxiv.org/abs/1506.03099.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training Diffusion Models with Reinforcement Learning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.13301.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Robotics: Science and Systems*, 2023. URL https://roboticsproceedings.org/rss19/p025.html.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, 2019. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

Perry Dong, Yueru Jia, Chelsea Finn, and Dorsa Sadigh. Q-Learning With World Models. *arXiv preprint arXiv:2608.17163*, 2026. URL https://arxiv.org/abs/2608.17163.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual Foresight: Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control. *arXiv preprint arXiv:1812.00568*, 2018. URL https://arxiv.org/abs/1812.00568.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised Learning for Physical Interaction through Video Prediction. In *Advances in Neural Information Processing Systems*, 2016. URL https://arxiv.org/abs/1605.07157.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, et al. DreamDojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026. URL https://arxiv.org/abs/2602.06949.

GigaWorld Team, Angyuan Ma, Boyuan Wang, Bohan Li, Chaojun Ni, Guo Li, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jingyu Liu, Jiwen Lu, Qiuping Deng, Tingdong Yu, Xuancheng Xu, Xinyu Zhou, Xiuwei Xu, Xinze Chen, Xiaofeng Wang, Xiaoyu Tian, Yang Wang, Yifan Chang, Yukun Zhou, Yun Ye, Zhenyu Wu, Zhanqian Wu, and Zheng Zhu. GigaWorld-1: A Roadmap to Build World Models for Robot Policy Evaluation. *arXiv preprint arXiv:2607.02642*, 2026. URL https://arxiv.org/abs/2607.02642.

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-World: A Controllable Generative World Model for Robot Manipulation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.10125.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *International Conference on Machine Learning*, pp. 2555–2565, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1912.01603.

Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. StreamingT2V: Consistent, Dynamic, and Extendable Long Video Generation from Text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2568–2577, 2025. URL https://arxiv.org/abs/2403.14773.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1904.09751.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2World: Crafting Video Diffusion Models to Interactive World Models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2505.14357.

Arnav Kumar Jain, Yilin Wu, Jesse Farebrother, Gokul Swamy, and Andrea Bajcsy. WEAVER, Better, Faster, Longer: An Effective World Model for Robotic Manipulation. *arXiv preprint arXiv:2606.13672*, 2026. URL https://arxiv.org/abs/2606.13672.

Jaeha Lee, Gio Huh, Ning Su, and Tony Yue Yu. Discovering Hidden Algebraic Structures via Transformers with Rank-Aware Beam GRPO. *arXiv preprint arXiv:2508.15766*, 2025. URL https://arxiv.org/abs/2508.15766.

Susie Lu, Haonan Chen, Weirui Ye, and Yilun Du. DriftWorld: Fast World Modeling through Drifting. *arXiv preprint arXiv:2607.15065*, 2026. URL https://arxiv.org/abs/2607.15065.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.15505.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A Large-Scale Simulation Framework for Training and Benchmarking Generalist Robots. In *International Conference on Learning Representations*, 2026. URL https://robocasa.ai/assets/robocasa365_iclr26.pdf.

NVIDIA. World Simulation with Video Foundation Models for Physical AI. *arXiv preprint arXiv:2511.00062*, 2025. URL https://arxiv.org/abs/2511.00062.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL https://jmlr.org/papers/v21/20-074.html.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence Level Training with Recurrent Neural Networks. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.06732.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-Critical Sequence Training for Image Captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. URL https://arxiv.org/abs/1612.00563.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Yu Shang, Zhuohang Li, Yiding Ma, Weikang Su, Xin Jin, Ziyou Wang, Lei Jin, Xin Zhang, Yinzhou Tang, Haisheng Su, et al. Worldarena: A unified benchmark for evaluating perception and functional utility of embodied world models. *arXiv preprint arXiv:2602.08971*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1711.00937.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse Beam Search for Improved Description of Complex Scenes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. URL https://ojs.aaai.org/index.php/AAAI/article/view/12340.

Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, Abraham Lee, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A Dataset for Robot Learning at Scale. In *Conference on Robot Learning*, pp. 1723–1736, 2023. URL https://proceedings.mlr.press/v229/walke23a.html.

Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. iVideoGPT: Interactive VideoGPTs are Scalable World Models. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.15223.

Jialong Wu, Shaofeng Yin, Ningya Feng, and Mingsheng Long. RLVR-World: Training World Models with Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 2025a. URL https://arxiv.org/abs/2505.13934.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video World Models with Long-term Spatial Memory. *arXiv preprint arXiv:2506.05284*, 2025b. URL https://arxiv.org/abs/2506.05284.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17453.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: Video Generation using VQ-VAE and Transformers. *arXiv preprint arXiv:2104.10157*, 2021. URL https://arxiv.org/abs/2104.10157.

Sherry Yang, Yilun Du, Seyed Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning Interactive Real-World Simulators. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.06114.

Yang Ye, Tianyu He, Shuo Yang, and Jiang Bian. Reinforcement Learning with Inverse Rewards for World Model Post-training. *arXiv preprint arXiv:2509.23958*, 2025. URL https://arxiv.org/abs/2509.23958.

Tenny Yin, Zhiting Mei, Zhonghe Zheng, Miyu Yamane, David Wang, Jade Sceats, Samuel M. Bateman, Lihan Zha, Apurva Badithela, Ola Shorinwa, and Anirudha Majumdar. PlayWorld: Learning Robot World Models from Autonomous Play. *arXiv preprint arXiv:2603.09030*, 2026. URL https://arxiv.org/abs/2603.09030.

Lvmin Zhang and Maneesh Agrawala. Packing Input Frame Context in Next-Frame Prediction Models for Video Generation. *arXiv preprint arXiv:2504.12626*, 2025. URL https://arxiv.org/abs/2504.12626.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. URL https://arxiv.org/abs/1801.03924.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2302.04867.

Zirui Zheng, Jiaqian Yu, Xiongfeng Peng, Jun Shi, Mingyi Li, Chao Zhang, Weiming Li, Dong Wang, Huchuan Lu, and Xu Jia. Mem-World: Memory-Augmented Action-Conditioned World Models for Persistent Robot Manipulation. *arXiv preprint arXiv:2606.18960*, 2026. URL https://arxiv.org/abs/2606.18960.

Fangqi Zhu et al. IRASim: A Fine-Grained World Model for Robot Manipulation. *arXiv preprint arXiv:2406.14540*, 2025. URL https://arxiv.org/abs/2406.14540.

A Theoretical Analysis

We analyze how alternative futures supply a learning signal in `FutureWorlds`. On a fixed candidate set, the local reward gradient of `MemSPO` admits a pairwise decomposition that links relative rewards to differences in policy score gradients. The analysis characterizes this update; it does not establish that search produces better candidates.

A.1 Pairwise Structure of the Reward Gradient

For a fixed input, consider $G \geq 2$ candidates with $L = HN$ tokens each. Flatten the token indices into $k \in \{1, \ldots, L\}$, with $z_k^{(i)}$ and $u_k^{(i)}$ denoting candidate i's token and condition, including its memory and within-frame prefix. Tokens, conditions, rewards, and normalization statistics remain fixed during differentiation. Selected tokens have positive old-policy probabilities, and current-policy probabilities are differentiable near θ_{old}.

Let $\sigma_R = \text{Std}(R_1, \ldots, R_G)$ use the same convention as Eq. (9). Define the group-relative advantage and mean token score gradient as

$$A_i = \frac{R_i - \bar{R}}{\sigma_R + \varepsilon}, \qquad \bar{R} = \frac{1}{G}\sum_{i=1}^{G} R_i, \qquad \mathbf{s}_i = \frac{1}{L}\sum_{k=1}^{L} \nabla_\theta \log \pi_\theta(z_k^{(i)} \mid u_k^{(i)})\Big|_{\theta=\theta_{\text{old}}}. \tag{A.1}$$

Here, $\varepsilon > 0$ stabilizes reward normalization. The implementation uses sample standard deviation, so $\sigma_R^2 = \frac{1}{G-1}\sum_i (R_i - \bar{R})^2$. We first analyze the normalized advantages above; the numerical zero-variance guard is addressed below. The reward component of Eq. (11) is

$$\mathcal{J}_R(\theta) = \frac{1}{GL}\sum_{i=1}^{G}\sum_{k=1}^{L} \ell_{\text{clip}}\left(\rho_k^{(i)}(\theta), A_i\right), \qquad \rho_k^{(i)}(\theta) = \frac{\pi_\theta(z_k^{(i)} \mid u_k^{(i)})}{\pi_{\text{old}}(z_k^{(i)} \mid u_k^{(i)})}. \tag{A.2}$$

Proposition A.1 (Pairwise reward-gradient identity)**.** *Suppose generation and policy scoring use matching candidate histories, the clipping radius satisfies $0 < \epsilon_c < 1$, and the negative-advantage dual-clipping constant satisfies $c_d > 1$. At $\theta = \theta_{\text{old}}$, the reward gradient admits the pairwise representation*

$$\mathbf{g}_R := \nabla_\theta \mathcal{J}_R(\theta)|_{\theta=\theta_{\text{old}}} = \frac{1}{G}\sum_{i=1}^{G} A_i \mathbf{s}_i = \frac{\sum_{i<j}(R_i - R_j)(\mathbf{s}_i - \mathbf{s}_j)}{G^2(\sigma_R + \varepsilon)}. \tag{A.3}$$

Proof. We first reduce the clipped objective locally. Write

$$v(\rho, A) = \min\{\rho A, \text{clip}(\rho, 1 - \epsilon_c, 1 + \epsilon_c)A\}. \tag{A.4}$$

The policy term is $v(\rho, A)$ for $A \geq 0$ and $\max\{v(\rho, A), c_d A\}$ for $A < 0$. At identical parameters, evaluating numerator and denominator under the same fixed condition gives $\rho_k^{(i)}(\theta_{\text{old}}) = 1$. Matching this condition to generation additionally ensures that the score refers to the history that produces the candidate. In a neighborhood of this point, ordinary clipping is inactive. For $A_i < 0$, the strict inequality $A_i > c_d A_i$ also makes dual clipping inactive; for $A_i = 0$, the term and its derivative are zero. Consequently,

$$\nabla_\theta \ell_{\text{clip}}(\rho_k^{(i)}(\theta), A_i)\Big|_{\theta=\theta_{\text{old}}} = A_i \, \nabla_\theta \rho_k^{(i)}(\theta)\Big|_{\theta=\theta_{\text{old}}}. \tag{A.5}$$

The denominator of the probability ratio is fixed, so its derivative satisfies

$$\nabla_\theta \rho_k^{(i)}(\theta) = \frac{\nabla_\theta \pi_\theta(z_k^{(i)} \mid u_k^{(i)})}{\pi_{\text{old}}(z_k^{(i)} \mid u_k^{(i)})} = \rho_k^{(i)}(\theta) \nabla_\theta \log \pi_\theta(z_k^{(i)} \mid u_k^{(i)}). \tag{A.6}$$

Substituting into Eq. (A.2) and evaluating at the old policy yields

$$\mathbf{g}_R = \frac{1}{G}\sum_{i=1}^{G} A_i \mathbf{s}_i = \frac{1}{G(\sigma_R + \varepsilon)}\sum_{i=1}^{G}(R_i - \bar{R})\mathbf{s}_i. \tag{A.7}$$

To obtain the pairwise form, symmetry and the vanishing diagonal terms give

$$\sum_{i<j}(R_i - R_j)(\mathbf{s}_i - \mathbf{s}_j) = \frac{1}{2}\sum_{i=1}^{G}\sum_{j=1}^{G}\left(R_i\mathbf{s}_i + R_j\mathbf{s}_j - R_i\mathbf{s}_j - R_j\mathbf{s}_i\right). \tag{A.8}$$

Collecting the four sums and using $\sum_i R_i = G\bar{R}$ gives

$$\sum_{i<j}(R_i - R_j)(\mathbf{s}_i - \mathbf{s}_j) = G\sum_{i=1}^{G} R_i\mathbf{s}_i - \left(\sum_{i=1}^{G} R_i\right)\left(\sum_{i=1}^{G}\mathbf{s}_i\right) = G\sum_{i=1}^{G}(R_i - \bar{R})\mathbf{s}_i. \tag{A.9}$$

Combining Eqs. (A.7) and (A.9) proves Eq. (A.3). □

Each pair contributes through its reward and score-gradient differences; either vanishing difference removes its contribution. No independence assumption is required, so the identity also holds for beam-search candidates. The algebra applies to any fixed conditions shared by numerator and denominator; history matching gives these scores their intended candidate-generation interpretation, rather than providing an independent performance guarantee.

A.2 A Bound under Candidate Collapse

Define the empirical reward variance and score-gradient dispersion by

$$\bar{\mathbf{s}} = \frac{1}{G}\sum_{i=1}^{G}\mathbf{s}_i, \qquad V_R = \frac{1}{G}\sum_{i=1}^{G}(R_i - \bar{R})^2, \qquad V_s = \frac{1}{G}\sum_{i=1}^{G}\|\mathbf{s}_i - \bar{\mathbf{s}}\|_2^2. \tag{A.10}$$

We distinguish V_R from σ_R^2 so that the result accommodates either population or sample standard deviation in advantage normalization.

Corollary A.1 (Reward-gradient magnitude)**.** *Under the conditions of Proposition A.1,*

$$\|\mathbf{g}_R\|_2 \le \frac{\sqrt{V_R V_s}}{\sigma_R + \varepsilon}. \tag{A.11}$$

In particular, the reward gradient is zero if all candidate rewards are equal or all candidate score gradients coincide.

Proof. Since $\sum_i(R_i - \bar{R}) = 0$, subtracting the mean score gradient leaves Eq. (A.7) unchanged:

$$\mathbf{g}_R = \frac{1}{G(\sigma_R + \varepsilon)}\sum_{i=1}^{G}(R_i - \bar{R})(\mathbf{s}_i - \bar{\mathbf{s}}). \tag{A.12}$$

The triangle inequality gives

$$\|\mathbf{g}_R\|_2 \le \frac{1}{G(\sigma_R + \varepsilon)}\sum_{i=1}^{G}|R_i - \bar{R}|\,\|\mathbf{s}_i - \bar{\mathbf{s}}\|_2. \tag{A.13}$$

Applying the Cauchy–Schwarz inequality to the two scalar sequences yields

$$\|\mathbf{g}_R\|_2 \le \frac{\sqrt{\sum_i(R_i - \bar{R})^2}\sqrt{\sum_i\|\mathbf{s}_i - \bar{\mathbf{s}}\|_2^2}}{G(\sigma_R + \varepsilon)} = \frac{\sqrt{V_R V_s}}{\sigma_R + \varepsilon}. \tag{A.14}$$

If either V_R or V_s vanishes, the upper bound is zero. □

We can also express dispersion through pairwise distances. Expanding the squared norms gives

$$\sum_{i<j}\|\mathbf{s}_i - \mathbf{s}_j\|_2^2 = G\sum_{i=1}^{G}\|\mathbf{s}_i\|_2^2 - \left\|\sum_{i=1}^{G}\mathbf{s}_i\right\|_2^2 = G\sum_{i=1}^{G}\|\mathbf{s}_i - \bar{\mathbf{s}}\|_2^2. \tag{A.15}$$

Hence $V_s = G^{-2} \sum_{i<j} \|\mathbf{s}_i - \mathbf{s}_j\|_2^2$. If every pair satisfies $\|\mathbf{s}_i - \mathbf{s}_j\|_2 \le \delta$, the $G(G-1)/2$ pairs imply

$$V_s \le \frac{G-1}{2G}\delta^2, \qquad \|\mathbf{g}_R\|_2 \le \frac{\sqrt{V_R}}{\sigma_R + \varepsilon}\sqrt{\frac{G-1}{2G}}\,\delta. \tag{A.16}$$

Thus, at a fixed reward scale, collapse in candidate score gradients limits the reward-driven update. Larger dispersion alone does not guarantee a larger or better-directed update, since pairwise contributions can cancel. Visual diversity is also distinct from score-gradient dispersion; connecting the two requires empirical analysis.

Scope. These statements concern the reward component at θ_{old}; the reference-regularization gradient can remain nonzero under collapse, and clipping can become active away from this point. Because search changes the candidate distribution, the analysis implies neither an unbiased on-policy gradient nor monotonic improvement. The benefit of diverse beam search remains an empirical question. In our implementation, each fresh candidate set receives one update evaluated at the old parameters, where clipping is locally inactive. If $\sigma_R \le 10^{-6}$, the implementation sets all advantages to zero; the reward gradient is then zero and the magnitude bound remains valid, although the unguarded pairwise equality need not hold for small nonzero reward differences. The full optimizer step also includes reference regularization and optimizer preconditioning, so it is distinct from $\mathbf{g}_R$.

B Implementation Details

B.1 Pretrained Initialization, SFT, and RL

Pretrained initialization and tokenizer adaptation. We initialize from released pretrained visual-tokenizer and autoregressive world-model checkpoints. RT-1 reuses the pretrained tokenizer directly; BridgeV2 and RoboCasa use dataset-adapted tokenizers. Adaptation optimizes the reconstruction objective in Eq. 3, with unit weights on $\mathcal{L}_1$ and VGG-LPIPS and an adversarial weight of 0.1. We use a global batch size of 32, a learning rate of 5×10^{-5}, and 5,000 warmup steps. The BridgeV2 and RoboCasa tokenizer checkpoints use 60,000 and 50,000 shared generator/discriminator steps, respectively; these counters do not denote generator-only updates. The resulting tokenizers remain frozen during world-model SFT and RL.

Supervised fine-tuning. Each dataset receives 50,000 SFT updates with global batch size 64. Teacher forcing supplies recorded histories, and cross-entropy supervises target dynamic tokens only. AdamW uses a backbone learning rate of 10^{-5}, a text-adapter learning rate of 10^{-4} where applicable, zero weight decay, and 500 warmup updates. RT-1 and RoboCasa use cosine decay to one tenth of the peak learning rate; BridgeV2 keeps the learning rate constant after warmup. Gradient norms are clipped at 1.0. We update the autoregressive backbone, action embeddings, and text adapters while freezing the visual tokenizer and T5 encoder. RT-1 and RoboCasa use language conditioning; BridgeV2 omits it. The SFT checkpoint initializes both the trainable RL policy and its frozen reference.

Reinforcement-learning post-training. MemSPO uses ten-frame predicted rollouts, global batch size 16, and fresh AdamW state with learning rate 10^{-6} and zero weight decay. Four search groups contain two beams each, with diversity penalty 0.1; one trajectory per group enters reward comparison. The clipping threshold is 0.2, the reference-regularization coefficient is 0.001, and the gradient-norm limit is 1.0. Each fresh candidate set receives one optimizer update. Candidate generation and policy scoring retain the same scene/state anchors and six recent states; future ground-truth frames enter reward computation only. Main-table checkpoints use 500/200/400 updates for RT-1/BridgeV2/RoboCasa, whereas the matched GRPO–MemSPO comparison uses 200 updates on every dataset. Dataset-specific reward choices are identified in Appendix B.3; Algorithm 1 details the update.

B.2 Training Algorithm

Algorithm 1 summarizes one-input updates; distributed training averages the losses across inputs. The colors identify candidate construction, history maintenance, and relative-quality learning. The tokenizer, text encoder, and SFT reference policy remain frozen throughout post-training.

Algorithm 1 FutureWorlds: learning from alternative futures

Input: SFT parameters θ_0; training examples $x = (\mathbf{c}_v, \mathbf{a}, \ell, o_{1:H})$; horizon H; N tokens/frame; G groups, b branches/group; memory budget B.
Output: Updated world-model parameters θ.

Initialize $\theta \leftarrow \theta_0$ and freeze $\pi_{\text{ref}} \leftarrow \pi_{\theta_0}$.
For each post-training iteration **do**
  Draw x; set $\theta_{\text{old}} \leftarrow \text{stopgrad}(\theta)$.
  Initialize G groups of b branches with the same scene and state anchors.
  Set the first branch score in each group to 0 and the others to $-\infty$.
  Construct / Maintain for $t = 1, \ldots, H$ **do**
    Build each branch's prefix from its memory and actions; reset its KV cache.
    For token position $j = 1, \ldots, N$ **do**
      For group $g = 1, \ldots, G$ **do**
        Compute valid-visual-token log probabilities with π_{old}.
        Extend branches using $S_g + \log \pi_{\text{old}} - \lambda_{\text{div}} d_g$ (Eq. 7).
        Retain the top b extensions; reorder prefixes, caches, and ancestry.
      End for; **end for**
    Reorder prior trajectory records and memories by the selected ancestry.
    Append each branch's frame; retain anchors and its recent history within B.
  End for; return the highest-search-score trajectory from each group.
  Compare: decode the G candidates and compute R_i (Appendix B.3).
  Compute $A_i = (R_i - \bar{R})/(\text{Std}(R) + \varepsilon)$; if $\text{Std}(R) \leq 10^{-6}$, set $A_i = 0$.
  Learn: score candidate tokens under their saved generation-time histories.
  Use these same conditions for old, current, and frozen reference policies.
  Form $\mathcal{J}_{\text{MemSPO}}$ (Eq. 11); take one optimizer step on $-\mathcal{J}$.
End for; **return** θ.

Search penalties select candidates; unmodified policy probabilities score them.

Implementation correspondence. Search scores accumulate across the full horizon. Candidate selection also reorders previously saved frame prefixes and responses, preserving each final trajectory's ancestry. Reward computation and candidate construction are detached from differentiation; ground-truth futures enter rewards only. The reference penalty uses the numerically clipped form $\text{clip}(q - \log q - 1, -10, 10)$. This training search differs from evaluation-time Beam4, which selects one frame prediction before proceeding to the next frame.

B.3 Flexible Reward Design

MemSPO separates trajectory evaluation from parameter optimization. A reward module assigns a scalar score to each candidate future, and group normalization converts these scores into the advantages in Eq. 9. Replacing the reward therefore leaves candidate construction, history maintenance, and the update objective unchanged. We describe two implemented designs that emphasize complementary aspects of prediction quality.

Global reconstruction reward. The base design in Eq. 8 measures whole-frame pixel and perceptual agreement. For candidate i, define

$$L^{(i)}_{\text{global},t} = \text{MAE}(\hat{o}^{(i)}_t, o_t) + \text{LPIPS}_{\text{VGG}}(\hat{o}^{(i)}_t, o_t), \qquad R^{(i)}_{\text{base}} = -\frac{1}{H}\sum_{t=1}^{H} L^{(i)}_{\text{global},t}. \tag{B.1}$$

MAE uses RGB values in $[0, 1]$, while frozen VGG-LPIPS receives images mapped to $[-1, 1]$. This design evaluates overall appearance without requiring region annotations.

Motion-aware extension. To emphasize changing regions and temporal variation, the extended design adds local appearance and frame-difference errors:

$$R^{(i)}_{\text{motion}} = -\frac{1}{H}\sum_{t=1}^{H}\left[L^{(i)}_{\text{global},t} + \lambda_{\text{roi}} L^{(i)}_{\text{roi},t} + \lambda_{\Delta} L^{(i)}_{\Delta,t}\right], \tag{B.2}$$

$$L^{(i)}_{\Delta,t} = \text{MAE}\Big(\hat{o}^{(i)}_t - \hat{o}^{(i)}_{t-1},\ o_t - o_{t-1}\Big). \tag{B.3}$$

At the first predicted frame, both preceding images equal the last observed frame. The local term $L^{(i)}_{\text{roi},t}$ combines masked MAE with LPIPS on the corresponding bounding crop. All candidates share a mask computed from the channel-averaged absolute ground-truth frame difference. We threshold it above the larger of its 90th percentile and 0.02, dilate it with an 11×11 window, and use the full frame if the mask covers less than 1% of the image. Masked MAE is normalized by the number of selected RGB values; the LPIPS crop spans at least 64×64 pixels. This region emphasizes visible changes without semantic segmentation or optical-flow estimation.

Fixed calibration. We set $\lambda_{\text{roi}} = 0.5 s_{\text{global}}/s_{\text{roi}}$ and $\lambda_{\Delta} = 0.25 s_{\text{global}}/s_{\Delta}$. Each scale is the median temporally averaged component loss over candidates from 64 training-only calibration episodes using the frozen SFT checkpoint. Four independently sampled and four diverse-search candidates per input yield 512 calibration trajectories. The scales are $s_{\text{global}} = 0.153368$, $s_{\text{roi}} = 0.190531$, and $s_{\Delta} = 0.026607$; they remain fixed throughout post-training.

Experimental configurations. Main-table RT-1 and RoboCasa checkpoints use R_{base}, whereas BridgeV2 uses R_{motion}. The matched GRPO–MemSPO comparison and the ordinary-versus-diverse search training comparison use R_{base} throughout, holding the reward fixed within each comparison. Both designs use ten-frame training rollouts and the same group-relative update. Ground-truth futures enter reward computation only; they never condition candidate generation or populate candidate memory. These designs demonstrate a shared optimization interface for different quality objectives, rather than establishing that every reward choice yields equivalent performance.

B.4 Baseline Adaptation and Evaluation

Shared evaluation. Table 1 evaluates 32-frame predictions on 128 matched trajectories per dataset. RT-1 and BridgeV2 use held-out development sets; RoboCasa uses a held-out test set. Predictions are resized to the reference resolution, and LPIPS-VGG, SSIM, and frame-wise PSNR are averaged over frames and trajectories. Recorded actions condition generation; ground-truth future images are evaluation targets and never refresh rollout histories. Models retain their native input interfaces and decoding procedures. Cosmos-Predict2.5-2B uses one initial observation, whereas `FutureWorlds` uses two. Ten-frame baseline results use prefixes of the same 32-frame rollouts.

FutureWorlds checkpoints. Main-table evaluations use a beam width of four and persistent memory, with 500/200/400 MemSPO updates for RT-1/BridgeV2/RoboCasa. BridgeV2 uses the motion-aware reward (Appendix B.3) and omits text conditioning. The matched post-training comparison instead uses 200 updates for both RL methods on every dataset.

iVideoGPT and PersistWorld. On BridgeV2, iVideoGPT adapts the official OXE-256 tokenizer and autoregressive backbone for 20,000 updates each, freezing the tokenizer during backbone training. It uses two observations, recorded seven-dimensional actions, temperature 1, and top-k 100 sampling. Ten-frame chunks condition subsequent chunks on the two latest generated frames. PersistWorld starts from the released Ctrl-World checkpoint, performs 2,000 target-domain updates followed by 30 NFT updates, and uses generated latent history with 25 denoising steps and guidance scale 2.0. Its visual tokenizer and CLIP encoders remain frozen during base adaptation. Neither rollout accesses future ground-truth frames.

Wan2.2 adaptation. We adapt the official Wan2.2-TI2V-5B checkpoint (Team Wan et al., 2025) using action and history adapters plus rank-32 LoRA (Hu et al., 2021) in attention layers (Figure 8). Following chunk-wise conditioning (Gao et al., 2026), four actions modulate each future latent step; the previous observation supplies spatial history. The latest observation remains a clean anchor. Only adapters and LoRA parameters are trained with future-latent flow matching; the pretrained transformer, VAE, and text encoder remain frozen. The BridgeV2 run uses two observed and twelve future frames, training-only action normalization, and AdamW with a cosine learning-rate schedule from 10^{-4} to 10^{-5}. A disjoint 32-example monitor selects the 8,000-update checkpoint using LPIPS-Alex; the final 128-example evaluation uses LPIPS-VGG. Inference uses 50 UniPC steps (Zhao et al., 2023), with guidance scale and time shift both set to 5.

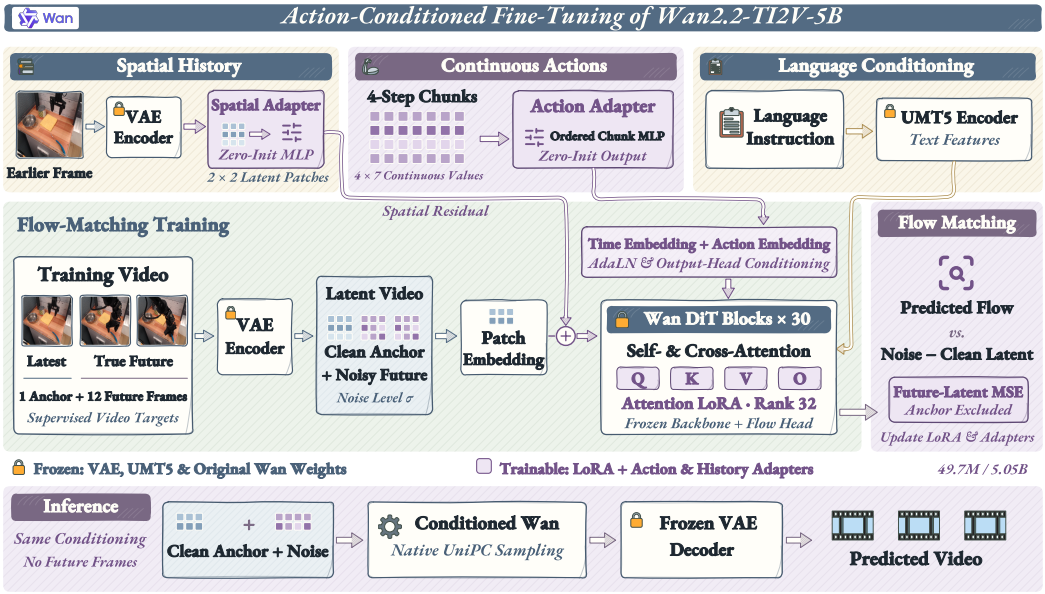

Figure 8: **Action-conditioned Wan2.2 adaptation.** Action chunks modulate time embeddings, and a history adapter supplies spatial context. Adapters and LoRA parameters are trainable; pretrained weights remain frozen.

Evaluation metrics. PSNR measures pixel fidelity as $-10\log_{10}(\text{MSE})$ in decibels for RGB images in $[0, 1]$. SSIM compares local luminance, contrast, and structure. LPIPS (Zhang et al., 2018) measures perceptual distance with pretrained VGG features and inputs in $[-1, 1]$. Higher PSNR/SSIM and lower LPIPS are better. Frame-level scores, including PSNR in decibels, are averaged over the prediction horizon and then over trajectories.

Release plan. We plan to release all implementation code and trained checkpoints, including adaptation configurations and evaluation scripts.

B.5 Learning-Curve and Qualitative Evaluation

Figure 4(a,b) evaluates 32 fixed monitoring clips per dataset with 10-frame prediction and a beam width of four. All checkpoints share inputs and decoding settings. Circles show measurements, curves are endpoint-preserving smoothed guides, and stars mark 200-update MemSPO checkpoints. SFT and RL use different horizontal scales; update counts do not imply equal compute.

Figure 4(c) shows selected improved examples at frame 32. The RT-1, BridgeV2, and RoboCasa models use 500, 200, and 400 MemSPO updates, respectively; the BridgeV2 example uses the base-reward checkpoint underlying the learning curve. Each SFT comparison matches inputs and decoding. Full frames remain visible, and detail crops use identical coordinates across methods. These examples illustrate prediction quality rather than measured robot task success.

B.6 Decoding Sensitivity

Table 3: LPIPS reduction (%) across decoding settings.

Temperature T	Top-p	RT-1	BridgeV2
1.0	1.00	4.02	3.45
0.8	1.00	3.03	2.97
1.2	1.00	4.32	2.60
1.0	0.95	3.70	3.84
1.0	0.90	4.04	3.53

Decoding sensitivity. Table 3 varies temperature T and the nucleus threshold p, keeping the beam width at four. On the same 32 clips per dataset, MemSPO reduces mean LPIPS under all five settings: 3.03–4.32% on RT-1 and 2.60–3.84% on BridgeV2. Each setting uses its own matched SFT reference, so the gains do not depend solely on the default decoding parameters.

B.7 Motion and Dynamic Evaluation

Figure 6 uses the same 128 held-out trajectories per dataset and scores 32 predicted frames at 256×320 resolution. We use WorldArena's RAFT and DINO components (Shang et al., 2026) to construct three reference-based diagnostics. Flow is the mean Euclidean difference between predicted-video and ground-truth optical-flow vectors over adjacent frames and spatial positions. Gap is the absolute difference between their mean flow magnitudes. DINO is the mean cosine similarity between corresponding predicted and ground-truth frame features. Each metric is averaged over trajectories; lower Flow and Gap and higher DINO indicate closer agreement with the recorded future. These diagnostics are supplementary measures rather than the official WorldArena aggregate score.

All metrics use GPU FP32 computation with TF32 disabled. FutureWorlds uses the main-comparison checkpoints: 500, 200 (motion-aware reward), and 400 MemSPO updates for RT-1, BridgeV2, and RoboCasa. RT-1 and RoboCasa videos are regenerated with the original decoding settings, while BridgeV2 retains the saved prediction tokens and decodes them uniformly on GPU. This supplementary evaluation does not replace the historical pixel-metric results. Ten- and twenty-frame evaluations use prefixes of the same rollouts. The qualitative cases are selected improved examples. For each case, the visualized pair has the largest ground-truth mean motion among the three stored pairs beginning at frames 8, 16, and 31. All methods share that pair and a color scale

set by the pooled 99th-percentile flow magnitude; larger magnitudes are clipped. Hue represents direction and saturation represents magnitude. Values in the main figure are rounded to two decimal places.

C Short-Horizon Evaluation

Table 4 complements the 32-frame comparison with ten-frame results, using the same model checkpoints and 128 trajectories per reported dataset. Baseline scores use the first ten frames of their 32-frame rollouts; FutureWorlds uses its existing ten-frame evaluations. We average frame-wise PSNR, SSIM, and LPIPS-VGG over each trajectory and then over trajectories.

Table 4: Ten-frame video prediction on 128 trajectories per reported dataset. Metrics: PSNR↑, SSIM↑, and LPIPS↓.

Method	RT-1			BridgeV2			RoboCasa		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Video Generation Models									
FitVid	19.89	0.6268	0.4529	17.86	0.6241	0.5079	14.29	0.5640	0.5373
iVideoGPT	16.62	0.6142	0.3333	21.91	0.8142	0.1544	17.56	0.7513	0.2009
Action-Conditioned World Models									
IRASim	18.64	0.6771	0.2747	18.65	0.7442	0.2120	14.75	0.6495	0.3069
Vid2World	20.82	0.7803	0.1551	21.21	0.7648	0.1482	16.62	0.7452	0.1873
Ctrl-World	20.67	0.7548	0.2094	20.84	0.7456	0.2071	16.27	0.7335	0.2325
DriftWorld	23.14	0.8185	0.1861	22.72	0.8103	0.1699	17.93	0.7746	0.2203
WEAVER	21.12	0.8038	0.1932	21.20	0.7929	0.1993	15.84	0.7523	0.2308
RL Post-Training									
RLVR-World	21.24	0.7937	0.1447	21.33	0.7702	0.1398	16.98	0.7551	0.1652
PersistWorld	20.56	0.7522	0.2154	21.62	0.7646	0.1907	16.97	0.7481	0.2166
Adapted Video Foundation Models									
Wan2.2-TI2V-5B	20.12	0.7707	0.1892	19.03	0.7784	0.1737	15.53	0.7388	0.2165
Cosmos-Predict2.5-2B	20.12	0.7746	0.2035	21.36	0.8365	0.1301	15.51	0.7382	0.2268
DreamDojo-2B	20.57	0.8197	0.1892	19.31	0.8190	0.1808	15.10	0.7454	0.2322
FutureWorlds (ours)	**24.21**	**0.8345**	**0.1252**	**24.34**	**0.8318**	**0.1072**	**20.18**	**0.8045**	**0.1434**

Native interfaces. IRASim on BridgeV2 receives recorded commands, whereas its released checkpoint is trained with state-derived relative motion. Its RoboCasa result instead uses a target-domain 52-dimensional action adapter.

D Policy Feedback in a Learned World Model

We examine a selected RT-1 drawer-closing example with a frozen, converged RT-1 policy. Starting from the same two recorded observations, the policy acts on each model's latest generated frame, and the world model predicts the next observation from that action. SFT, GRPO, and MemSPO share the initial policy state, random seed, beam width of four, and memory configuration; both post-trained models use 200 updates. No recorded future frame refreshes the interaction.

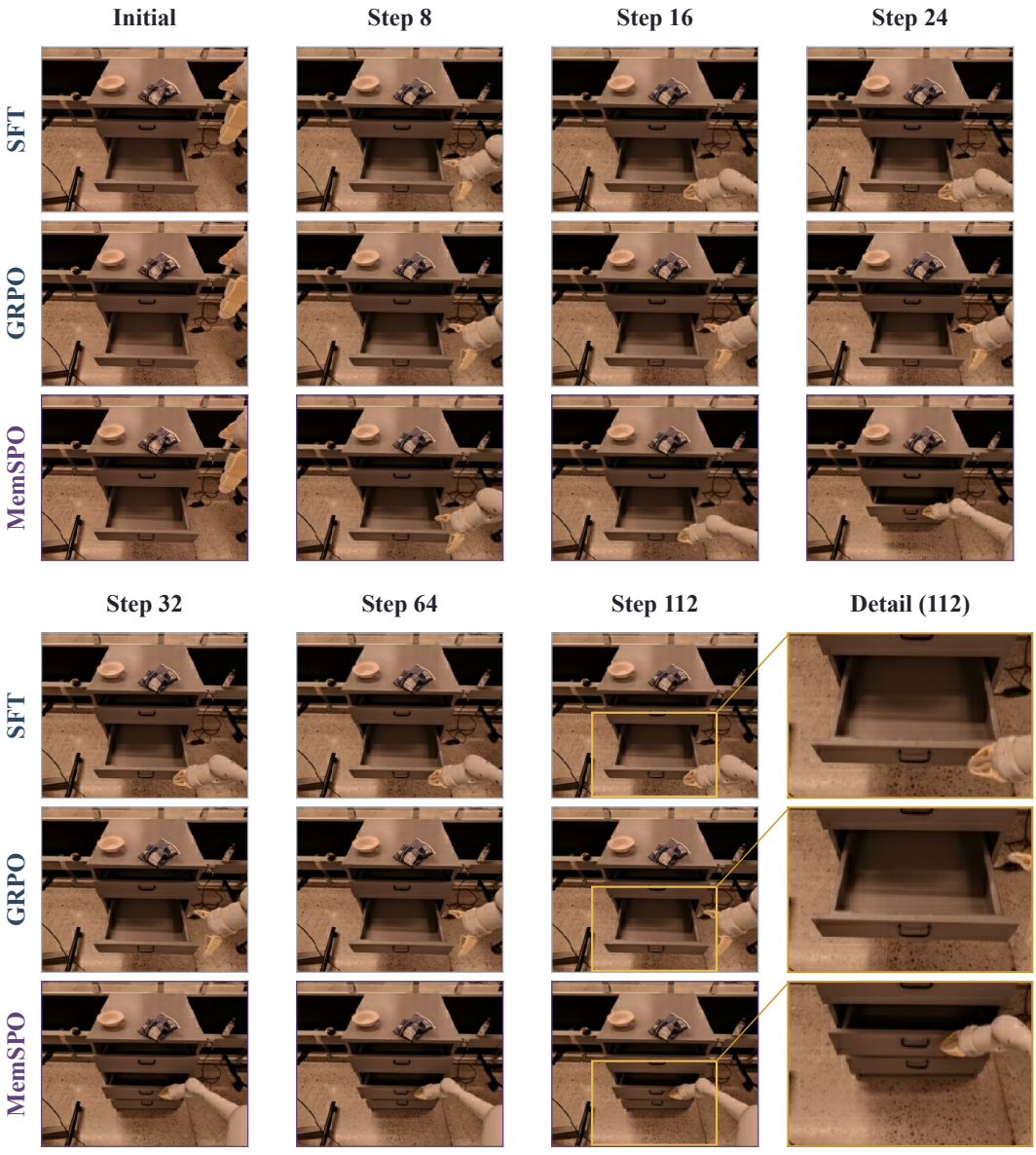

Figure 9: **Policy feedback in a learned world model.** A selected example for *close middle drawer* with a fixed RT-1 policy. The two blocks show the same interaction at seven time points; gold boxes mark matched details at step 112.

Figure 9 shows a visibly smaller drawer opening under MemSPO from step 32 onward, while SFT and GRPO retain larger openings. The contrast illustrates how post-training changes the scene evolution observed by the same policy during feedback interaction. This qualitative example concerns learned-model behavior and does not establish task success in a physical environment.

E Additional Qualitative Comparisons

Figures 10 and 11 extend Figure 3 with six additional cases, two per dataset. We compare saved frame-32 predictions from WEAVER, Cosmos-Predict2.5-2B, DreamDojo-2B, and FutureWorlds against the same recorded future. Full images accompany detail views with identical crop coordinates across methods. These are selected positive examples; Table 1 reports performance over the full evaluation cohorts.

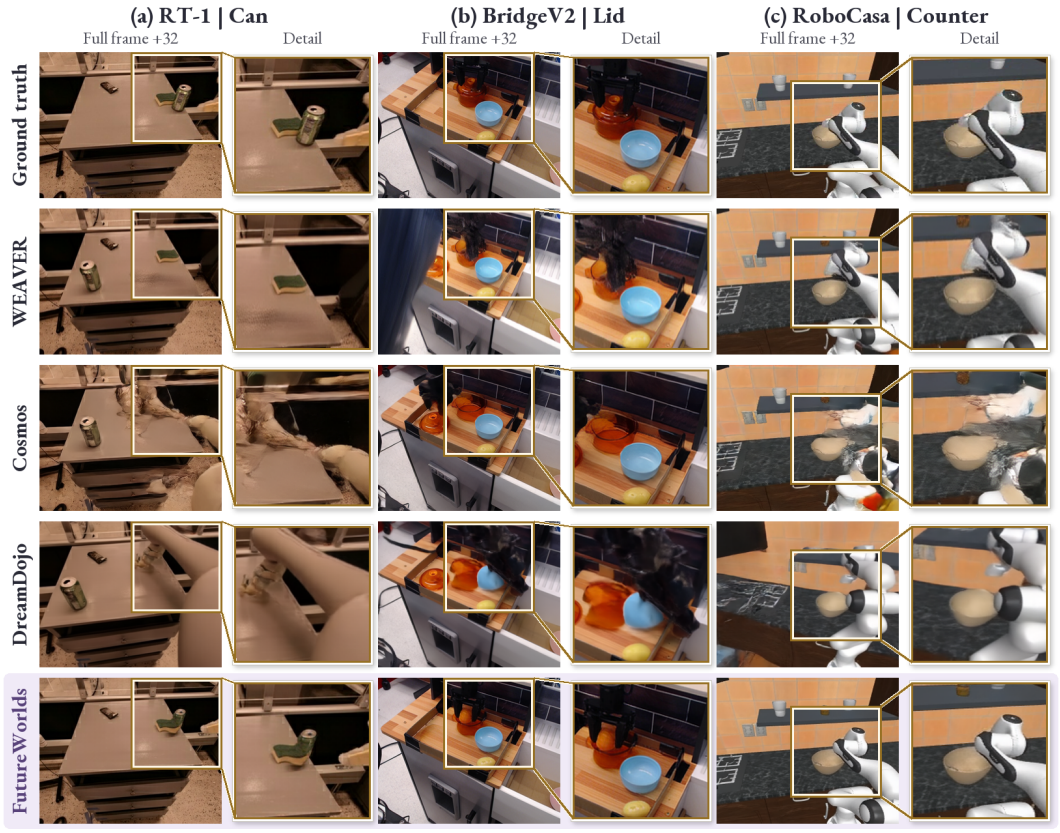

Figure 10: ***Additional comparisons of object placement and robot pose.*** Selected RT-1, BridgeV2, and RoboCasa examples at frame 32. Gold boxes identify matched detail views; model names are abbreviated in the figure.

In the RT-1 case, FutureWorlds places the can nearer the sponge, as in the recorded future, while the shown baselines retain the can near its earlier position. The BridgeV2 example preserves a more recognizable lid–gripper configuration. In RoboCasa, the predicted arm configuration more closely resembles the recorded pose above the counter. These examples illustrate local visual agreement rather than robot task success.

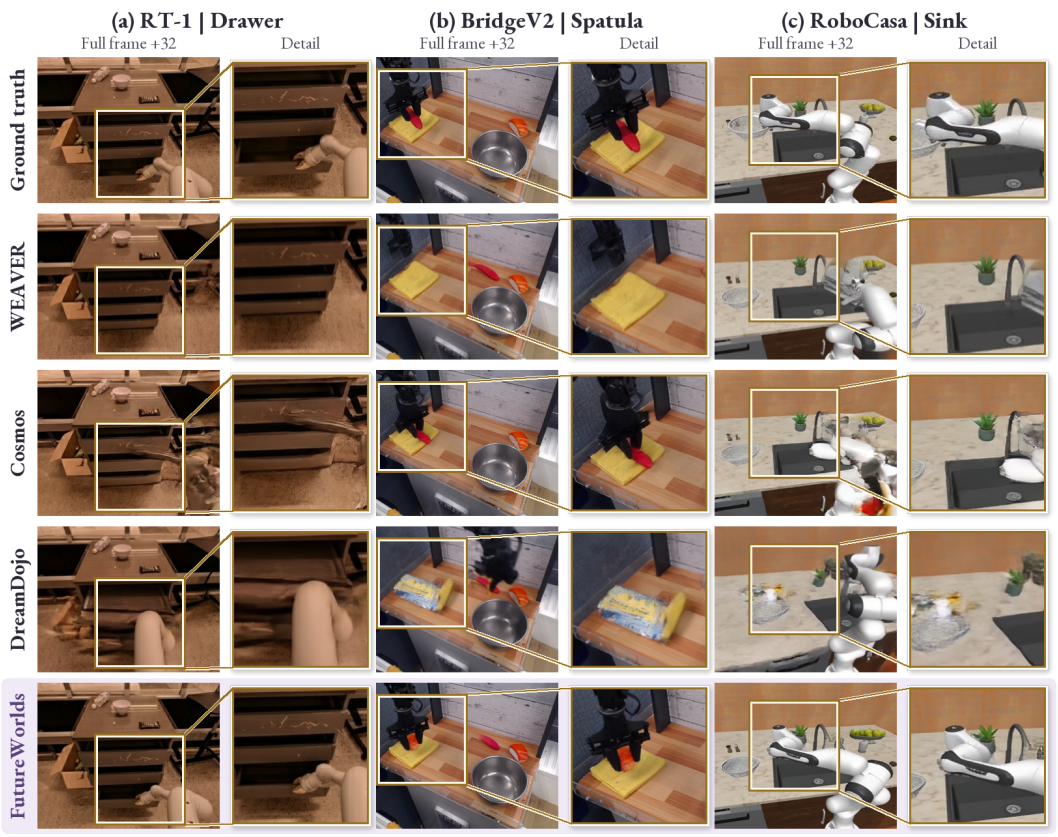

Figure 11: ***Additional comparisons of articulated motion and object appearance.*** Three further selected cases at frame 32, using the same models and display convention as Figure 10.

The RT-1 example shows closer agreement in the drawer opening and gripper location. In BridgeV2, FutureWorlds retains the spatula near the yellow cloth, although its appearance and placement still differ from the reference; Cosmos also preserves the object in this example. The RoboCasa sink case shows a closer arm configuration than the displayed baselines. Full frames expose remaining differences outside the highlighted regions.

E.1 Temporal Comparisons

Figures 12–14 show the evolution of three selected cases from the preceding comparisons. Each row follows one model at the same five future time points, with full frames retained throughout. The aligned sequences expose differences in motion timing, object displacement, and appearance over the rollout, complementing the frame-32 detail views.

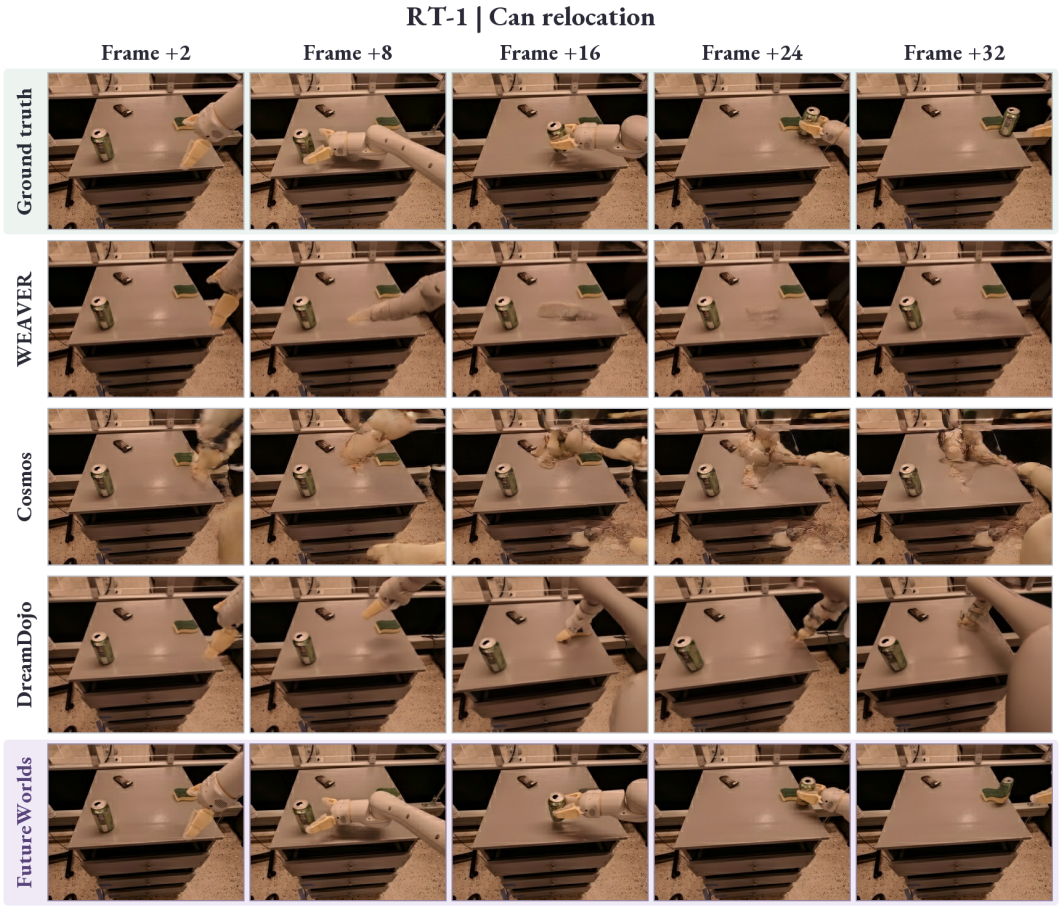

Figure 12: ***Temporal comparison on RT-1.*** The instruction is *move green can near sponge*. Columns show frames +2, +8, +16, +24, and +32 from the same saved prediction sequence for each method. FutureWorlds follows the can's displacement toward the sponge more closely, while the displayed baselines largely retain its earlier position.

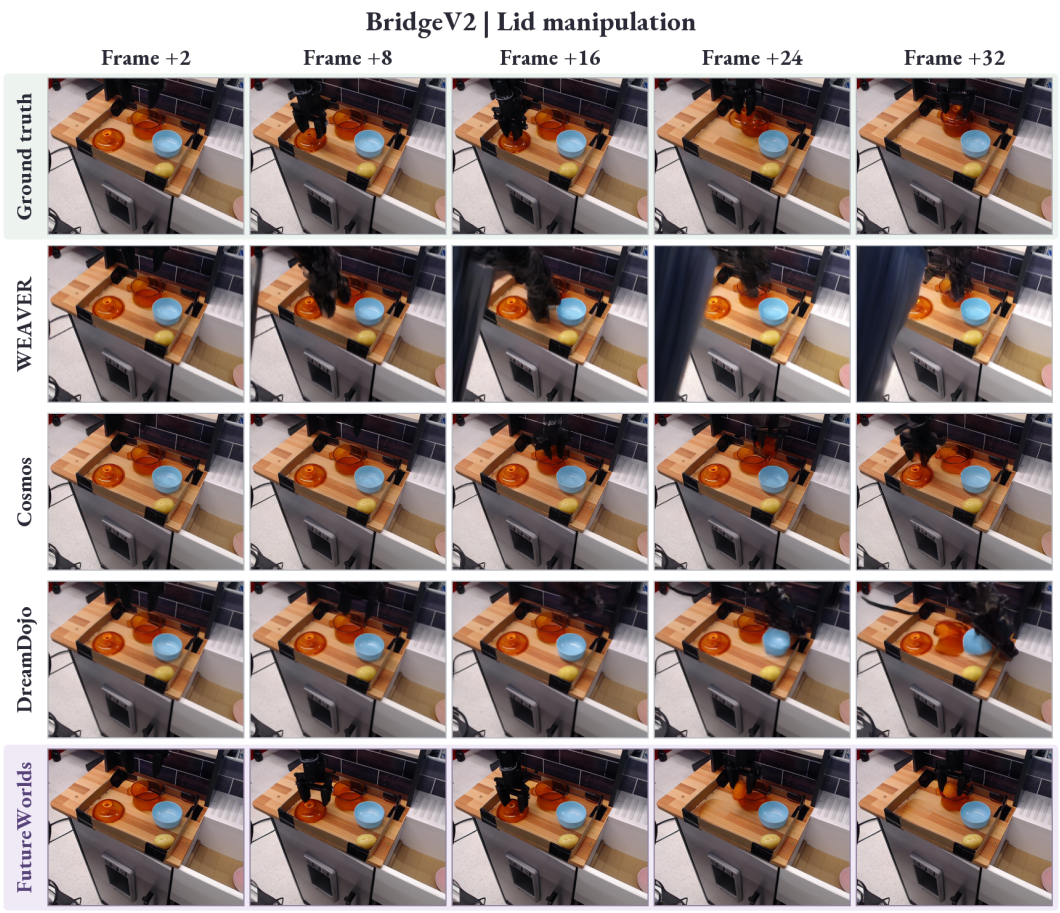

Figure 13: ***Temporal comparison on BridgeV2.*** A selected lid-manipulation sequence at the same five time points as Figure 12. Full frames reveal the evolving gripper–object configuration and remaining differences from the recorded trajectory.

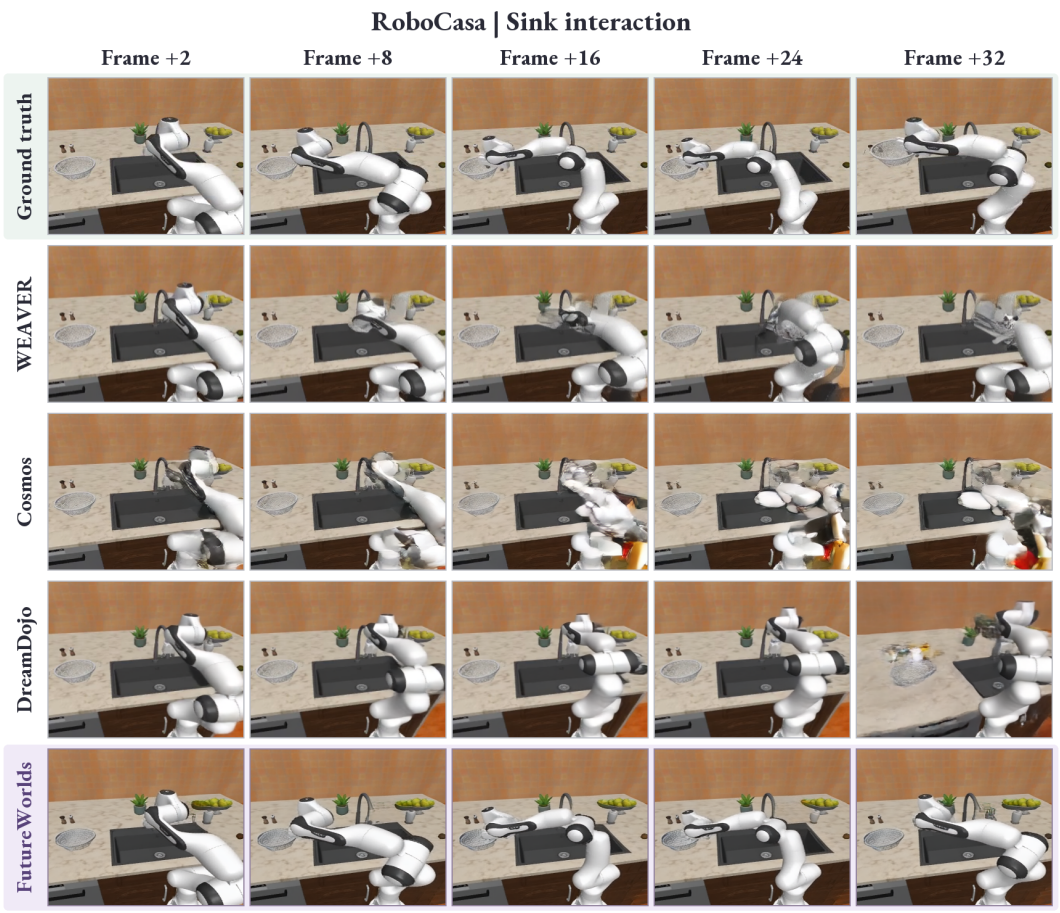

Figure 14: ***Temporal comparison on RoboCasa.*** A selected sink-interaction sequence under the same display convention. FutureWorlds more closely follows the changing arm configuration, while deviations in pose, geometry, and scene appearance are visible in the baseline sequences. These selected examples illustrate prediction behavior, not measured task success.

E.2 Candidate Diversity and Relative Learning Signals

We inspect saved candidates from the first BridgeV2 training batch, before either strategy updates the shared SFT checkpoint. Both strategies use the same inputs, frozen tokenizer, ten-frame horizon, and reward. Ordinary beam search maintains eight global paths and retains the top four; diverse search uses four groups of two with a diversity penalty of 0.1, retaining one candidate per group.

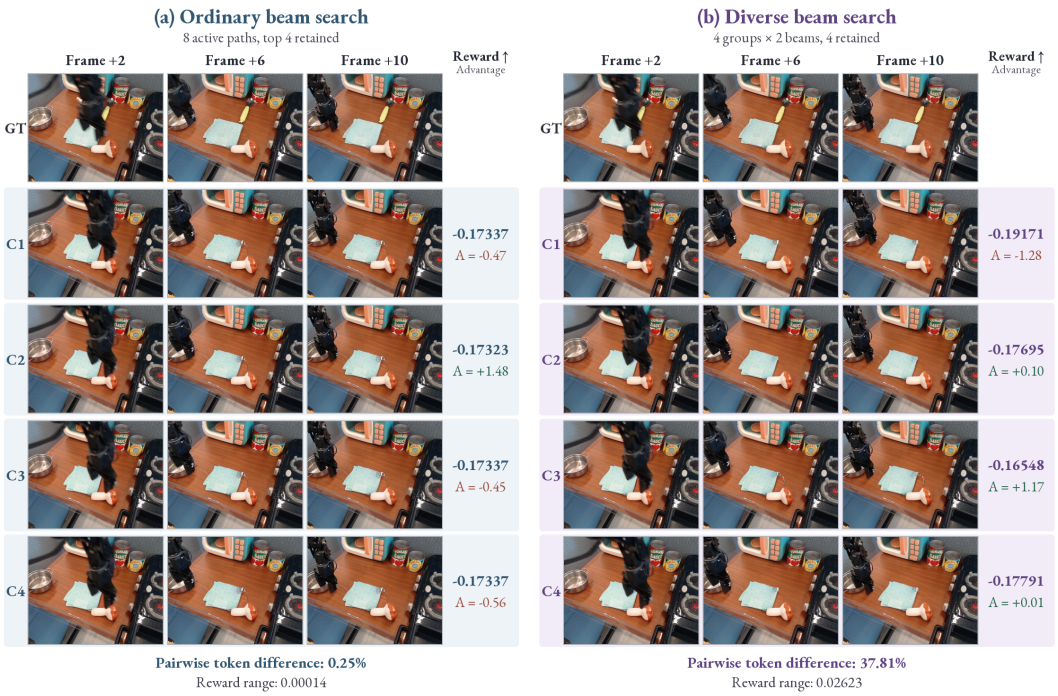

Figure 15: ***Candidate trajectories and relative learning signals.*** Ordinary and diverse beam search from the same SFT model and input. All four retained candidates appear in their original order at frames +2, +6, and +10. Annotations give the original ten-frame reward and group-relative advantage A. The illustrated input has the largest diverse-search reward range among the 16 inputs in the initial batch.

Ordinary search produces nearly identical visible continuations, whereas diverse search exposes differences in gripper configuration and object appearance. Positive and negative advantages give the RL objective alternatives to promote or suppress. Across all 16 inputs in this initial batch, mean pairwise token disagreement is 0.28% for ordinary search and 30.32% for diverse search; the mean within-group reward range is 6.44×10^{-5} and 8.66×10^{-3}, respectively. Token disagreement averages over all six candidate pairs and 10×80 token positions; it is distinct from perceptual diversity.

Diverse search also includes weaker candidates: its mean reward is -0.14353, compared with -0.14041 for ordinary search. These diagnostics illustrate reduced candidate redundancy and a broader quality distribution, rather than an immediate increase in average quality. Since advantages are group-normalized, a wider raw reward range alone does not establish stronger gradients or improved post-training performance. Table 5 complements these candidate-level diagnostics with final-model results.

E.3 Candidate Construction and Final Post-Training Quality

Matched training and evaluation. All RL methods start from the same dataset-specific SFT checkpoint at 50,000 updates and use 200 post-training updates, batch size 16, ten-frame rollouts, and the same first 3,200 training windows in order. We fix the base reward (Appendix B.3), optimizer settings, reference policy, visual tokenizer, memory, and history-matched scoring. Ordinary beam search maintains eight global paths and returns the top four complete trajectories by cumulative model log probability. MemSPO uses four groups of two paths with a diversity penalty of 0.1, returning one trajectory per group; GRPO samples four independent trajectories. At evaluation, all methods use beam width four, two initial observations, and anchor-plus-six-state memory without future ground-truth refreshes. The 10/20-frame scores use prefixes of the same 32-frame predictions.

Table 5: **Post-training with different candidate construction strategies.** Matched GPU evaluation on 128 trajectories per dataset. PSNR↑, SSIM↑, and LPIPS↓; bold marks the best score. Reduction rows compare MemSPO with ordinary beam search using unrounded LPIPS.

Method (updates)	**10 frames**			**20 frames**			**32 frames**		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
RT-1									
SFT (50k)	23.76	0.8298	0.1308	22.49	0.8075	0.1527	21.37	0.7860	0.1734
GRPO (200)	23.97	0.8319	0.1289	22.60	0.8089	0.1517	21.39	0.7858	0.1739
Ordinary beam (200)	24.03	0.8329	0.1279	22.85	0.8124	0.1479	21.70	0.7908	0.1690
MemSPO (200)	**24.33**	**0.8357**	**0.1240**	**23.42**	**0.8192**	**0.1397**	**22.45**	**0.8016**	**0.1565**
LPIPS reduction		3.04%			5.54%			7.37%	
BridgeV2									
SFT (50k)	24.06	0.8300	0.1095	22.74	0.8085	0.1270	21.74	0.7942	0.1448
GRPO (200)	24.11	0.8300	0.1090	22.78	0.8086	0.1266	21.79	0.7944	0.1442
Ordinary beam (200)	24.08	0.8298	0.1095	22.86	0.8091	0.1259	21.89	0.7956	0.1424
MemSPO (200)	**24.31**	**0.8320**	**0.1073**	**22.99**	**0.8108**	**0.1244**	**21.95**	**0.7960**	**0.1421**
LPIPS reduction		1.99%			1.18%			0.20%	
RoboCasa									
SFT (50k)	19.50	0.7957	0.1513	18.22	0.7689	0.1787	17.50	0.7519	0.1971
GRPO (200)	19.68	0.7965	0.1514	18.64	0.7729	0.1758	18.03	0.7581	0.1919
Ordinary beam (200)	19.69	**0.7985**	0.1490	18.57	**0.7746**	0.1745	17.98	**0.7607**	0.1906
MemSPO (200)	**19.92**	0.7984	**0.1476**	**18.88**	0.7743	**0.1718**	**18.29**	0.7600	**0.1877**
LPIPS reduction		0.94%			1.53%			1.51%	

Evaluation consistency. Every row uses GPU FP32 with TF32 disabled and the same ordered 128 cases within its dataset. BridgeV2 controls are reevaluated together with ordinary beam search; Table 2 retains its earlier CPU-generated BridgeV2 results. Checkpoints and inputs match, but CPU/GPU numerical differences can alter discrete beam decisions and subsequent rollouts, so these values are not interchangeable. RT-1 and RoboCasa reuse their verified matched GPU controls. Comparisons in this table use only its internally matched results.

Final-model evidence. MemSPO improves PSNR and LPIPS over ordinary-beam post-training on all three datasets at every tested horizon. At 32 frames, LPIPS falls by 7.37%, 0.20%, and 1.51% on RT-1, BridgeV2, and RoboCasa, respectively. The advantage is metric-dependent: ordinary beam search has lower BridgeV2 MAE at 32 frames (0.042731 versus 0.042887). Together with Figure 15, the final-model results support the usefulness of less redundant candidate construction for post-training. This compares global and grouped diverse search, including their different beam allocation; it does not isolate the diversity coefficient or establish statistical significance.

F MEMORY ABLATION: PROTOCOL AND FULL RESULTS

Controlled inference-time intervention. We hold the MemSPO checkpoint at 200 updates, beam width at four, and the ordered set of 128 held-out trajectories per dataset fixed. Actions, language conditioning where applicable, and the visual tokenizer are identical across memory variants. Only the retained state history changes; the variants do not undergo separate training. Each evaluation generates 32 frames once, with its first 10 and 20 frames providing the shorter-horizon results. Evaluation uses GPU FP32 with TF32 disabled, and no ground-truth future frame refreshes the rollout. These settings reproduce the comparison in Figure 7; the main-table checkpoints use different post-training budgets and reward configurations as detailed in Appendix B.3.

Memory configurations. *w/o Historical Memory* retains only the latest state frame, excluding earlier dynamic history. *w/o Initial Anchor* retains a sliding window of the six most recent state frames without a persistent initial state anchor. *Full Model* retains the initial state anchor alongside the six most recent states. All three variants retain the same fixed initial scene encoding; removing historical memory therefore does not remove all initial visual information. State frames retain their corresponding action tokens, and the action indexing and causal prediction order remain unchanged.

Table 6: **Memory ablation across prediction horizons.** PSNR↑ (dB), SSIM↑ (×100), and MSE↓ ($\times 10^3$), averaged over 128 trajectories per dataset. Bold marks the best unrounded value within each dataset and horizon; display values use two decimals.

Memory configuration	10 frames			20 frames			32 frames		
	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE
RT-1									
w/o Historical Memory	22.45	81.81	7.37	20.81	78.83	11.00	19.89	76.73	13.21
w/o Initial Anchor	24.32	83.56	4.57	23.39	81.85	6.12	22.43	80.07	7.99
Full Model	**24.33**	**83.57**	**4.56**	**23.42**	**81.92**	**6.01**	**22.45**	**80.16**	**7.82**
BridgeV2									
w/o Historical Memory	20.04	79.82	14.38	18.63	77.26	18.34	17.95	75.67	20.51
w/o Initial Anchor	24.29	83.17	4.94	22.94	81.01	6.99	21.93	79.58	8.70
Full Model	**24.31**	**83.20**	**4.91**	**22.99**	**81.08**	**6.90**	**21.95**	**79.60**	**8.68**
RoboCasa									
w/o Historical Memory	18.21	77.28	20.92	17.21	74.69	26.63	16.70	73.18	29.88
w/o Initial Anchor	19.91	79.83	**13.77**	18.85	77.39	**18.34**	18.24	75.91	21.18
Full Model	**19.92**	**79.84**	13.79	**18.88**	**77.43**	18.35	**18.29**	**76.00**	**21.15**

Results and interpretation. Table 6 reports the three metrics shown in Figure 7 at all evaluated horizons. Retaining recent history yields the largest improvement over removing historical memory on every dataset. At 32 frames, the full model achieves the best PSNR, SSIM, and MSE, reducing MSE by 40.83%, 57.69%, and 29.22% relative to the history-removed variant on RT-1, BridgeV2, and RoboCasa. The initial anchor provides smaller additional gains. Its effect varies with horizon and metric: on RoboCasa, removing the anchor yields slightly lower MSE at 10 and 20 frames, while the full model retains higher PSNR and SSIM. These results distinguish the substantial role of recent context from the more modest contribution of a persistent anchor. We report cohort means without inferring statistical significance.

G Inference Resource Measurements

Benchmark scope. Table 7 complements Figure 5 with parameters, memory, and latency for all seven plotted models. Each model uses one NVIDIA H20 on the same node, batch size one, and the first three fixed trajectories in the BridgeV2 evaluation cohort. Models run on separate GPUs rather than sharing a device. Each returns 32 RGB frames at 256×320 resolution, using its native checkpoint, internal resolution, input interface, and decoding settings. Costs use three cases, whereas plotted quality uses all 128 evaluation trajectories.

Table 7: **Native inference resource measurements on BridgeV2.** One H20 per model, batch size one, and 32 predicted frames. Latency summarizes three fixed cases after one complete warmup. Peak memory is the maximum across the three measured runs.

Model	Parameters (B)		Peak memory (GiB)		Latency (s)	
	Inference	Loaded	Allocated	Reserved	Median	Range
iVideoGPT	0.449	0.449	5.85	6.53	4.14	4.12–4.16
WEAVER	0.998	1.134	7.52	9.02	6.31	6.29–6.31
PersistWorld	1.624	2.320	6.68	7.84	22.18	22.16–22.18
Wan2.2-TI2V-5B	5.754	5.754	23.06	23.72	11.18	11.18–11.19
Cosmos-Predict2.5-2B	2.254	2.254	5.97	6.85	12.83	12.77–12.83
DreamDojo-2B	2.278	2.278	12.43	16.26	47.17	47.11–47.19
FutureWorlds (ours)	0.257	0.257	1.89	3.91	26.76	26.68–26.77

Parameter accounting. Inference parameters include the prediction backbone, action/history adapters, and visual encoder–decoder, counting tied tensors once. Offline language encoders and unused task, reward, or critic modules are excluded. Loaded parameters additionally include retained unused modules inventoried in the native pipeline: WEAVER retains a task encoder and reward/critic networks, while PersistWorld retains unused image and text encoders. FutureWorlds contains 127,101,696 backbone parameters and 129,448,016 visual-codec parameters, totaling 256,549,712. Its frozen RL reference and training reward network are absent from inference.

Warmup and timing. We preload input image/action arrays and model weights, disable gradients and TF32, and perform one complete 32-frame warmup. We then time each of the three cases once using a wall-clock timer, synchronizing CUDA immediately before and after inference. The table reports the median and observed range, not a confidence interval. Timing includes input preprocessing, CPU–GPU transfers, visual encoding, prediction, decoding, resizing, and transfer of the output to CPU. Model loading, trajectory-file reading, video saving, and quality metrics are excluded. Text features are precomputed. Wan loads and transfers its cached instruction embedding inside the timed call; this cache-file access is included, while language encoding is excluded.

Memory and residency. Before each measured call, we reset PyTorch's CUDA peak-memory counters after synchronization. Allocated memory counts live tensor allocations; reserved memory also includes the allocator's cached blocks. We report the maximum of each over the three calls, in GiB (2^{30} bytes). Figure 5 uses allocated memory. These measurements include resident model tensors and inference intermediates, not just incremental activation memory. Active components reside on GPU; the benchmark does not add dynamic CPU offloading. WEAVER's unused EMA copy moves to CPU before warmup, whereas the unused modules counted above remain in the native GPU pipeline. The resulting memory costs describe these implementations, rather than minimum architectural requirements.

G.1 Native Precision, Decoding, and Caches

Table 8 records the model-specific execution settings. FP32 and BF16 denote parameter storage; autocast denotes mixed-precision execution with the original storage retained. We preserve native differences rather than forcing a shared precision or resolution that changes the evaluated pipelines. All models produce the same final output dimensions, but internal processing costs differ.

Table 8: **Native settings for the resource benchmark.** Resolution is internal height × width. Chunk lengths count retained future frames; Cosmos and DreamDojo generate a full final chunk and retain its first eight frames.

Model	Resolution	Precision	Generation configuration
iVideoGPT	256×256	FP32; BF16 autocast	Sampling, $T = 1$, top-$k = 100$; chunks 10 + 10 + 10 + 2.
WEAVER	192×320	FP32; BF16 autocast	Native full-memory generation, 16 denoising steps; history 2, memory 6.
PersistWorld	256×320	BF16 predictor/action encoder; FP32 VAE	25 denoising steps, guidance 2; retained chunks 10 + 10 + 10 + 2.
Wan2.2-TI2V-5B	256×320	FP32; BF16 autocast	UniPC, 50 steps, guidance 5, shift 5; direct 32-frame prediction.
Cosmos-Predict2.5-2B	256×320	BF16 predictor and codec	35 denoising steps, guidance 0; retained chunks 12 + 12 + 8.
DreamDojo-2B	480×640	BF16; BF16 autocast	35 denoising steps, guidance 0; retained chunks 12 + 12 + 8.
FutureWorlds	256×320	FP32, without autocast	Beam width 4; persistent memory; 32 autoregressive frames.

Caching and software. Input trajectories are loaded into CPU arrays before measurement; visual encoding remains inside the timed call. Language features are computed offline where used. FutureWorlds reuses the key–value cache within each frame's beam search and starts a new generation call for the next frame's retained prefix. Histories and predicted-state caches are rebuilt for each rollout, rather than reused from the warmup or another case. CUDA allocator caches remain warm between measured calls: the harness calls `empty_cache` before warmup, not between the three timed runs. PyTorch versions are 2.9.1 for FutureWorlds, iVideoGPT, and Wan; 2.7.0+cu128 for WEAVER, Cosmos, and DreamDojo; and 2.5.1+cu124 for PersistWorld.

Interpretation. FutureWorlds uses the fewest inference parameters and the lowest observed allocated memory among these seven implementations, while achieving the best BridgeV2 LPIPS in the main comparison. It is slower than iVideoGPT and several diffusion pipelines under their native settings. The measurements therefore support a quality–model-size–memory tradeoff, not universal inference-speed superiority. Native precision, internal resolution, retained modules, and runtime versions all affect cost; three timed cases provide a small-sample operational comparison rather than a hardware-independent complexity estimate. We do not report FLOPs.