ACCELERATING VIDEO WORLD MODELS AT THE EDGE VIA DEFECT-ADAPTIVE REANCHORING

Anonymous authors
Paper under double-blind review

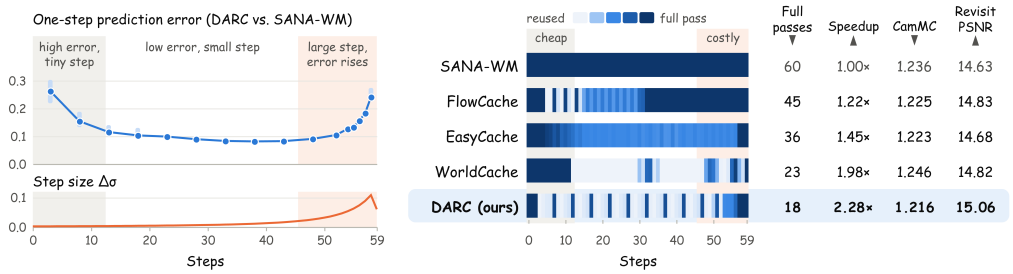


Figure 1: **DARC decides when a video world model can predict instead of recompute. Left (H200):** The baseline SANA-WM (Zhu et al., 2026a) camera-controlled rollout on H200. The strips mark each denoising step as a full model call (filled) or a prediction (hollow). In this scene, WorldCache (Nawaz et al., 2026) brings back a stale pillar at the revisit (red box), while DARC stays close to the baseline (PSNR 24.6 vs. 22.2 dB) with fewer calls and lower latency (432 vs. 497 s). **Top right (Jetson AGX Thor):** DARC gives consistent gains across world-model families, 2.4-3.0× on SANA-WM, Cosmos (Ali et al., 2025), MiniWorld (Zhao et al., 2026) and Astra (Zhu et al., 2026b), which is 5-31% faster than the strongest prior cache while cutting energy by 59-70%. **Bottom right (Cosmos):** DARC's routing signal tracks the true reuse error (Spearman ρ=0.95), whereas the signals of EasyCache (Zhou et al., 2025), WorldCache (Nawaz et al., 2026), FlowCache (Ma et al., 2026) and NaviCache (Lv et al., 2026b) do not ($\rho \leq 0.50$).

ABSTRACT

Video world models require many expensive neural evaluations along their sampling trajectories. Caching skips some of them, but each approximate output alters the subsequent trajectory, and its error is unobservable precisely when the evaluation is skipped. We introduce the **Defect-Adaptive Reanchoring Controller (DARC)**, a training-free method that uses every fresh evaluation twice: as an anchor for predicting complete model outputs, and as feedback on the forecast it replaces. DARC predicts outputs from recent anchors using latent-state and timestep geometry, calibrates this predictor from pre-call forecast errors, and re-anchors once accumulated exposure indicates that further prediction is unreliable. It adds no neural evaluation on skipped steps and leaves the native schedule and solver unchanged. Its routing signal tracks the true reuse error with Spearman ρ=0.95, versus at most 0.50 for prior caches. Across four world-model families and seven benchmarks, DARC is faster and more energy-efficient than prior caches

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.



(a) Where reuse is cheap and where it is costly

(b) Where each method spends its full passes, and the outcome

Figure 2: **When reuse happens matters as much as how often.** **(a)** shows DARC's one-step prediction error against Exact and the step size $\Delta\sigma$. The reuse is affordable early, where steps are tiny, and costly at the end, where both grow. **(b)** shows steps with a full transformer pass (dark). DARC places refreshes where reuse is costly, using the fewest full passes while achieving the highest speedup and the best CamMC and revisit PSNR.

on every family, reaching 2.4-3.0× end-to-end speedup with 59-70% lower energy on the edge-class Jetson AGX Thor and up to 3.8× across five GPU platforms, with near-reference quality. Our code and models will be publicly released.

1 Introduction

Video world models generate visual futures conditioned on observations, text, camera motion, or actions (Zhu et al., 2026a; Ali et al., 2025). Their usefulness depends on whether these generated videos remain consistent with the input conditions, e.g., a camera trajectory should produce the requested viewpoint changes, and revisiting a location should reveal a consistent scene. However, generating these future videos is computationally expensive. Diffusion and flow models repeatedly evaluate a neural network to update a spatiotemporal latent (Ho et al., 2020; Lipman et al., 2023), while autoregressive-based models repeat the sampling process across successive video chunks (Lv et al., 2026a). A practical approach to accelerate such models must reduce this cost while preserving visual quality and fidelity to conditioning.

One approach to reducing this repeated computation is caching, which stores information from earlier sampling steps for reuse in later computations (Ma et al., 2026; Zhou et al., 2025; Yu et al., 2025). In complete-output caching, previous model outputs are reused or extrapolated to replace selected neural evaluations, while the solver continues to perform every scheduled update. The resulting approximation can change the next latent state and therefore the inputs to subsequent model evaluations. A later fresh evaluation uses the state actually reached, so it does not generally restore the trajectory that would have been obtained by evaluating the model at every step. This dependence on earlier reuse decisions is visible in Figure 2, where we can observe that moving reuse between different portions of sampling produces substantially different agreement with the fully evaluated reference, despite similar evaluation counts. Effective caching therefore depends on both predicting skipped outputs accurately and deciding when a fresh evaluation is needed.

Therefore, both these decisions must be made without observing the current model output, since obtaining it would require the evaluation that reuse seeks to avoid. Recent methods guide reuse through online calibration, forecast disagreement, and accumulated risk estimates (Lv et al., 2026b; Li et al., 2026a; Chen et al., 2026). For complete-output prediction, a fresh evaluation provides feedback relevant to both decisions. The observed output serves as a new prediction anchor and can be compared with the forecast constructed before the call. This comparison reveals how the preceding predictor performed at the reached state and provides information for subsequent predictions and refresh decisions. As sampling proceeds, however, the measured discrepancy describes an earlier state rather than the current one. We therefore investigate how this delayed feedback can be combined with inexpensive information about the current state to adapt the predictor and determine when another fresh evaluation is required.

For this purpose, we introduce the **Defect-Adaptive Reanchoring Controller (DARC)**, a training-free controller that combines current state information with feedback from fresh evaluations to adapt complete-output prediction and refresh decisions. DARC retains the two most recent fresh outputs and compact signatures of their input states, obtained by deterministic coordinate sampling. It uses the current state's displacement relative to these anchors, together with timestep progress, to extrapolate the change between the stored outputs. Furthermore, combining this secant forecast with the latest fresh output moderates the extrapolation. After initialization, DARC uses each fresh output to measure the error of the forecast saved before the call and fit a newly fused coefficient. In parallel, the calibration and error measurement are performed in the representation provided to the solver. The fitted coefficient and measured error then inform prediction and refresh decisions in the next anchor segment.

Now, to determine when this calibrated predictor should be refreshed, DARC combines the measured forecast error with current state geometry. It tracks the disagreement between state and timestep progress, together with deviation from the chord joining the anchor state signatures. These quantities determine an exposure score that accumulates within the current segment, weighted by native timestep gaps. DARC requests a fresh evaluation when this score reaches its budget or when the span limit is reached, and lifecycle safeguards require a refresh. The score serves as an empirical criterion for reanchoring and does not provide a certified trajectory-error bound. This control loop adapts the predictor and the timing of fresh evaluations without additional neural probes on transported steps, while at the same time preserving the pretrained checkpoint, native timestep grid, and solver update.

We evaluate DARC across four world-model families, SANA-WM (Zhu et al., 2026a), Cosmos-Predict2.5 (Ali et al., 2025), MiniWorld (Zhao et al., 2026), and Astra (Zhu et al., 2026b), covering camera-controlled, robot-action-conditioned, and text-conditioned generation settings. We further assess its practical efficiency across five GPU platforms, ranging from datacenter accelerators (H200) to edge devices (AGX Thor), by reporting latency as well as energy consumption under matched workloads and execution settings. Our key contributions are as follows:

- **State-conditioned complete-output prediction.** We develop a predictor that combines state and schedule geometry with a hold-secant blend calibrated online in the solver-facing representation, while preserving all native solver updates.
- **Adaptive reanchoring through forecast feedback.** We combine measured pre-call forecast errors with current state geometry in a segment-local exposure criterion. The fresh evaluations thereby provide feedback for both predictor calibration and subsequent refresh decisions.
- **Evaluation across different models, tasks, and hardware.** We assess generation quality and task fidelity across four world-model families on six benchmarks and measure inference efficiency across five GPU platforms, reporting realized latency, throughput, and energy alongside model evaluation counts.

2 Related Work

Caching for video and world models. Caching is harder in video world models because the predictability of intermediate representations varies with scene motion, spatial content, token type, and autoregressive context. Content-aware WorldCache (Nawaz et al., 2026) adapts reuse to scene dynamics through motion-dependent thresholds and saliency-weighted drift and improves reused features through blending and warping. FlowCache (Ma et al., 2026) considers variation across autoregressive video chunks by assigning each chunk an independent caching policy and combining this policy with KV-cache compression. These methods show that reuse policies must account for the structure and evolving content of world-model generation. They also leave a central question of how the reliability of a cached prediction should be assessed as the sampling trajectory progresses.

Adaptive prediction and refresh. As the error of a prediction cannot be observed at the step where it replaces an evaluation, controllers that combine prediction with refresh must decide from other signals when to recompute. NaviCache (Lv et al., 2026b) updates a recursive estimate of the ratio between input and output-feature changes at each fresh computation, and triggers refresh once the accumulated *predicted* output variation grows large. DARC instead measures how wrong its forecast *actually was*: at each fresh computation, it compares the saved pre-call forecast with the fresh output

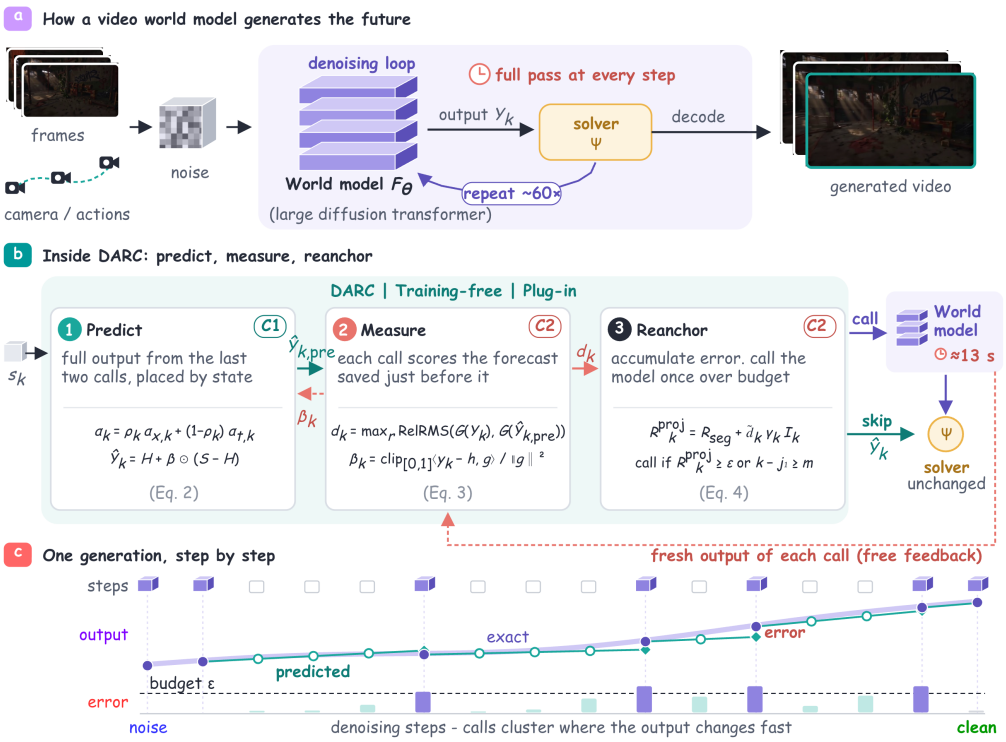


Figure 3: **Overview of DARC. (a)** A video world model runs its diffusion transformer F_θ at each of $\sim$60 denoising steps, which dominates inference cost. **(b)** DARC is a training-free plug-in, which either calls F_θ or predicts its output at each step, leaving the solver Ψ unchanged. *Predict* (C1) forecasts the full output from the last two calls, placed by the current state. *Measure* and *Reanchor* (C2) use each call as free feedback: the fresh output scores the preceding forecast, and the accumulated defect triggers the next call at budget ε (or after m predictions). **(c)** shows generation trajectory where filled steps call the model and empty steps predict (teal).

at the same state and uses this discrepancy to calibrate both its predictor and its subsequent refresh decisions. The distinction matters because expected change and realized error can diverge. A large but predictable change incurs little defect, whereas a small mispredicted one can incur a large defect.

3 METHOD

Diffusion and flow-based world models generate a sample by repeatedly running a neural model along a prescribed numerical trajectory. At each node, the model processes the current spatiotemporal latent and conditioning, and its output drives the next solver update. Autoregressive world models on the other side repeat this trajectory across successive video chunks, making these neural evaluations the principal target for acceleration. Figure 3 shows our main method, named after Defect-Adaptive Reanchoring Controller (DARC), which operates at the boundary between the world model and its solver. It preserves the checkpoint, conditioning, timestep sequence, solver nodes, and native update, while selecting whether each node receives a fresh or predicted complete model output. DARC closes a prediction $\rightarrow$ measurement $\rightarrow$ reanchoring loop: two fresh evaluations define a local predictor, a later fresh evaluation measures the forecast saved before that call, and this causal feedback governs the next refresh.

3.1 WORLD-MODEL GENERATION AND CAUSAL ROUTING

Core Question: *How should fresh evaluations be allocated when the same-state transport defect is revealed only by performing the call under consideration?*

Consider one descending pass with N native solver nodes. Let $\mathbf{s}_k$ denote the complete numerical state at node k, including the current latent and any solver history, and let $\mathbf{x}_k = \pi_x(\mathbf{s}_k)$ be the corresponding model input. The world model F_θ produces a complete boundary output $\mathbf{Y}_k$ from $\mathbf{x}_k$, timestep t_k, and conditioning $\boldsymbol{\xi}$ (Camera or action based conditioning). A deterministic adapter G converts this tensor to the solver-facing quantity, and the native update Ψ_k advances the state. Supposing superscript d denotes the trajectory reached by DARC, the controller chooses $a_k \in \{0, 1\}$ before any possible model call and executes

$$\begin{aligned} \mathbf{x}_k^d &= \pi_x(\mathbf{s}_k^d), \\ \widetilde{\mathbf{Y}}_k &= \begin{cases} F_\theta(\mathbf{x}_k^d, t_k; \boldsymbol{\xi}), & a_k = 1, \\ \widehat{\mathbf{Y}}_k, & a_k = 0, \end{cases} \\ \mathbf{s}_{k+1}^d &= \Psi_k\Big(\mathbf{s}_k^d, G(\widetilde{\mathbf{Y}}_k)\Big). \end{aligned} \tag{1}$$

Thus, $a_k = 1$ requests a fresh evaluation and $a_k = 0$ supplies a transported prediction. Here, both choices execute the same solver update. The resulting count $N_{\text{fresh}} = \sum_{k=0}^{N-1} a_k$ is thus input and trajectory-dependent. We use EXACT for the matched discrete reference that evaluates F_θ at every native node, rather than an exact continuous-time solution.

Now, the ideal routing signal would compare $G(\widehat{\mathbf{Y}}_k)$ with the fresh output evaluated at the same reached state. That comparison is unavailable before routing because it requires the model call under consideration. Moreover, a transported output changes the next state and hence all subsequent model inputs. Hence, a later fresh call cannot restore the fully evaluated trajectory. DARC therefore uses delayed measurements from fresh calls together with current state geometry. This is also evident in our ablation shown in Appendix F.

3.2 STATE-CONDITIONED COMPLETE-OUTPUT TRANSPORT

Core Question: *How can DARC predict the complete output from recent fresh evaluations and current state geometry without an additional model call?*

The model output depends on both the timestep and the current latent state. Now, as earlier transported outputs can alter the state reached by the solver, the timestep progress alone may not capture how the output has evolved. DARC therefore combines timestep progress with current state geometry to predict the output from previous fresh evaluations.

For this purpose, DARC retains the two most recent fresh anchors $j_0 < j_1$, with $\mathcal{A}_j = (j, t_j, \mathbf{z}_j, \mathbf{Y}_j)$ and $\mathbf{z}_j = P_x(\mathbf{x}_j^d)$. Each anchor stores its solver index, timestep, compact deterministic latent state signature, and complete model output, respectively. For a candidate node $k > j_1$, DARC estimates the progress beyond the latest anchor in two ways. The schedule coordinate $\alpha_{t,k}$ measures timestep progress relative to the anchor interval, while the state coordinate $\alpha_{x,k}^{(i)}$ measures the current signature displacement along the chord joining the two anchor signatures. A regularized axis-alignment score $\rho_k^{(i)}$ weights these estimates, where a strong alignment favors the state coordinate, while a weak alignment shifts weight toward the schedule coordinate.

The combined coordinate thus determines a secant prediction that extends the output change between the two anchors. To moderate this extrapolation, DARC integrates the secant prediction with a zero-order hold of the latest fresh output:

$$\begin{aligned} \alpha_k^{(i)} &= \rho_k^{(i)} \alpha_{x,k}^{(i)} + (1 - \rho_k^{(i)}) \alpha_{t,k}, \\ \mathbf{H}_{k|j_1} = \mathbf{Y}_{j_1}, &\qquad \mathbf{S}_{k|j_0,j_1}^{(i)} = \mathbf{Y}_{j_1}^{(i)} + \alpha_k^{(i)} \big(\mathbf{Y}_{j_1}^{(i)} - \mathbf{Y}_{j_0}^{(i)}\big), \\ \widehat{\mathbf{Y}}_k &= \mathbf{H}_{k|j_1} + \mathcal{E}_G(\boldsymbol{\beta}_{j_1}) \odot \big(\mathbf{S}_{k|j_0,j_1} - \mathbf{H}_{k|j_1}\big). \end{aligned} \tag{2}$$

Here i indexes state-batch elements and $\boldsymbol{\beta}_{j_1} \in [0, 1]^{B_G}$ contains the carried fused coefficients for the B_G solver-view samples. Thus, setting $\beta = 0$ recovers the hold ($\mathbf{H}_{k|j_1}$), while $\beta = 1$ recovers the secant prediction ($\mathbf{S}_{k|j_0,j_1}$), with intermediate values dampening the extrapolation. The operator $\mathcal{E}_G$ expands these coefficients to the adapter-output representation and $\odot$ denotes elementwise multiplication. The fresh evaluations calibrate the coefficients for subsequent predictions, as described in the next subsection.

Furthermore, the same state geometry provides a normalized chord-mismatch signal $r^{(i)}_{\mathrm{ch},k}$, which informs the refresh decision when the current displacement departs from the anchor-based prediction. Appendix J gives the exact geometric definitions, clipping rules, numerical floors, adapter compatibility condition, and secant-only variant.

3.3 Fresh-anchor calibration and defect feedback

Question: *How a fresh evaluation assess the current predictor and calibrate the next anchor segment?*

When a node k is selected for a fresh evaluation, DARC first saves the counterfactual prediction $\widehat{\mathbf{Y}}_{k,\mathrm{pre}}$ formed from the old anchors. After observing $\mathbf{Y}_k = F_\theta(\mathbf{x}^d_k, t_k; \boldsymbol{\xi})$, it uses that same output to calibrate the hold-secant mixture and measure the saved forecast. Let $\boldsymbol{\varphi}(\mathbf{Y}) = P_y(G(\mathbf{Y}))$ be a compact, deterministic solver-view signature. For the solver-view sample r, we define $\mathbf{y}^{(r)}_k = \boldsymbol{\varphi}(\mathbf{Y}_k)^{(r)}$, $\mathbf{h}^{(r)}_k = \boldsymbol{\varphi}(\mathbf{H}_{k|j_1})^{(r)}$, and $\mathbf{g}^{(r)}_k = \boldsymbol{\varphi}(\mathbf{S}_{k|j_0,j_1})^{(r)} - \mathbf{h}^{(r)}_k$. Thus, DARC computes the following:

$$\beta^{(r)}_k = \mathrm{clip}_{[0,1]} \frac{\langle \mathbf{y}^{(r)}_k - \mathbf{h}^{(r)}_k, \mathbf{g}^{(r)}_k \rangle}{\max(\|\mathbf{g}^{(r)}_k\|_2^2, \eta_y)},$$
$$d_k = \max_r \mathrm{RelRMS}\Big(G(\mathbf{Y}_k)^{(r)}, G(\widehat{\mathbf{Y}}_{k,\mathrm{pre}})^{(r)}\Big). \tag{3}$$

Here, the first expression $(\beta^{(r)}_k)$ is a stabilized, clipped scalar fit in the sampled solver view. The second measures the full solver-facing forecast defect, with RelRMS defined in the Appendix J. Both are computed before the anchors advance. The β_k uses candidates from the old anchors, while d_k scores the saved pre-call forecast at the state reached by DARC. It is not a discrepancy from an independently executed Exact trajectory. DARC then advances $(j_0, j_1) \leftarrow (j_1, k)$ and carries β_k and d_k into the next segment. The fresh output remains the output supplied at node k and thus the calibration acts only prospectively.

3.4 Defect-exposure control and adaptive reanchoring

Core Question: *How can DARC combine the last measured forecast defect with current trajectory geometry to decide when to reanchor?*

The measured defect d_{j_1} is grounded in a fresh output but becomes lagged as the transport continues. DARC complements it with the current state-time gap $q_k = \max_i |\alpha^{(i)}_{x,k} - \alpha_{t,k}|$, its capped value $\bar{q}_k = \min(q_k, q_{\max})$, the batch-reduced chord mismatch $r_{\mathrm{ch},k} = \max_i r^{(i)}_{\mathrm{ch},k}$, and the carried secant magnitude $\beta^{\max}_{j_1} = \max_r |\beta^{(r)}_{j_1}|$. Now, subject to the safeguards described below, the controller determines the transport action using the following scores, where R_{seg} denotes the exposure committed since the most recent fresh call:

$$\widetilde{d}_k = \max\big(d_{j_1}, w_{\beta q}\beta^{\max}_{j_1} q_k\big), \qquad I_k = 1 + w_{\mathrm{ch}} r_{\mathrm{ch},k} + w_q \bar{q}_k,$$
$$\gamma_k = \frac{|t_k - t_{k-1}|}{T_{\mathrm{norm}}}, \qquad R^{\mathrm{proj}}_k = R_{\mathrm{seg}} + \widetilde{d}_k \gamma_k I_k,$$
$$a_k = 0 \iff R^{\mathrm{proj}}_k < \varepsilon \quad \text{and} \quad k - j_1 < m. \tag{4}$$

Here ε is the empirical segment budget, m is the maximum anchor span, $q_{\max}$ caps the coordinate gap, $w_{\beta q}, w_{\mathrm{ch}}, w_q \geq 0$ are fixed controller weights, and $T_{\mathrm{norm}} > 0$ normalizes native timestep gaps. The observed defect supplies the baseline; the uncapped $\beta^{\max}_{j_1} q_k$ term raises the proxy when state and schedule progress disagree under an active secant contribution; and I_k inflates exposure under current trajectory mismatch. The factor γ_k weights this exposure by the native timestep gap. Transport commits $R_{\mathrm{seg}} \leftarrow R^{\mathrm{proj}}_k$; otherwise, DARC evaluates the model, measures the saved forecast, updates the anchors and calibration, and resets $R_{\mathrm{seg}} \leftarrow 0$.

4 Experiments and Results

Models and Benchmarks. Our evaluation protocol spans four world-model families, five checkpoint configurations, and seven evaluation settings covering text, image, camera, and action-conditioned

Table 1: **Quality and efficiency on H200 across four world models and seven benchmarks.** The first row of each model block is its unaccelerated baseline, and every method is compared only with the BASELINE row of the same benchmark. Small second-line values give the change, speedup, or energy saving relative to that row. Latency and energy are measured separately for each benchmark and are not compared across benchmarks. Among accelerated methods, the best result in each column is in **bold** and the second best is underlined.

Model / method	VBench-I2V				SANA-Bench			
	I2V ↑	Total ↑	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑	$\text{CamMC}_{\text{rel}}$ ↓	Revisit P/S/L ↑ / ↑ / ↓	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑
SANA-WM	97.7	87.6	39.9 (1.0×)	6.2 (0.0%)	123.6	14.6 / 31.1 / 44.1	986.2 (1.0×)	138.8 (0.0%)
+ EasyCache	97.7 (0.0)	87.7 (0.0)	22.5 (1.8×)	3.7 (40.2%)	122.3 (−1.3)	14.7 / 31.2 / 44.0 (+0.05/+0.08/−0.12)	679.8 (1.5×)	92.3 (33.5%)
+ WorldCache	**98.4** (+0.7)	**88.0** (+0.4)	20.2 (2.0×)	3.1 (50.0%)	124.6 (+1.0)	14.8 / 31.2 / 43.8 (+0.19/+0.07/−0.33)	497.1 (2.0×)	59.9 (56.9%)
+ FlowCache	97.7 (0.0)	87.7 (0.0)	30.2 (1.3×)	5.0 (19.3%)	122.6 (−1.1)	14.8 / 31.4 / 43.9 (+0.20/+0.23/−0.21)	810.2 (1.2×)	113.8 (18.0%)
+ NaviCache	97.7 (0.0)	87.7 (0.0)	29.9 (1.3×)	4.8 (23.1%)	122.2 (−1.4)	14.9 / 31.5 / 44.8 (+0.27/+0.39/+0.49)	815.0 (1.2×)	112.7 (18.9%)
+ DARC (Ours)	97.6 (−0.1)	87.4 (−0.2)	**11.9** (3.3×)	**2.2** (65.0%)	**121.6** (−2.0)	**15.1 / 31.9 / 43.6** (+0.44/+0.80/−0.53)	**431.7** (2.3×)	**54.4** (60.8%)

Model / method	VBench (T2V)				PAI-Bench-G (I2W)			
	Quality ↑	Total ↑	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑	Quality score ↑	Domain score ↑	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑
Cosmos-Predict2.5-2B	85.4	84.1	289.6 (1.0×)	46.1 (0.0%)	76.1	84.5	55.0 (1.0×)	8.7 (0.0%)
+ EasyCache	**85.4** (0.0)	**84.1** (−0.1)	237.7 (1.2×)	37.9 (17.8%)	75.5 (−0.6)	84.0 (−0.5)	40.3 (1.4×)	8.8 (-0.4%)
+ WorldCache	85.0 (−0.4)	83.4 (−0.7)	129.5 (2.2×)	23.1 (49.8%)	75.6 (−0.5)	84.0 (−0.6)	24.5 (2.2×)	6.5 (25.6%)
+ FlowCache	84.8 (−0.6)	83.8 (−0.3)	252.3 (1.1×)	40.3 (12.7%)	76.0 (−0.1)	**84.7** (+0.2)	53.6 (1.0×)	8.3 (4.7%)
+ NaviCache	85.4 (−0.1)	84.1 (−0.1)	289.2 (1.0×)	46.6 (-0.9%)	**76.0** (−0.1)	84.4 (−0.1)	54.6 (1.0×)	8.6 (1.2%)
+ DARC (Ours)	85.1 (−0.3)	83.9 (−0.2)	**104.3** (2.8×)	**16.5** (64.3%)	76.0 (−0.1)	83.9 (−0.6)	**23.1** (2.4×)	**5.9** (32.0%)

Model / method	DROID-100				RealEstate10K			
	FVD ↓	PSNR/SSIM/LPIPS ↑ / ↑ / ↓	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑	FVD ↓	PSNR/SSIM/LPIPS ↑ / ↑ / ↓	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑
MiniWorld-1B	52.4	23.7 / 83.9 / 6.6	71.0 (1.0×)	8.5 (0.0%)	203.7	15.3 / 50.6 / 39.0	72.2 (1.0×)	9.1 (0.0%)
+ EasyCache	61.5 (+9.1)	23.4 / 82.9 / 6.8 (−0.27/−1.03/+0.19)	40.1 (1.8×)	4.7 (44.3%)	207.2 (+3.6)	15.0 / 50.4 / 39.8 (−0.26/−0.25/+0.75)	45.0 (1.6×)	5.6 (38.4%)
+ WorldCache	53.1 (+0.8)	23.7 / 84.1 / 6.5 (+0.04/+0.17/−0.04)	47.6 (1.5×)	5.6 (33.5%)	205.4 (+1.7)	**15.3** / 50.8 / 39.1 (+0.02/+0.19/+0.12)	48.2 (1.5×)	6.0 (33.7%)
+ FlowCache	**52.6** (+0.3)	23.7 / 83.9 / 6.6 (0.00/0.00/+0.01)	68.6 (1.0×)	8.2 (3.6%)	**204.5** (+0.9)	**15.3** / 50.7 / **39.0** (+0.02/+0.04/−0.07)	69.6 (1.0×)	8.8 (3.6%)
+ NaviCache	57.8 (+5.5)	23.5 / 83.5 / 6.6 (−0.15/−0.41/+0.07)	48.4 (1.5×)	5.6 (34.6%)	205.9 (+2.2)	15.1 / 50.5 / 39.6 (−0.20/−0.19/+0.56)	48.9 (1.5×)	5.9 (35.6%)
+ DARC (Ours)	52.9 (+0.6)	**23.9 / 84.1 / 6.3** (+0.24/+0.18/−0.26)	**31.4** (2.3×)	**3.9** (53.5%)	204.5 (+0.9)	15.2 / **50.9** / 39.4 (0.00/+0.29/+0.33)	**30.7** (2.4×)	**5.0** (44.7%)

Model / method	Sekai			
	FVD ↓	PSNR/SSIM/LPIPS ↑ / ↑ / ↓	s/video / speedup ↓ / ↑	Wh/video / saving ↓ / ↑
Astra	213.1	17.6 / 57.9 / 49.9	70.0 (1.0×)	10.5 (0.0%)
+ EasyCache	227.5 (+14.5)	17.9 / 58.8 / **49.6** (+0.28/+0.84/−0.28)	39.0 (1.8×)	5.7 (45.5%)
+ WorldCache	221.8 (+8.7)	17.7 / 58.0 / 49.9 (+0.05/+0.09/−0.03)	39.2 (1.8×)	5.7 (46.1%)
+ FlowCache	**215.3** (+2.2)	17.8 / 58.3 / 49.7 (+0.13/+0.35/−0.19)	62.5 (1.1×)	9.4 (10.8%)
+ NaviCache	216.3 (+3.9)	17.7 / 58.1 / 49.9 (+0.03/+0.13/−0.02)	58.4 (1.2×)	8.7 (17.5%)
+ DARC (Ours)	223.2 (+10.1)	**18.0 / 59.3** / 49.9 (+0.38/+1.37/+0.03)	**28.3** (2.5×)	**4.0** (61.8%)

generation. We evaluate SANA-WM (Zhu et al., 2026a) on SANA-Bench and VBench-I2V (Huang et al., 2025), and Cosmos-Predict2.5-2B Text2World (Ali et al., 2025) on PAI-Bench-G (Zhou et al., 2026) and VBench-T2V (Huang et al., 2024). Furthermore, MiniWorld (Zhao et al., 2026) provides separate checkpoints for its two conditioning regimes. We evaluate MiniWorld-1B-DROID on DROID and MiniWorld-1B-RealEstate10K on RealEstate10K. We further evaluate Astra (Zhu et al., 2026b) on a fixed held-out subset of the Sekai dataset (Li et al., 2026b), using the Astra-Bench metric suite. Dataset splits, input construction, and generation settings are provided in the Appendix A.

Caching Methods. We compare DARC with EasyCache (Zhou et al., 2025), WorldCache (Nawaz et al., 2026), FlowCache (Ma et al., 2026), and NaviCache (Lv et al., 2026b). For the comparison with vanilla model state, we use EXACT as a reference which evaluates the model at every native evaluation node. Within each model-benchmark setting, all methods use the same checkpoint, conditioning, timestep schedule, solver, resolution, generation horizon, and downstream pipeline. Moreover, each method-specific parameters and interface adaptation is provided in the Appendix B.

Metrics. Efficiency is end-to-end latency and accelerator energy per video, with speedup and energy saving computed against full generation on the same model, benchmark and hardware. Quality uses standard metrics: VBench-I2V (I2V, Total), VBench T2V (Quality, Total), PAI-Bench-G (Quality, Domain), SANA-Bench ($\text{CamMC}_{\text{rel}}$, revisit PSNR/SSIM/LPIPS), and DROID-100, RealEstate10K and Sekai (FVD, paired PSNR/SSIM/LPIPS). More details are in Appendix A.4.

Table 2: **Efficiency across four GPU platforms.** For each device, we report energy per video (with its change relative to BASELINE on the same device) and end-to-end speedup over BASELINE. OOM: out of memory. Methods are compared within a device, not across devices. Among accelerated methods, the best result in each column is in **bold** and the second best is underlined.

Method	NVIDIA AGX Thor		GeForce RTX 3090Ti		NVIDIA RTX A6000		NVIDIA A100 PCIe 40 GB	
	Energy (Wh/video) ↓	E2E speedup ↑	Energy (Wh/video) ↓	E2E speedup ↑	Energy (Wh/video) ↓	E2E speedup ↑	Energy (Wh/video) ↓	E2E speedup ↑
SANA-WM								
Exact	209.8 (0.0%)	1.0×	203.7 (0.0%)	1.0×	261.9 (0.0%)	1.0×	143.0 (0.0%)	1.0×
+ EasyCache	131.0 (−37.6%)	1.6×	*OOM*	*OOM*	166.1 (−36.6%)	1.6×	*OOM*	*OOM*
+ WorldCache	89.4 (−57.4%)	2.2×	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*	*OOM*
+ FlowCache	159.5 (−24.0%)	1.3×	152.1 (−25.3%)	1.3×	193.4 (−26.1%)	1.3×	105.9 (−25.9%)	1.4×
+ NaviCache	112.1 (−46.6%)	1.9×	137.7 (−32.4%)	1.5×	227.9 (−13.0%)	1.1×	89.2 (−37.6%)	1.6×
+ DARC (Ours)	**86.1** (−59.0%)	**2.4**×	**78.2** (−61.6%)	**2.6**×	**95.6** (−63.5%)	**2.7**×	**53.9** (−62.3%)	**2.7**×
Cosmos-Predict2.5-2B								
Exact	93.3 (0.0%)	1.0×	90.1 (0.0%)	1.0×	116.1 (0.0%)	1.0×	64.2 (0.0%)	1.0×
+ EasyCache	91.8 (−1.7%)	1.0×	86.9 (−3.6%)	1.0×	114.1 (−1.7%)	1.0×	59.7 (−7.0%)	1.1×
+ WorldCache	38.8 (−58.5%)	2.3×	45.1 (−50.0%)	2.0×	58.3 (−49.8%)	1.9×	31.8 (−50.4%)	2.0×
+ FlowCache	84.0 (−10.0%)	1.1×	81.4 (−9.7%)	1.1×	106.1 (−8.5%)	1.1×	56.6 (−11.8%)	1.2×
+ NaviCache	93.3 (0.0%)	1.0×	93.0 (+2.0%)	1.0×	117.7 (+1.4%)	1.0×	68.0 (+6.0%)	1.0×
+ DARC (Ours)	**36.0** (−61.5%)	**2.5**×	**32.2** (−64.3%)	**2.9**×	**43.8** (−62.2%)	**2.6**×	**20.5** (−68.1%)	**3.2**×
MiniWorld-1B								
Exact	2.1 (0.0%)	1.0×	12.5 (0.0%)	1.0×	16.6 (0.0%)	1.0×	8.4 (0.0%)	1.0×
+ EasyCache	1.0 (−53.0%)	2.0×	5.5 (−56.1%)	2.4×	7.9 (−52.4%)	2.1×	3.1 (−63.4%)	**2.7**×
+ WorldCache	0.8 (−62.1%)	2.4×	5.9 (−52.5%)	2.0×	6.4 (−61.7%)	2.5×	5.5 (−34.2%)	1.5×
+ FlowCache	1.7 (−19.3%)	1.3×	10.2 (−18.2%)	1.4×	15.9 (−4.3%)	1.0×	4.5 (−46.1%)	1.8×
+ NaviCache	0.7 (−68.2%)	2.9×	5.3 (−41.6%)	2.1×	9.9 (−40.4%)	1.7×	5.3 (−37.1%)	1.7×
+ DARC (Ours)	**0.6** (−69.5%)	**3.0**×	**3.7** (−70.2%)	**3.2**×	**4.4** (−73.5%)	**3.6**×	**3.0** (−63.6%)	**2.7**×
AstraWorld								
Exact	14.0 (0.0%)	1.0×	14.2 (0.0%)	1.0×	19.1 (0.0%)	1.0×	9.5 (0.0%)	1.0×
+ EasyCache	8.2 (−41.4%)	1.7×	7.5 (−47.4%)	2.3×	11.4 (−40.3%)	1.7×	2.8 (−70.6%)	3.6×
+ WorldCache	6.8 (−51.1%)	2.0×	6.4 (−55.1%)	2.2×	8.6 (−55.2%)	1.9×	3.7 (−60.9%)	2.6×
+ FlowCache	10.9 (−22.3%)	1.3×	11.4 (−19.6%)	1.2×	15.0 (−21.7%)	1.3×	8.4 (−11.7%)	1.1×
+ NaviCache	6.5 (−53.8%)	2.1×	8.1 (−43.2%)	1.8×	12.6 (−34.4%)	1.5×	5.2 (−45.4%)	1.8×
+ DARC (Ours)	**5.0** (−64.6%)	**2.8**×	**5.0** (−65.1%)	**3.1**×	**7.2** (−62.6%)	**2.6**×	**2.8** (−71.0%)	**3.8**×

4.1 MAIN RESULTS

Table 1 reports quality and H200 efficiency for four world models on seven benchmarks.

Efficiency. DARC is the fastest and most energy-efficient method on every benchmark. It achieves 2.26-3.35× end-to-end speedup and 32-65% lower energy per video than EXACT, while the fastest prior cache (WorldCache) on each benchmark reaches 1.61-2.25×. Moreover, for MiniWorld and Astra, where no prior cache exceeds 1.8×, DARC reaches a gain of about 2.26-2.47×.

Quality. DARC's VBench and PAI-Bench-G scores stay within 0.7% of EXACT, and its FVD rises by at most 1.1% on DROID-100 and RealEstate10K. On SANA-Bench it runs 2.29× faster and still gives the best camera accuracy ($\text{CamMC}_{\text{rel}}$ -0.020) and the largest revisit gains of all methods.

Trade-off against prior caches. Prior caches that approach DARC's speed often lose more quality. On Cosmos, WorldCache lowers VBench Quality by 0.45 at 2.23×, against 0.35 for DARC at 2.77×. On DROID-100, EasyCache raises FVD by 9.09 at 1.77×, against 0.56 for DARC at 2.26×. However, conservative methods such as FlowCache and NaviCache never exceed 1.34×.

4.2 CROSS-HARDWARE EFFICIENCY

A training-free accelerator is useful only if its savings withstand the transition from a data-center GPU to the hardware on which world models are deployed. Table 2 evaluates four models on edge (AGX Thor), consumer (RTX 3090Ti), workstation (RTX A6000) and data-center (A100) hardware.

DARC is the most efficient method on every device. DARC is the most energy-efficient method in all 16 model–device settings and the fastest in 15. It provides 2.4-3.8× speedup and cuts energy

Table 3: **Cumulative ablation of DARC on SANA-WM.** Components are added one at a time at a matched budget of ∼21.5 model calls.

	DARC components				Work	Comparison with SANA-WM			
Variant	Output transport	Online calibration	Defect routing	Geometry risk	Fresh evals. ↓	CamMC error ↓	PSNR (dB) ↑	SSIM ↑	LPIPS ↓
SANA-WM	–	–	–	–	60.0	reference			
Periodic hold	✓	–	–	–	21.6	0.021	19.52	0.698	0.254
+ State-conditioned secant	✓	–	–	–	21.6	0.015	20.14	0.768	0.189
+ Online calibrated blend	✓	✓	–	–	21.6	0.014	22.15	0.778	0.168
+ Accumulated defect exposure	✓	✓	✓	–	21.5	0.013	22.67	0.789	0.168
Full DARC	✓	✓	✓	✓	21.5	**0.012**	**23.18**	**0.792**	**0.162**

by 5974%, including 2.4-3.0× and 5970% on the edge Thor. Its energy savings closely follow its speedups (e.g., 3.6× and −73.5% for MiniWorld on the A6000).

Prior caches do not transfer reliably across hardware. No prior method is consistently strongest: on MiniWorld, the best baseline is NaviCache on Thor, EasyCache on the RTX 3090Ti and A100, and WorldCache on the A6000. Gains can also collapse, as WorldCache falls from 2.5× on the A6000 to 1.5× on the A100, or fail outright: WorldCache (A6000, A100) and EasyCache (A100) run out of memory for SANA-WM, and NaviCache gives no speedup on Cosmos on any device. DARC runs in every setting, matching EasyCache's speed on MiniWorld on the A100 (2.7×) at lower energy.

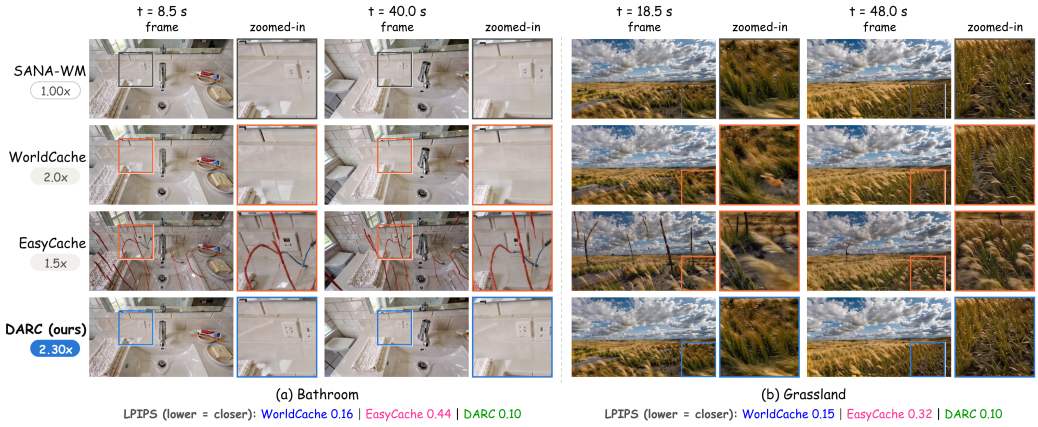


Figure 4: **Qualitative comparison.** WorldCache loses fine detail (the wall outlet in (a)), and EasyCache adds artifacts (red streaks in (a), bare trees in (b)). DARC stays closest to the unaccelerated reference while running fastest (LPIPS 0.10 in both scenes).

4.3 Ablation Study

Table 3 adds DARC's components one at a time on SANA-WM at a fixed budget of about 21.5 of 60 calls, so every gain comes from using the same calls better. In **(C1)**, the complete-output secant cuts LPIPS from 0.254 to 0.189 and calibration adds 2.0 dB, while **(C2)** adds 1.0 dB more and is best on all four metrics. Overall, DARC improves PSNR by 3.7 dB and cuts LPIPS by 36% over the periodic hold, with no metric degrading at any stage. Appendix G further covers budget and span sensitivity (Table 8), sampler length and horizon (Table 9), speed-matched baselines (Table 5) and routing-signal alignment (Table 7). DARC also extends to distilled samplers: on four-step SANA-WM it preserves the exact output rather than guessing, and its training-free companion DARC-FS applies the same predict → bound → fall-back principle to chunks and refiner calls, giving 1.45× end-to-end and 35.9% energy savings on Jetson AGX Thor (Appendix H).

5 Conclusion

We introduced DARC, a training-free controller that accelerates video world models by predicting model outputs in place of selected evaluations, leaving the model and solver unchanged. Each fresh evaluation serves twice, as an anchor for prediction and as feedback that calibrates the predictor and decides when to reanchor. Across four models, seven benchmarks and five GPUs, DARC achieves up to 3.8× speedup and 74% lower energy with near-reference quality.

AI USE STATEMENT

In this work, We used generative AI tools to polish language and improve clarity in the writing of this manuscript, including refining phrasing, tightening and restructuring prose, shortening captions, and formatting LaTeX tables. The research ideas, method, experiments, results, and their analysis are the authors' own. All AI-assisted text was reviewed and edited by the authors for accuracy and to ensure it faithfully represents our work. We also take full responsibility for the final content of this paper, including all text, claims and artifacts.

ETHICS STATEMENT

This work proposes a training-free method that reduces the computation needed to sample from existing video world models. It involves no human subjects, collects no new data and releases no new datasets or model weights. All experiments use publicly released models (SANA-WM, Cosmos-Predict2.5, MiniWorld-1B, Astra) and public benchmarks (SANA-Bench, VBench, PAI-Bench-G, DROID, RealEstate10K, Sekai) under their respective licenses. DARC does not change model weights, training data or capabilities, so it neither introduces nor removes biases present in the underlying models. It does lower the cost of generating video, which benefits deployment on energy-constrained devices such as robots.

REPRODUCIBILITY STATEMENT

DARC is training-free and uses released model checkpoints and public benchmarks. Sec. 3 defines the method, and Appendix J gives every definition, clipping rule, numerical floor, lifecycle safeguard and per-model interface detail (transported tensor, adapter G, coefficient expansion $\mathcal{E}_G$ and timestep handling), together with the complete execution order in Algorithm 1. Table 4 reports the per-setting budget and span cap together with all shared constants, and Appendix C describes how they were chosen. Appendix A specifies the models, benchmarks, samples, seeds, resolutions, samplers, hardware, and timing, energy and metric protocols. Appendix B gives the released and speed-matched settings of every baseline, and Appendix H states all constants of the few-step companion. We will release the code, evaluation manifests, and evaluation scripts.

REFERENCES

Arslan Ali et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Leyang Chen, Junyi Wu, Shaoqiu Zhang, and Yulun Zhang. Worlddyncache: Risk-controlled latent dynamics approximation for diffusion world model. *arXiv preprint arXiv:2608.01845*, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024.

Ziqi Huang et al. Vbench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Yanchao Li et al. Disagree to accelerate: Closing the loop on diffusion feature forecasts. *arXiv preprint arXiv:2608.01740*, 2026a.

Zhen Li et al. Sekai: A video dataset towards world exploration. *Advances in Neural Information Processing Systems*, 2026b.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

Chengtao Lv, Yumeng Shi, Yushi Huang, Ruihao Gong, Shen Ren, and Wenya Wang. Light forcing: Accelerating autoregressive video diffusion via sparse attention. In *ICML*, 2026a.

Zheqi Lv et al. Navicache: Test-time self-calibration caching for video generation. In *ICML*, 2026b.

Yuexiao Ma et al. Flow caching for autoregressive video generation. In *ICLR*, 2026.

Umair Nawaz, Ahmed Heakl, Ufaq Khan, Abdelrahman Shaker, Salman Khan, and Fahad Shahbaz Khan. Worldcache: Content-aware caching for accelerated video world models. In *European Conference on Computer Vision*, 2026.

Zichao Yu et al. Ab-cache: Training-free acceleration of diffusion models via adams-bashforth cached feature reuse. In *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025.

Yian Zhao, Ruochong Zheng, Hongcan Guo, Yu Yan, Jian Zhang, and Jie Chen. Miniworld: Democratizing the training of video world models from scratch. *arXiv preprint arXiv:2608.01127*, 2026.

Fengzhe Zhou, Jiannan Huang, Jialuo Li, Deva Ramanan, and Humphrey Shi. Pai-bench: A comprehensive benchmark for physical ai. In *CVPR*, 2026.

Xin Zhou et al. Less is enough: Training-free video diffusion acceleration via runtime-adaptive caching. *arXiv preprint arXiv:2507.02860*, 2025.

Haoyi Zhu et al. Sana-wm: Efficient minute-scale world modeling with hybrid linear diffusion transformer. *arXiv preprint arXiv:2605.15178*, 2026a.

Yixuan Zhu, Feng Jiaqi, Wenzhao Zheng, Yuan Gao, Xin Tao, Pengfei Wan, Jiwen Lu, and Jie Zhou. Astra: General interactive world model with autoregressive denoising. In *ICLR*, 2026b.

APPENDIX

Accelerating Video World Models at the Edge via Defect-Adaptive Reanchoring

§A	Experimental protocol: models, benchmarks, settings, hardware, timing, energy and metrics
§B	Baseline configurations and speed-matched variants
§C	DARC hyperparameters, shared constants and tuning protocol
§D	Speed-quality trade-offs at matched speed
§E	Statistical reliability and evaluation subsets
§F	Routing-signal alignment under each method's own target and a common target
§G	Additional ablations: sensitivity, safeguards and operating regimes
§H	Few-step and distilled samplers
§I	Discussion and limitations
§J	DARC: complete definitions and per-model implementation
§K	Additional qualitative results
§L	DARC Pseudo algorithm

Conventions. EXACT denotes the unaccelerated model with its native sampler executed in vanilla mode. Every change, speedup and saving in this appendix is relative to EXACT on the same model, samples and hardware. Fresh NFE counts complete model evaluations (Sec. A). Appendix tables report SSIM, LPIPS and $\text{CamMC}_{\text{rel}}$ on their native 0–1 scale, whereas Table 1 multiplies them by 100.

A EXPERIMENTAL PROTOCOL

A.1 MODELS AND BENCHMARKS

We evaluate four world models that differ in architecture, conditioning and generation mode, each on the benchmarks that match its intended use (Table 1).

SANA-WM (Zhu et al., 2026a) is a camera-controlled video world model from the SANA diffusion-transformer family. It generates minute-long videos in two stages: a 60-step Stage-1 DiT sampler followed by a refiner. *SANA-Bench* evaluates whether generated videos follow prescribed camera paths and stay consistent when the camera returns to a previously seen pose. We use all 160 videos (80 simple, 80 hard), each with 961 frames at 16 fps and 1280×704, CFG $w{=}5$ and seed 42. *VBench-I2V* is the standard image-to-video benchmark, which scores visual quality and consistency with the conditioning image. We compute all nine official dimensions with the official weights: Video-Image Subject Consistency, Video-Image Background Consistency and Camera Motion for the I2V score, and Subject Consistency, Background Consistency, Motion Smoothness, Dynamic Degree, Aesthetic Quality and Imaging Quality for the Quality score (Total $=$ (Quality $+$ I2V)/2, camera-motion weight 0.1).

Cosmos-Predict2.5-2B (Ali et al., 2025) is a 2B-parameter world foundation model for physical AI that generates future video from text, image or video conditioning with a flow-based sampler. *VBench (T2V)* is the standard text-to-video benchmark, which scores visual quality and semantic alignment with the prompt. We use the official benchmark samples, 93 frames at 704×1280, 36 realized FlowUniPC steps and guidance 7. *PAI-Bench-G* (Image2World) evaluates world generation in physical-AI domains such as robotics and driving and reports a general quality score and a domain score. We use all 1,044 samples, 93 frames at 768×432, seed 0 and guidance 7.

MiniWorld-1B (Zhao et al., 2026) is a 1B-parameter autoregressive video world model that generates long videos chunk by chunk with streaming diffusion forcing, in which one denoising window slides over the chunks being generated. We evaluate it on two real-world video datasets: *DROID-100*, which contains robot-manipulation videos, and *RealEstate10K*, which contains camera walkthroughs

of indoor and outdoor scenes. We generate 253 frames per video (64 chunks) with 100 outer steps (8 in-flight chunks, one sink chunk), CFG 2.0 and initial noise shared across methods.

Astra (Zhu et al., 2026b) is an autoregressive, camera-controllable world model that generates video in sliding windows, each denoised as an independent trajectory. We evaluate it on *Sekai*, a world-exploration video dataset with camera annotations, using 36 game-drone clips (four from each of nine games, covering all 15 source videos). Each clip is conditioned on its first frame and real camera poses and generated in three windows of 8 latents with 50 steps each, giving 96 frames at 832×480 and 15 fps. These clips come from Astra's training distribution, so we use them to measure efficiency and fidelity relative to EXACT, not generalization.

A.2 SHARED GENERATION SETTINGS

Within a benchmark, all methods share the checkpoint, conditioning, samples, seeds, timestep schedule, solver, resolution, horizon and post-processing. They differ only in which denoiser evaluations are skipped. On SANA-Bench, every method, including EXACT and DARC, uses the same three-call exact refiner.

A.3 HARDWARE, TIMING AND ENERGY

Devices. We measure efficiency on five platforms that span the settings in which world models are deployed: a data-center H200 NVL (141 GB), which also produces all quality results in Table 1; a data-center A100 PCIe (40 GB); a workstation RTX A6000 (48 GB); a consumer system with two GeForce RTX 3090 Ti GPUs (24 GB each); and the Jetson AGX Thor, a power-constrained edge module for robotics and embodied AI.

Timing. We report end-to-end generation time per video, from ingesting the conditioning inputs to the final latent video, excluding model loading and VAE decoding, which no method changes.

Energy. Energy is the accelerator energy consumed to generate one video. We sample board power with `nvidia-smi` at 10 Hz (module input power via `tegrastats` at 1 Hz on Thor), integrate it over each video's generation window, and report Wh per video without subtracting idle power, so the numbers reflect what a deployment actually draws. Workloads, memory offloading and telemetry boundaries differ between devices, so only ratios within a device are meaningful. When average power is unchanged, the saving equals $1 - 1/\text{speedup}$. Energy therefore mainly verifies that time savings are not offset by a higher power draw, and we do not read differences of less than one percentage point between methods as a ranking.

Work accounting. Fresh NFE counts complete DiT evaluations, the unit of work that caching removes. A classifier-free-guidance pair counts as one guided evaluation on SANA-WM and Cosmos, and partial block execution (WorldCache probes) is counted fractionally.

A.4 METRICS

Arrows in the tables give the direction of improvement.

Camera control. $\text{CamMC}_{\text{rel}}$ (↓) measures how closely the generated video follows the conditioning camera path. We estimate camera poses from every fourth generated frame with Pi3 and compute their trajectory error relative to the prescribed path.

Revisit consistency. Revisit PSNR/SSIM (↑) and LPIPS (↓) compare the frames generated when the camera returns to a previously seen pose, using the five same-pose pairs per scene that SANA-Bench prescribes within a 10-second window. Two frames also agree more when both lose fine detail, so we read revisit scores together with paired fidelity (Sec. D).

Paired fidelity. Paired PSNR/SSIM (↑) and LPIPS (↓) compare each generated frame with a reference frame at the same time step, excluding the conditioning frame 0. In Table 1, the reference is the ground-truth video (DROID-100, RealEstate10K and Sekai). In the ablations and trade-off analyses, it is the EXACT output of the same sample, which isolates the effect of caching.

Table 4: **DARC hyperparameters.** Only the budget ε and the span cap m are set per setting; the number of native nodes N and the solver view G are fixed by each model's sampler. The lower block lists the constants shared by all settings.

	SANA-WM SANA-Bench	SANA-WM VBench-I2V	Cosmos VBench-T2V	Cosmos PAI-Bench-G	MiniWorld DROID / RE10K	Astra Sekai
Budget ε	0.012	0.024	0.006	0.012	0.024	0.006
Span cap m	5	8	5	5	5	5
Native nodes N	60	60	36	36	100 (outer)	50 per window
Solver view G	CFG (w=5)	CFG (w=5)	post-CFG (w=7)	post-CFG (w=7)	post-CFG (w=2)	no CFG

Shared by all settings
Signature length p=8192; exposure weights $w_{\beta q}$=w_{ch}=w_q=1; gap cap $q_{\max}$=1; predictor: calibrated blend.
Coordinate clip $[\alpha_{\min}, \alpha_{\max}]$=$[-0.25,\ 2]$; T_{norm}: span of the native timestep scale (1000 for timesteps on a 0–1000 scale).
Numerical floors: η_r=10^{-6}; η_t=η_x=η_y=10^{-8}; trajectory-reset tolerance 10^{-4}.

Distribution-level quality. FVD (↓) compares the feature distributions of generated and real videos, capturing realism and temporal coherence rather than frame-by-frame agreement. We compute it on 16-frame clips with StyleGAN-V I3D features.

Perceptual quality. VBench scores (↑) aggregate dimensions such as consistency, motion smoothness, aesthetic and imaging quality (and, for image-to-video, agreement with the conditioning image), computed with the released implementation and official weights.

B Baseline Configurations

Released settings. All baselines use their released defaults, without per-benchmark tuning: EasyCache τ=0.10 on SANA-WM and 0.05 elsewhere (3–5 warm-up steps); WorldCache base threshold 0.08, retention 0.2, probe depth 2; FlowCache relative-L1 threshold 0.010 with 5 warm-up steps; NaviCache τ=0.05, N_{align}=20% of steps, Q=R=0.05, P_0=1. Every number in Tables 1 and 2 uses these settings.

Speed-matched variants. To separate the quality of a routing rule from the amount of work it skips, we also raise each baseline's threshold until its fresh-NFE count matches DARC's (≈19 of 60 on SANA-WM): EasyCache 0.35, NaviCache 0.40, WorldCache 0.20 and FlowCache 0.080. These variants appear only in Sec. D.

C DARC Hyperparameters and Tuning Protocol

Shared constants. The constants in the lower block of Table 4 were fixed once and reused unchanged for every model and benchmark. The three exposure weights and $q_{\max}$ are set to one, so the geometry terms enter the exposure score (Eq. (25)) unweighted. The coordinate clip bounds how far the secant may extrapolate, and the floors only prevent division by zero.

Parameter Selection. All controller weights, clipping limits, and floors in Table 4 were fixed once and reused for every model and benchmark. Only ε and m were chosen per setting, from small sweeps as they are the key parameters that drive our method. The sweep over $\varepsilon \in \{0.003, 0.006, 0.012\}$ and $m \in \{3, 5\}$ is provided in Table 5), VBench-I2V ($\varepsilon \in \{0.012, 0.024, 0.048\}$, $m \in \{5, 8\}$), MiniWorld ($\varepsilon \in \{0.006, 0.012, 0.024\}$) and PAI-Bench-G ($\varepsilon$=0.006 and 0.012).

D Speed-Quality Trade-offs at Matched Speed

The official thresholds of each method leave each baseline at a different speed, so they cannot show whether a method's quality comes from a better routing rule or from doing more work. Table 5 therefore sweeps DARC's budget/span and adds a threshold that matches DARC's work (18.8-19.9 fresh evaluations against DARC's 18.4). All rows use the same samples we chose for SANA-Bench ablations.

Paired fidelity. At matched speed, NaviCache and EasyCache stay closest to Exact (paired LPIPS 0.248 and 0.241), DARC and WorldCache are tied (0.242 and 0.243), and FlowCache falls far

Table 5: **Speed-quality trade-off on SANA-Bench (SANA-WM, H200).** Speedup is end-to-end. Paired LPIPS is the distance from EXACT on the same scene and seed. ΔRevisit is the change in the benchmark's revisit PSNR relative to EXACT on the same scenes. The matched-speed block raises each baseline's threshold until its work matches DARC's.

Method	Setting	Speedup ↑	Fresh NFE ↓	Paired LPIPS ↓	ΔRevisit PSNR (dB)
EXACT	60 steps	1.00×	60.0	ref.	+0.00
DARC budget and span sweep					
DARC	ε=0.003, m=5	1.71×	28.9	0.186	+0.36
DARC	ε=0.006, m=3	1.81×	26.5	0.192	+0.29
DARC	ε=0.012, m=3	1.92×	24.1	0.206	+0.34
DARC	ε=0.006, m=5	2.04×	21.8	0.215	+0.57
DARC	ε=0.012, m=5	2.27×	18.4	0.242	+0.56
Baselines at released thresholds					
EasyCache	τ=0.10	1.46×	35.9	0.076	+0.04
WorldCache	τ=0.08	1.98×	23.3	0.193	+0.15
FlowCache	τ=0.010	1.22×	44.5	0.159	+0.31
Baselines at DARC's speed (matched thresholds)					
EasyCache	τ=0.35	2.12×	19.6	0.241	+0.32
NaviCache	τ=0.40	2.10×	19.9	0.248	+0.22
WorldCache	τ=0.20	2.19×	19.5	0.243	+0.24
FlowCache	τ=0.080	2.25×	18.8	0.470	−0.11

Table 6: **Uncertainty of SANA-Bench quality changes.** Mean change relative to EXACT over the 160 videos, with paired-bootstrap 95% confidence intervals (10,000 resamples). Revisit PSNR and LPIPS improve significantly for every method shown, and DARC's PSNR and SSIM gains are the largest, with intervals that do not overlap those of the other methods.

Method	$\Delta\text{CamMC}_{\text{rel}}$	ΔRevisit PSNR (dB)	ΔRevisit SSIM	ΔRevisit LPIPS
EasyCache	−0.013 [−0.039, +0.011]	+0.05 [+0.04, +0.07]	+0.0008 [−0.0001, +0.0016]	−0.0012 [−0.0019, −0.0004]
WorldCache	+0.010 [−0.034, +0.052]	+0.19 [+0.15, +0.24]	+0.0007 [−0.0015, +0.0029]	−0.0033 [−0.0052, −0.0013]
FlowCache	−0.011 [−0.082, +0.050]	+0.20 [+0.17, +0.23]	+0.0023 [+0.0008, +0.0038]	−0.0021 [−0.0036, −0.0005]
DARC	−0.020 [−0.123, +0.059]	+0.44 [+0.37, +0.50]	+0.0080 [+0.0046, +0.0111]	−0.0053 [−0.0084, −0.0021]

behind (0.470, up from 0.159 at its released threshold). Tightening the budget moves DARC towards the reference (0.186 at ε=0.003, 1.71×). Residual reuse keeps each rollout close to the reference trajectory, whereas a transported complete output changes the state that every later call sees, so small differences compound over a 60-second rollout (paired LPIPS 0.054 after one chunk and 0.168 after eight; Table 9).

Revisit consistency. DARC has the largest revisit gain at matched speed (+0.56 dB, against +0.32 for EasyCache, +0.24 for WorldCache, +0.22 for NaviCache and −0.11 for FlowCache). Revisit gains tend to grow as a method moves away from EXACT: raising EasyCache's or WorldCache's threshold increases both, and so does loosening DARC's budget (+0.36 at ε=0.003, +0.56 at 0.012). Part of what revisits PSNR rewards may therefore be reduced fine detail, which makes frames agree more. We use it as a benchmark-level consistency score and interpret it only together with paired fidelity.

E STATISTICAL RELIABILITY

Confidence intervals. DARC's revisit gain over EXACT is not noise: +0.44 dB with a 95% interval of [+0.37, +0.50], above the interval of every other method in Table 6. As discussed in Sec. D, this establishes a benchmark-level effect, not its cause.

Scope. Confidence intervals are available for SANA-Bench only; the other benchmarks in Table 1 report point estimates. FVD estimated from 36 (Sekai) or 100 (DROID-100) videos has high sampling variance, so we do not rank methods by FVD differences of a few percent.

Evaluation subsets for cross-hardware results. The efficiency measurements of Table 2 use a fixed subset of 50 samples per benchmark, model and device, and every method, including EXACT, runs on exactly the same subset. Speedups and energy savings are therefore computed on identical workloads within each device.

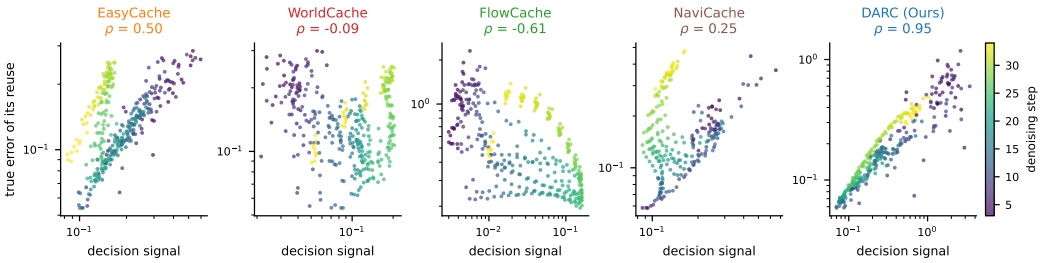


Figure 5: **Alignment between routing signals and each method's own reuse error (Cosmos-Predict2.5-2B, 10 prompts).** Fully evaluated observers for EasyCache, WorldCache, FlowCache, NaviCache and DARC pair each method's routing signal, available before evaluation, with the error of the output that the method would have reused or predicted, measured against a fresh output at the same reached state. Color denotes the denoising node; each panel pools about 320-350 node-prompt observations, and ρ is the pooled Spearman correlation. Table 7 repeats the analysis under a common target.

F Routing-Signal Alignment

Fig. 5 correlates each method's routing signal with the error of that method's own reuse decisions. It is the quantity each signal is designed to predict. This is the analysis behind the main-paper value ρ=0.95. Pooled correlations of this kind have three weaknesses: a shared trend across denoising steps can inflate them, each method is scored against a different target, and DARC's signal is built from past errors of the same kind of forecast it is scored against. We therefore add a passive observer on SANA-WM that evaluates every node exactly and scores all routing signals against one common target, the same-state error of the complete-output forecast (6 SANA-Bench videos, 282 node observations).

Table 7: **Routing alignment under a common target** (SANA-WM observer, 6 videos, 282 node observations). Within-step ρ uses ranks normalized within each node. Step-partialled ρ correlates rank residuals after regressing both ranks on the node index.

Routing signal	Pooled ρ	Step-partialled ρ	Within-step ρ
DARC	+0.61	+0.88	+0.68
EasyCache	+0.36	+0.35	+0.34
WorldCache	+0.03	+0.10	−0.00
FlowCache	−0.09	+0.19	+0.10

Under the common target, DARC's pooled correlation is 0.61, which we regard as the conservative estimate. The denoising step alone is only weakly related to the target (Spearman −0.19), and after the step trend is removed, DARC's signal remains strongly aligned (step-partialled ρ=0.88, within-step ρ=0.68), EasyCache's remains moderate (0.35, 0.34), and WorldCache's and FlowCache's stay near zero (Table 7). The ordering of Fig. 5 is therefore preserved under a shared target and within individual steps. However, two caveats still remain. The common target is the error that a complete-output cache must control, so it naturally suits a signal built around it, which is why we report it alongside, rather than instead of, the own-error analysis. In addition, both analyses measure local same-state error, not the final deviation of the video from Exact.

G Additional Ablations

G.1 Set-up

Unless stated otherwise, the ablations use five SANA-Bench scenes on H200, the ablation default ($\varepsilon = 0.006$ and $m = 5$, i.e. 21.6 of 60 fresh Stage-1 evaluations) and the three-call exact refiner. They report paired PSNR/SSIM/LPIPS to the Exact output of the same scene, and a VBench quality score on the same five scenes. Speedups are Stage-1 ratios against Exact measured in the same run.

Table 8: **Sensitivity and safeguards (five scenes).** One setting is changed at a time from the ablation default (ε=0.006, m=5). Speedup is Stage-1; PSNR and LPIPS are paired to EXACT.

Variant	Fresh NFE	Speedup	VBench ↑	PSNR ↑	LPIPS ↓	ΔLPIPS vs. default [95% CI]
$\varepsilon = 0.003$	28.4	2.11×	79.74	23.70	0.1289	-0.0386 [-0.0695, -0.0136]
$\varepsilon = 0.006$ (default)	21.6	2.77×	79.53	22.18	0.1675	—
$\varepsilon = 0.009$	19.4	3.08×	79.47	21.90	0.1736	$+0.0061$ [$+0.0009$, $+0.0143$]
$\varepsilon = 0.012$	18.2	3.28×	79.38	21.46	0.1830	$+0.0155$ [$+0.0018$, $+0.0336$]
$m = 2$	34.0	1.76×	79.77	26.83	0.0827	-0.0848 [-0.1050, -0.0615]
$m = 3$	26.2	2.28×	79.60	23.47	0.1352	-0.0323 [-0.0503, -0.0116]
$m = 5$ (default)	21.6	2.77×	79.53	22.18	0.1675	—
$m = 7$	21.0	2.84×	79.54	21.53	0.1850	$+0.0175$ [$+0.0027$, $+0.0326$]
$m = \infty$	21.0	2.84×	79.56	21.17	0.1940	$+0.0265$ [$+0.0027$, $+0.0555$]
$p = 2048$	21.4	2.79×	79.54	22.24	0.1667	-0.0008 [-0.0020, $+0.0004$]
$p = 8192$ (default)	21.6	2.77×	79.53	22.18	0.1675	—
$p = 32768$	21.6	2.77×	79.55	22.16	0.1681	$+0.0006$ [$+0.0000$, $+0.0012$]
remove r_{ch}	20.4	2.93×	79.44	21.71	0.1764	$+0.0089$ [$+0.0003$, $+0.0180$]
remove r_{ch} and q_k	20.4	2.93×	79.45	21.49	0.1812	$+0.0137$ [$+0.0003$, $+0.0319$]
keep both	21.6	2.77×	79.53	22.18	0.1675	—

We do not compare absolute times with Table 1. Differences carry paired-bootstrap 95% intervals over the five scenes, which indicate the direction of an effect rather than its precise size.

G.2 SENSITIVITY AND SAFEGUARDS

The budget and the span cap set the operating point. Both move the operating point monotonically, as tightening either buys fidelity with fresh evaluations, and loosening either buys speed (Table 8). The span cap is the stronger of the two (paired LPIPS 0.083 at m=2 and 0.194 at m=∞). Fresh NFE saturates as the budget grows (28.4, 21.6, 19.4 and 18.2 for ε=0.003 to 0.012) because the span cap and the protected calls bound the skip pattern, so larger budgets buy little additional speed.

The cap and the exposure score are complementary. At m=7 and m=∞, DARC makes the same 21.0 fresh evaluations, yet the cap lowers LPIPS by 0.009. It does not add more computation, but instead it moves fresh evaluations into long transported stretches that the exposure score alone would let continue. The exposure score is therefore not sufficient on its own, and the cap bounds how long any prediction runs without measured feedback.

Components with little or no effect. The signature length p change LPIPS by less than 0.001. Of the two geometry terms in the exposure score, removing the chord residual saves 1.2 fresh evaluations and costs $+0.009$ LPIPS, a trade-off similar to raising the budget from 0.006 to 0.009 (2.2 fewer evaluations, $+0.006$). Both terms can only add fresh evaluations, never remove them, so we keep them as inexpensive safeguards.

G.3 OPERATING REGIMES

Sampler length. DARC's benefit grows with the number of native nodes (Table 9): 1.05× at 12 steps, 1.99× at 36 and 2.77× at 60. At 4 and 8 steps, the protected calls (two bootstrap anchors, the calibration call, and the terminal call) take up half or all of the trajectory, and no remaining node passes the transport rule, so DARC returns EXACT's output with no speedup and no quality loss. Sec. H analyses this regime.

Horizon length. Over longer autoregressive horizons the speedup is stable (2.68–2.77×), while paired fidelity to EXACT decreases (LPIPS 0.054 after one chunk and 0.168 after eight), because transported outputs perturb the states on which later chunks are conditioned. Over long rollouts, paired fidelity therefore measures accumulated trajectory divergence as well as per-step caching error.

Table 9: **Operating regimes (five scenes).** Each row is compared with EXACT *at the same setting*. At 4 and 8 sampler steps, DARC evaluates every node fresh and returns EXACT's output. Lower block: the 60-step controller over increasing autoregressive horizons.

Setting	Fresh NFE	Speedup	SSIM ↑	LPIPS ↓
4 steps	4.0	1.00×	1.0000	0.0000
8 steps	8.0	1.00×	1.0000	0.0000
12 steps	11.4	1.05×	0.9557	0.0399
36 steps	18.0	1.99×	0.8225	0.1573
60 steps (default)	21.6	2.77×	0.7798	0.1675
1 chunk (121 frames)	21.6	2.68×	0.9126	0.0535
2 chunks (241)	21.6	2.73×	0.8245	0.1129
4 chunks (481)	21.6	2.76×	0.8449	0.1105
8 chunks (961, default)	21.6	2.77×	0.7798	0.1675

H FEW-STEP AND DISTILLED SAMPLERS

Step distillation is a common way to deploy world models on constrained hardware: a distilled checkpoint replaces a long denoising chain with a handful of steps. This section asks what DARC contributes in that regime. We first show that DARC, by design, stops acting on very short samplers, then locate the work that a four-step streaming pipeline still repeats, and finally evaluate a training-free companion that applies DARC's principles to that work.

DARC's boundary: protected calls leave nothing to transport. DARC reserves fresh evaluations for three purposes: *(i)* two anchors that bootstrap its predictor, *(ii)* a calibration call that measures the first pre-call forecast defect, and *(iii)* the terminal node. These protected calls are a fixed cost per trajectory, so the part of the trajectory that DARC can transport shrinks as the sampler gets shorter. With $N_{\mathrm{fresh}}(K)$ the evaluations DARC makes on a K-step sampler, the realized transport opportunity $\eta_K = 1 - N_{\mathrm{fresh}}(K)/K$ falls from 64% at K=60 and 50% at K=36 to 5% at K=12 and exactly 0% at K=8 and K=4 (Table 9). On the official distilled four-step SANA-WM checkpoint, DARC therefore evaluates the model at every node and returns the exact solver's output. In a 60-second evaluation, it made all 160 of 160 Stage-1 evaluations. Thus, transporting within a four-call trajectory would require skipping a bootstrap anchor or the terminal call, that is, predicting without measured feedback, and DARC's contract is to fall back to EXACT rather than guess.

Where a four-step pipeline still repeats work. In streaming SANA-WM, every chunk passes through three stages: four Stage-1 evaluations, a refiner with three exact calls per block, and a clean-history forward that writes the causal K/V cache for the next chunk. Once Stage-1 has only four steps, the refiner and the history forward account for much of the runtime, so reducing Stage-1 evaluations alone is Amdahl-limited: halving them yields only $1.15\times$ end-to-end (Table 10). The remaining redundancy lies between neighboring evaluations on a chunk's Stage-1 grid and between successive refiner calls.

H.1 DARC-FS: DARC'S PRINCIPLES ON THE CHUNK AND REFINER AXES

We therefore keep DARC unchanged for multi-step samplers and, when the native sampler is too short for it to act, we can opt for a training-free companion, DARC-FS, with two parts: ACE-2 for Stage-1 and AERO-CB for the refiner. Both follow the three principles of DARC: *(i)* predict an evaluation from fresh ones rather than recomputing it, *(ii)* bound the prediction with an explicit error measure, and *(iii)* fall back to an exact evaluation whenever that bound is exceeded. DARC-FS is not a distilled DARC and does not use DARC's denoising-axis transport rule. It applies the same $predict \rightarrow bound \rightarrow fall - back$ loop, where a four-step pipeline still repeats work.

ACE-2: bounded Stage-1 prediction on the native four-step grid. The distilled sampler visits four timesteps, $t = 1000 \rightarrow 960 \rightarrow 889 \rightarrow 727$, before producing the clean output at t=0. ACE-2 evaluates the model at only two of them, t=1000 and t=889, predicts the model output at the other two, and still runs every Euler update of the solver. At t=960 only one fresh output exists, so ACE-2 reuses it unchanged (a zero-order hold). At t=727 two fresh outputs exist, so ACE-2 extends the

change between them (a secant) but limits how far this extrapolation may move:

$$\widehat{v}_{727} = v_{889} + s\,\Delta v, \qquad \Delta v = \tfrac{889-727}{1000-889}\,(v_{889} - v_{1000}) \approx 1.46\,(v_{889} - v_{1000}),$$
$$s = \min\Big(1,\ \tfrac{\kappa}{\mathrm{RMS}(\Delta v)+\epsilon}\Big). \tag{5}$$

Here Δv is the change predicted by the secant, which is obtained by scaling the change between the two fresh outputs by the ratio of the timestep gaps, and s shrinks that change whenever its size exceeds a trust radius κ:

$$\kappa = \frac{(0.12 + 0.13\,c_{\mathrm{cam}})\ \mathrm{RMS}(v_{889})}{1 + 2\,\delta_x}. \tag{6}$$

In words, the prediction may move by at most 12–25% of the magnitude of the last fresh output. It is allowed the full 25% only when the camera is steady: $c_{\mathrm{cam}} = \mathrm{clip}(1 - r_c/0.85,\ 0,\ 1)$ equals 1 for a static camera and falls to 0 as the chunk's normalised camera-motion score r_c approaches the threshold 0.85. The radius shrinks further when the latent has already moved far from the last anchor, measured by $\delta_x = \mathrm{RMS}(x_{727} - x_{889})/(\mathrm{RMS}(x_{889}) + \epsilon)$. Now, three cases always stay exact: *(i)* the first chunk, which has no history to predict from, *(ii)* chunks whose camera motion exceeds the threshold ($r_c > 0.85$), and *(iii)* the clean-history forward of every chunk. A 40-chunk video therefore needs four fresh Stage-1 evaluations for its first chunk and two for each of the other 39, i.e. $(4 + 39 \cdot 2)/40 = 2.05$ per chunk on average.

AERO-CB: online refiner completion under a causal budget. The refiner makes three exact calls per block, and the third output is largely predictable from the first two. A wrong prediction, however, does not stay local as it is written into the K/V history that every later chunk reads. AERO-CB therefore predicts the third call only while the accumulated prediction error stays within a causal budget and otherwise runs the refiner exactly.

What is predicted? AERO-CB predicts the third output $p_b^{(2)}$ of block b as the second output plus a learned correction:

$$\widehat{p}_b^{(2)} = p_b^{(1)} + \phi_b\,\theta, \qquad \phi_b = \big[\,p_b^{(1)} - p_b^{(0)},\ \ z_b - p_b^{(1)},\ \ a_b^{(2)} - p_b^{(1)}\,\big], \tag{7}$$

where ϕ_b collects three differences that are available before the third call: the change between the first two outputs, and the offsets of z_b and $a_b^{(2)}$ from the second output. Here z_b is the clean input of block b, i.e. the Stage-1 latent frames that the refiner is refining, and $a_b^{(2)}$ is the refiner's latent state immediately before the third prediction, i.e. after its first two updates. The coefficients θ are fitted online per channel by ridge regression (strength 10^{-3}) on exact blocks only and clipped to $[-4, 4]$. If the implied correction exceeds 75% of the block's own update, the block falls back to the exact step generation.

When is the prediction trusted? A debt tracks how much unverified error has accumulated:

$$D_b = \lambda_D\,D_{b-1} + \ell_b\,w_b, \qquad \lambda_D = 0.75, \tag{8}$$

where ℓ_b is the completion error validated on exact blocks and w_b weights it by the remaining horizon, since an early error is inherited by more chunks, and by camera motion. Older debt decays by a factor of 0.75 per block. Once the debt exceeds the budget of 0.45, the next block runs the exact refiner, and every exact block resets the debt to zero. The debt thus plays the role of DARC's exposure as it accumulates error between exact evaluations and forces a fresh one when the budget is spent.

Safeguards. Besides the debt trigger, the exact refiner is used for the first two (warm-up) blocks at scheduled calibration anchors under high camera motion, whenever eight blocks have passed since the last exact one, and whenever the completion stops paying off. The last condition is checked at every exact block, before the fit is updated, with the validation-error ratio:

$$e_b = \frac{\|\widehat{p}_b^{(2)} - p_b^{(2)}\|_2}{\|p_b^{(2)} - p_b^{(1)}\|_2}, \tag{9}$$

which compares the completion's error with the error of simply reusing the second output. A value $e_b > 1.02$ means that the completion is no better than this reuse, so AERO-CB returns to the exact refiner. Finally, once the refiner has converged, its final prediction supplies the chunk's K/V history directly, saving the separate clean-history forward. Convergence is measured by:

$$\rho_b = \frac{\mathrm{RMS}\left(p_b^{(1)} - p_b^{(0)}\right)}{\mathrm{RMS}\left(p_b^{(1)} - z_b\right) + \epsilon}, \tag{10}$$

Table 10: **Few-step companion on the distilled four-step SANA-WM (NVIDIA AGX Thor).** *Setup.* Ten fixed 60-second scenes (five simple, five hard; 961 frames at 1280×704) with fixed seeds. Every method is compared with a fresh EXACT run of the same scene. *Rows.* DARC is our multi-step controller applied unchanged. Phase A adds ACE-2, which predicts two of the four Stage-1 evaluations per chunk. Phase B, DARC-FS, adds AERO-CB, which also predicts refiner calls. *Columns.* Fresh NFE is the number of Stage-1 model evaluations per chunk (the clean-history forward is not counted). *Full* time is the complete request, including model loading and final muxing. *Stream* time excludes both. VBench is a nine-dimension score, Revisit is PSNR / SSIM / LPIPS over five same-pose pairs, and energy is module-input power integrated over the request. The grey line under each value gives the speedup (time columns) or the change from EXACT (quality and energy columns). Speedups are geometric means of per-scene ratios, while all other values are arithmetic means.

Method	Fresh NFE / chunk ↓	Full time ↓ speedup ↑	Stream time ↓ speedup ↑	Stage-1 speedup ↑	VBench diag. quality / total ↑	Revisit P ↑ / S ↑ / L ↓	Energy Wh/video ↓
EXACT four-step	4.00	13m55s 1.00×	10m45s 1.00×	1.00×	78.90 / 0.679 +0.00 / +0.000	15.10 / 0.310 / 0.550 +0.00 / +0.000 / +0.000	16.80 +0.0%
DARC (multi-step controller)	4.00	– 1.00×	– 1.00×	1.00×	no transport within exact run-to-run variation		–
ACE-2 + exact refiner [Phase A]	2.05	12m08s 1.15×	8m56s 1.20×	1.51×	78.76 / 0.678 −0.15 / −0.001	14.74 / 0.313 / 0.552 −0.36 / +0.003 / +0.003	14.80 −11.9%
DARC-FS (ACE-2 + AERO-CB) [Phase B]	**2.05**	**9m35s** **1.45×**	**6m24s** **1.68×**	**1.52×**	**79.62 / 0.686** +0.72 / +0.007	14.48 / 0.299 / 0.584 −0.62 / −0.010 / +0.034	**10.77** −35.9%

which is the change between the first two outputs relative to the second output's distance from the clean input z_b. The shortcut is taken when $\rho_b \leq 0.42$, after the two warm-up blocks. AERO-CB uses no learned router, offline fitting, checkpoint change, or quantization.

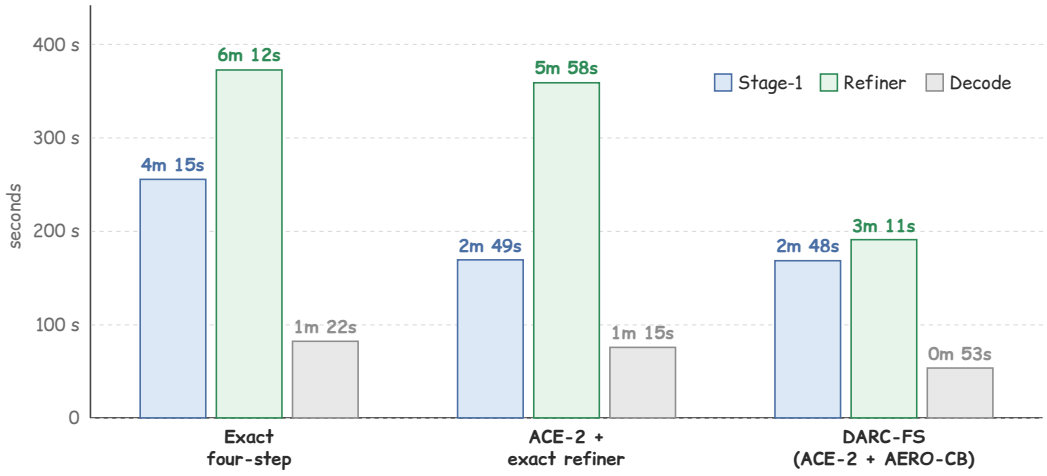


Figure 6: **Where the time goes in four-step SANA-WM on Jetson AGX Thor.** Mean time per stage and video. ACE-2 cuts Stage-1 from 4m15s to 2m49s, but the exact refiner stays the largest cost (5m58s). DARC-FS adds AERO-CB, which predicts refiner calls and nearly halves refiner time to 3m11s (1.95×), leaving Stage-1 unchanged.

Results: the refiner, not Stage-1, is the bottleneck. Table 10 adds the two parts one at a time. ACE-2 alone (Phase A) halves the fresh Stage-1 evaluations (from 4.00 to 2.05 per chunk) and makes Stage-1 1.51× faster, but the full pipeline gains only 1.15×, because the refiner and the history forward still run exactly. Figure 6 illustrates that the refiner takes more time than Stage-1. Adding AERO-CB (Phase B, DARC-FS) leaves Stage-1 unchanged yet runs the exact refiner on only 17.5% of refiner blocks on average. This halves refiner time relative to EXACT (6m12s → 3m11s, 1.95×) and raises the end-to-end speedup to 1.45× (1.68× in stream time), while cutting module energy by 35.9%. Relative to ACE-2 alone, it adds a further 1.27× speedup and a 27.3% energy saving. As the Amdahl analysis predicts, once the sampler has only four steps, efficiency is recovered by predicting across refiner calls rather than by removing more denoising evaluations.

Quality and validation. ACE-2 leaves quality nearly unchanged. The full DARC-FS slightly raises the VBench scores ($+0.72$ quality, $+0.007$ total) but lowers revisit consistency. The loss enters with the refiner completion and is concentrated in two scenes with close or repeated geometry, `indoor_009` (LPIPS $+0.151$) and `outdoor_city_020` ($+0.108$). On the hard split the largest change is $+0.037$, and two scenes improve. All constants of ACE-2 and AERO-CB were fixed on two calibration scenes before the remaining eight scenes were generated and were not changed afterwards. DARC-FS is therefore a quality-efficiency operating point for the four-step regime, not lossless acceleration.

Caching and distillation. Where a distilled checkpoint of adequate quality exists, distillation is the larger lever at the edge. On Thor, the four-step EXACT pipeline uses 16.8 Wh per video, several times less than 60-step DARC on SANA-WM in Table 2. The two answer different situations. Distillation trains a separate checkpoint for each model, whereas DARC applies to any released multi-step checkpoint without training, and DARC-FS carries the same principle into the distilled regime.

Scope. The few-step results complement DARC's main claims rather than change them. On multi-step samplers, DARC's measured-defect control applies unchanged. On four-step distilled samplers, its contract leaves nothing to transport, so it returns the exact output by design, and DARC-FS instead recovers efficiency across chunks and refiner blocks. DARC-FS has so far been validated on one model family (SANA-WM), one device (AGX Thor) and ten scenes. Extending it to the other world models of Table 1, and merging its refiner completion into DARC's exposure accounting so that a single budget governs both are open to future work.

I DISCUSSION AND LIMITATIONS

DARC is based on a simple objective: to predict model outputs between fresh evaluations and to let each fresh evaluation measure how good the last prediction was. This section discusses what DARC does and does not promise, where it applies, and two practical clarifications about its use.

I.1 SCOPE AND LIMITATIONS

Empirical control. DARC's exposure score is calibrated online from measured forecast defects, which makes it adaptive to each trajectory without offline fitting. It is an empirical criterion rather than a formal bound on the final video's error, and deriving such a guarantee is a natural extension.

Fidelity to the reference rollout. Like any cache, DARC trades a small amount of per-frame agreement with the EXACT rollout for speed. Over long horizons, this agreement naturally decreases as rollouts diverge (Table 9), while benchmark-level quality stays close to EXACT (Table 1). The budget ε gives a direct option towards higher fidelity when an application requires it (Sec. D).

Regime of applicability. DARC acts where a trajectory has enough native steps to amortize its protected calls, and its benefit grows with the number of steps, from $1.05\times$ at 12 steps to $2.77\times$ at 60 (Table 9). On very short samplers, it returns the exact output by design, and for distilled few-step models, the same *predict* $\rightarrow$ *bound* $\rightarrow$ *fall back* principle carries over to the chunk and refiner axes (Sec. H).

I.2 PRACTICAL CONSIDERATIONS

Autoregressive chunks. Many world models generate long videos chunk by chunk, which raises the question of whether DARC's anchors and calibration can safely span several chunks. DARC follows one rule: its state never outlives the trajectory on which it was measured, because a defect measured on one trajectory says nothing about another. Astra denoises each window as a separate 50-step trajectory, so DARC resets its state at the start of every window. MiniWorld instead uses streaming diffusion forcing, in which one denoising window slides over the chunks. DARC then attaches its two anchors to the window, re-indexing them by global frame index as the window moves. Every chunk that enters the window is evaluated fresh at its first and final step, so new frames start from measured rather than predicted outputs, and all state belonging to a chunk is discarded once the chunk leaves the window.

Memory. Long videos already strain device memory, so a cache must keep its state small. DARC stores the two most recent complete model outputs, which serve as its anchors, two state signatures of p=8192 values each, and one blend coefficient per solver-view sample. With M the size of one model output, this is about $2M$ scalars, the size of two model outputs. WorldCache instead stores block-level hidden-state residuals and probes for both guidance branches over the full token sequence. We did not measure peak memory, so we do not attribute the out-of-memory outcomes of Table 2 to cache state alone; they also depend on each device's offloading configuration.

J DARC: Complete Definitions and Per-Model Implementation

This section gives the complete definitions behind Sec. 3, including clipping rules, numerical floors and the adapter condition, followed by the per-model interface used in our experiments. Notation follows Sec. 3.

J.1 Model–solver interface

Consider one descending pass with N native solver nodes and a non-increasing sequence of scalar timesteps $t_0, \dots, t_{N-1}$. We express one native step at an adapter-defined model–solver boundary as

$$\mathbf{Y}_k = F_\theta(\mathbf{x}_k, t_k; \boldsymbol{\xi}), \qquad \mathbf{V}_k = G(\mathbf{Y}_k), \qquad \mathbf{s}_{k+1} = \Psi_k(\mathbf{s}_k, \mathbf{V}_k). \tag{11}$$

Here $\mathbf{s}_k$ is the complete numerical state at node k, including the current latent and any history the solver needs, and $\mathbf{x}_k = \pi_x(\mathbf{s}_k)$ is the model input. The timestep t_k identifies the noise or flow level, and $\boldsymbol{\xi}$ collects the external conditioning. The model F_θ produces the complete boundary output $\mathbf{Y}_k$, not an intermediate feature or a prediction of the next state. The deterministic adapter G converts it into the solver-facing quantity $\mathbf{V}_k$, including any guidance or parameterization conversion, and Ψ_k is the native numerical update. DARC changes only the source of $\mathbf{Y}_k$ and preserves G, Ψ_k and the node sequence.

At node k, DARC selects a fresh evaluation ($a_k = 1$) or a predicted complete output ($a_k = 0$). With superscript d denoting the DARC trajectory,

$$\widetilde{\mathbf{Y}}_k = \begin{cases} F_\theta(\mathbf{x}_k^d, t_k; \boldsymbol{\xi}), & a_k = 1, \\ \widehat{\mathbf{Y}}_k, & a_k = 0, \end{cases} \qquad \mathbf{s}_{k+1}^d = \Psi_k\Big(\mathbf{s}_k^d, G(\widetilde{\mathbf{Y}}_k)\Big), \qquad \mathbf{x}_k^d = \pi_x(\mathbf{s}_k^d), \tag{12}$$

where $\widehat{\mathbf{Y}}_k$ is built from stored fresh anchors and the current state. Both choices execute the same solver update. The decision a_k uses only information available before the current model call, so the fresh-evaluation count $N_{\text{fresh}} = \sum_k a_k$ depends on the input and the trajectory. The ideal routing signal would be the same-state field defect

$$\delta_k = \Big\| G\big(F_\theta(\mathbf{x}_k^d, t_k; \boldsymbol{\xi})\big) - G(\widehat{\mathbf{Y}}_k) \Big\|_2, \tag{13}$$

which is unavailable at decision time because evaluating it requires the very call being decided. DARC instead acts on a causal history of fresh anchors, lagged forecast defects and inexpensive geometry of the reached state. It performs no auxiliary neural probe at a transported node.

Locality assumptions. DARC assumes only short-range regularity: within an anchor segment, complete outputs vary regularly enough for a damped first-order model to be useful, the state signature retains informative local geometry, and the preferred damping and forecast difficulty persist long enough for one anchor measurement to inform the next segment. These assumptions are empirical, and the budget ε is specific to the adapter and sampling schedule.

J.2 State-conditioned complete-output transport

DARC keeps the two most recent fresh anchors $j_0 < j_1$. Each anchor stores

$$\mathcal{A}_j = \big(j, t_j, \mathbf{z}_j, \mathbf{Y}_j\big), \qquad \mathbf{z}_j = P_x(\mathbf{x}_j^d), \tag{14}$$

where the fixed signature map P_x flattens each batch element, selects entries at a deterministic integer stride and truncates to at most p values.

State–time geometry. For a candidate $k > j_1$, let $\mathbf{z}_k = P_x(\mathbf{x}_k^d)$, $\Delta\mathbf{z}_{01} = \mathbf{z}_{j_1} - \mathbf{z}_{j_0}$ and $\Delta\mathbf{z}_k = \mathbf{z}_k - \mathbf{z}_{j_1}$. For state-batch element $i \in \{1, \dots, B_x\}$, let $D_{01}^{(i)} = \max(\|\Delta\mathbf{z}_{01}^{(i)}\|_2^2, \eta_x)$ and $D_k^{(i)} = \max(\|\Delta\mathbf{z}_k^{(i)}\|_2^2, \eta_x)$. Schedule progress and state progress beyond the latest anchor are

$$\alpha_{t,k} = \mathrm{clip}_{[\alpha_{\min},\alpha_{\max}]} \frac{t_k - t_{j_1}}{t_{j_1} - t_{j_0}}, \qquad \alpha_{x,k}^{(i)} = \mathrm{clip}_{[\alpha_{\min},\alpha_{\max}]} \frac{\langle \Delta\mathbf{z}_k^{(i)}, \Delta\mathbf{z}_{01}^{(i)} \rangle}{D_{01}^{(i)}}. \tag{15}$$

If $|t_{j_1} - t_{j_0}| < \eta_t$, the unclipped time coordinate is set to zero. The two coordinates are combined with a regularized squared alignment score,

$$\rho_k^{(i)} = \left[\mathrm{clip}_{[0,1]} \frac{|\langle \Delta\mathbf{z}_k^{(i)}, \Delta\mathbf{z}_{01}^{(i)} \rangle|}{\sqrt{D_k^{(i)} D_{01}^{(i)}}} \right]^2, \qquad \alpha_k^{(i)} = \rho_k^{(i)} \alpha_{x,k}^{(i)} + (1 - \rho_k^{(i)}) \alpha_{t,k}, \tag{16}$$

so strong alignment favors state progress and weak alignment shifts weight towards schedule progress. The same geometry gives the chord residual used by the exposure score,

$$r_{\mathrm{ch},k}^{(i)} = \frac{\|\Delta\mathbf{z}_k^{(i)} - \alpha_{x,k}^{(i)} \Delta\mathbf{z}_{01}^{(i)}\|_2}{\sqrt{D_k^{(i)}}}. \tag{17}$$

When clipping and the floor are inactive, its numerator is the component of the displacement orthogonal to the anchor chord; otherwise it may also contain an along-chord component. We therefore treat it as a normalized trajectory-mismatch signal, not a curvature estimate.

Complete-output prediction. DARC forms a zero-order hold and a secant candidate,

$$\mathbf{H}_{k|j_1} = \mathbf{Y}_{j_1}, \qquad \mathbf{S}_{k|j_0,j_1}^{(i)} = \mathbf{Y}_{j_1}^{(i)} + \alpha_k^{(i)} \big(\mathbf{Y}_{j_1}^{(i)} - \mathbf{Y}_{j_0}^{(i)} \big), \tag{18}$$

and predicts with the calibrated blend

$$\widehat{\mathbf{Y}}_k = \mathbf{H}_{k|j_1} + \mathcal{E}_G(\boldsymbol{\beta}_{j_1}) \odot \big(\mathbf{S}_{k|j_0,j_1} - \mathbf{H}_{k|j_1} \big), \qquad \boldsymbol{\beta}_{j_1} \in [0,1]^{B_G}, \tag{19}$$

where B_G is the solver-view batch size, $\odot$ denotes elementwise multiplication, and $\mathcal{E}_G$ expands the solver-view coefficients to the corresponding adapter-output elements. The endpoints $\beta = 0$ and $\beta = 1$ recover the hold and the secant. For calibration in the solver view and transport before the adapter to agree, $\mathcal{E}_G$ must satisfy

$$G(\mathbf{H} + \mathcal{E}_G(\boldsymbol{\beta}) \odot (\mathbf{S} - \mathbf{H})) = G(\mathbf{H}) + \boldsymbol{\beta} \odot \big(G(\mathbf{S}) - G(\mathbf{H}) \big). \tag{20}$$

This holds whenever G is affine in $\mathbf{Y}$ within each solver-view sample and $\mathcal{E}_G$ copies $\beta^{(r)}$ to every element of $\mathbf{Y}$ that maps to sample r; Sec. J.5 shows that all four of our adapters have this form.

J.3 Fresh-anchor calibration and defect feedback

Before a fresh call at node k, the current anchors define $\mathbf{H}_{k|j_1}$, $\mathbf{S}_{k|j_0,j_1}$ and the pre-call forecast $\widehat{\mathbf{Y}}_{k,\mathrm{pre}}$. The fresh output $\mathbf{Y}_k = F_\theta(\mathbf{x}_k^d, t_k; \boldsymbol{\xi})$ then supports two measurements before the anchors are updated. At the first calibration node, the pre-call forecast is the hold ($\beta = 0$), since no coefficient has been fitted yet.

Solver-view calibration. Let $\boldsymbol{\varphi}(\mathbf{Y}) = P_y(G(\mathbf{Y}))$, where P_y builds a compact output signature by deterministic flattening, strided sampling and truncation. For solver-view sample $r \in \{1, \dots, B_G\}$, let $\mathbf{y}_k^{(r)} = \boldsymbol{\varphi}(\mathbf{Y}_k)^{(r)}$, $\mathbf{h}_k^{(r)} = \boldsymbol{\varphi}(\mathbf{H}_{k|j_1})^{(r)}$ and $\mathbf{g}_k^{(r)} = \boldsymbol{\varphi}(\mathbf{S}_{k|j_0,j_1})^{(r)} - \mathbf{h}_k^{(r)}$, the secant correction relative to the hold. DARC fits

$$\beta_k^{(r)} = \mathrm{clip}_{[0,1]} \frac{\langle \mathbf{y}_k^{(r)} - \mathbf{h}_k^{(r)}, \mathbf{g}_k^{(r)} \rangle}{\max(\|\mathbf{g}_k^{(r)}\|_2^2, \eta_y)}, \tag{21}$$

the least-squares damping that best explains the fresh output in the sampled solver view.

Forecast-defect feedback. DARC also compares the saved pre-call forecast with the fresh output on the full solver-facing tensors. With $\mathrm{RelRMS}(\mathbf{u}, \widehat{\mathbf{u}}) = \mathrm{RMS}(\mathbf{u} - \widehat{\mathbf{u}}) / \max(\mathrm{RMS}(\mathbf{u}), \eta_r)$ computed over the dimensions of each solver-view sample, the recorded forecast defect is

$$d_k = \max_r \mathrm{RelRMS}\Big(G(\mathbf{Y}_k)^{(r)}, G(\widehat{\mathbf{Y}}_{k,\mathrm{pre}})^{(r)} \Big). \tag{22}$$

Because the forecast was saved before the call, d_k measures the predictor that was available when the routing decision was made, at the state DARC actually reached. DARC then advances the anchors, $(j_0, j_1) \leftarrow (j_1, k)$, and carries β_k and d_k into the new segment. The fresh output $\mathbf{Y}_k$ is the output supplied at node k.

J.4 Defect-exposure control and adaptive reanchoring

Accumulation is motivated by error propagation. For a transported explicit-Euler step with coefficient h_k and an L-Lipschitz solver-facing field, the difference $\mathbf{e}_k = \mathbf{x}_k^d - \mathbf{x}_k^e$ between the DARC and the fully evaluated discrete trajectories satisfies

$$\|\mathbf{e}_{k+1}\|_2 \le (1 + |h_k|L)\|\mathbf{e}_k\|_2 + |h_k|\delta_k. \tag{23}$$

Each update propagates existing error and adds a step-weighted local defect. Since δ_k is unobservable, DARC regulates transport with a causal surrogate.

Exposure score. For candidate $k > j_1$, let

$$q_k = \max_i |\alpha_{x,k}^{(i)} - \alpha_{t,k}|, \qquad \bar{q}_k = \min(q_k, q_{\max}), \qquad r_{\mathrm{ch},k} = \max_i r_{\mathrm{ch},k}^{(i)}, \tag{24}$$

let $\beta_{j_1}^{\max} = \max_r |\beta_{j_1}^{(r)}|$, and let R_{seg} be the exposure accumulated on transported nodes since the latest fresh evaluation. Before routing node k, DARC computes

$$\begin{aligned} \widetilde{d}_k &= \max\big(d_{j_1},\, w_{\beta q}\, \beta_{j_1}^{\max} q_k\big)\,, \qquad I_k = 1 + w_{\mathrm{ch}} r_{\mathrm{ch},k} + w_q \bar{q}_k, \qquad \gamma_k = \frac{|t_k - t_{k-1}|}{T_{\mathrm{norm}}}, \\ R_k^{\mathrm{proj}} &= R_{\mathrm{seg}} + \widetilde{d}_k\, \gamma_k\, I_k. \end{aligned} \tag{25}$$

The proxy $\widetilde{d}_k$ starts from the measured defect d_{j_1} and is raised when state and schedule progress disagree while the secant component is active. The factor I_k inflates exposure with the chord residual and the capped coordinate gap, and γ_k weights it by the native timestep gap. Since all weights are non-negative, the geometry terms can only add fresh evaluations.

Reanchoring rule. DARC transports if and only if

$$R_k^{\mathrm{proj}} < \varepsilon \qquad \text{and} \qquad k - j_1 < m, \tag{26}$$

where ε is the segment budget and m the span cap. An accepted transport commits $R_{\mathrm{seg}} \leftarrow R_k^{\mathrm{proj}}$. Otherwise DARC evaluates the model, measures the forecast defect (Eq. (22)), fits β (Eq. (21)), advances the anchors and resets $R_{\mathrm{seg}} \leftarrow 0$. The reset starts a new predictor segment; it does not remove existing trajectory error. Eq. (23) motivates the score but does not certify it: the implemented defect is relative and lagged, and γ_k is a normalized schedule gap rather than the solver's integration coefficient, so R_{seg} is an empirical routing statistic, not a probability or an error bound.

Protected calls and resets. Four nodes of every trajectory are always evaluated fresh: the first two, which form the initial anchor pair; node 2, which measures the first forecast defect and fits the first β (its pre-call forecast is the hold); and the terminal node, so that the final update always uses a measured output. A fresh evaluation is also forced whenever no defect has been measured yet or the anchor span reaches m, so at most $m - 1$ consecutive nodes are transported. Controller state lives within one descending trajectory: a timestep increase larger than the reset tolerance (Table 4) marks a new trajectory and clears the anchors, the calibration state and the segment score, and the stored anchors are released after the terminal node.

Multistep solvers. Eq. (23) covers a single explicit step. Cosmos samples with FlowUniPC, whose predictor and corrector reuse model outputs from earlier nodes, so a transported output also enters the solver's history and influences the next few updates. DARC does not model this propagation. Two properties limit it: the span cap allows at most $m - 1$ consecutive transported outputs in the history, and every fresh evaluation inserts a measured output. We give no error bound for this case; the Cosmos results in Tables 1 and 2 are its empirical test.

J.5 Per-model interface

Transported tensor and adapter. On SANA-WM, DARC transports the conditional and unconditional outputs of the Stage-1 DiT as one tensor $\mathbf{Y} = [\mathbf{Y}^c, \mathbf{Y}^u]$, and G is the guidance combination

$G(\mathbf{Y}) = \mathbf{Y}^u + w(\mathbf{Y}^c - \mathbf{Y}^u)$ with w=5. $\mathcal{E}_G$ copies each sample's coefficient $\beta^{(r)}$ to both branches of that sample, so Eq. (20) holds exactly because G is linear. On Cosmos-Predict2.5 and MiniWorld-1B, the boundary is placed after guidance: $\mathbf{Y}$ is the already guided output (w=7 and w=2), and G is the identity or the sampler's fixed per-sample affine conversion to its parameterization, with $\mathcal{E}_G$ broadcasting $\beta^{(r)}$ over sample r. Astra uses no guidance, and the same identity adapter applies. In every case, one fresh evaluation is one complete model call at a node, including both guidance branches where present.

Timestep scalarization in MiniWorld. Streaming diffusion forcing assigns each in-flight chunk its own noise level, so the timestep at an outer step is a vector. DARC scalarizes it by its maximum, the level of the noisiest chunk in the window. This scalar enters only the time coordinate $\alpha_{t,k}$, the gap weight γ_k and the reset test; all other quantities are computed from complete window outputs and reduced by their maximum over batch elements (Eqs. (22) and (24)). Each outer step therefore receives one routing decision for the whole window, so the decision follows its worst-case element.

Resets and boundary. Each Astra window is an independent trajectory and starts from an empty controller state; MiniWorld's sliding window is handled as described in Sec. I. Decoding and any downstream refiner remain outside the controlled boundary.

Overhead. With M the number of scalars in the boundary output and $p_{\text{eff}} \leq p$ the realized signature length, a transported node costs $O(M + B_x p_{\text{eff}})$ elementwise work and no model call, and calibration at a fresh node adds $O(M + B_G p_{\text{eff}})$. Persistent state is about $2M + 2B_x p_{\text{eff}} + B_G$ scalars: two output anchors, two state signatures and one blend coefficient per solver-view sample.

K Additional Qualitative Results

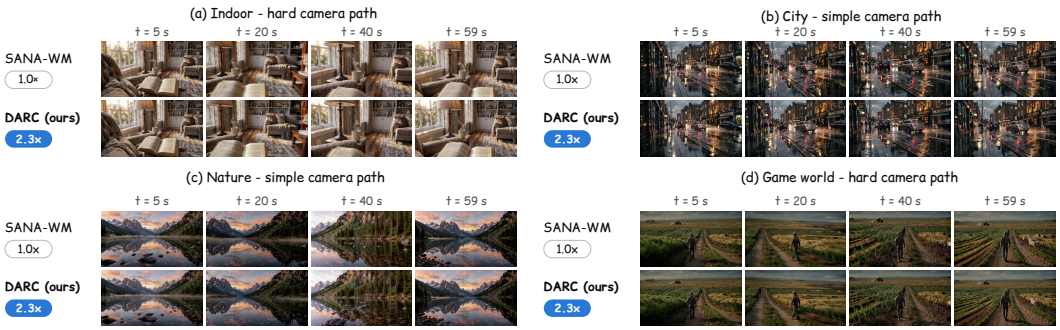


Figure 7: **Long-horizon rollouts on SANA-Bench.** Four 60-second scenes covering indoor, urban, natural and game-like environments under simple and hard camera paths, with frames at 5, 20, 40 and 59 s. In each pair, the top row is the unaccelerated SANA-WM and the bottom row is DARC at the Table-1 operating point (2.3× end-to-end). DARC preserves scene content, layout and camera motion over the full minute, including on the hard camera paths in (a) and (d).

Figures 7, 8 and 9 compare each unaccelerated model with DARC at the operating point of Table 1, and Fig. 10 contrasts DARC with WorldCache. Frames are sampled across the full generated horizon, so drift that accumulates over a rollout would be visible in the later frames. These examples illustrate typical behavior and do not replace the averages of Tables 1 and 5; per-scene paired fidelity varies widely.

Long horizons on SANA-Bench. Figure 7 shows four 60-second SANA-WM rollouts that span indoor, urban, natural and game-like environments under simple and hard camera paths. Over the full minute, DARC preserves the scene content, layout and camera motion of the reference, including on hard camera paths where the camera changes direction and revisits parts of the scene.

Other world models. Figure 8 shows MiniWorld-1B on robot manipulation (DROID-100) and on camera-controlled indoor walkthroughs (RealEstate10K), and Astra on drone exploration from Sekai. DARC keeps the robot arm's motion and its interaction with objects, the room layout under camera

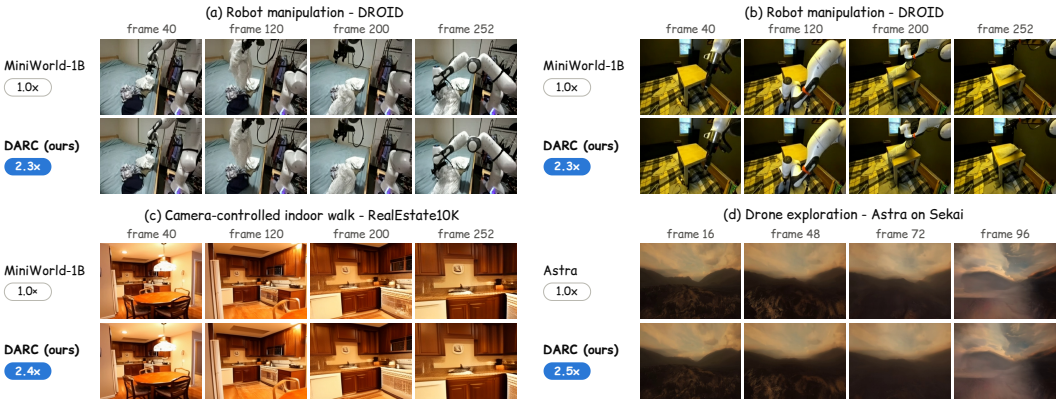


Figure 8: **Qualitative results on other world models.** MiniWorld-1B on robot manipulation from DROID-100 (a, b) and on camera-controlled indoor walkthroughs from RealEstate10K (c), and Astra on drone exploration from Sekai (d). Frames are sampled over the generated horizon (253 frames for MiniWorld-1B, 96 for Astra). In each pair, the top row is the unaccelerated model and the bottom row is DARC at its Table-1 operating point (2.3–2.5× end-to-end). DARC keeps the robot arm's motion and object interactions (a, b), the room layout under camera motion (c), and the terrain and sky structure (d) consistent with the reference.

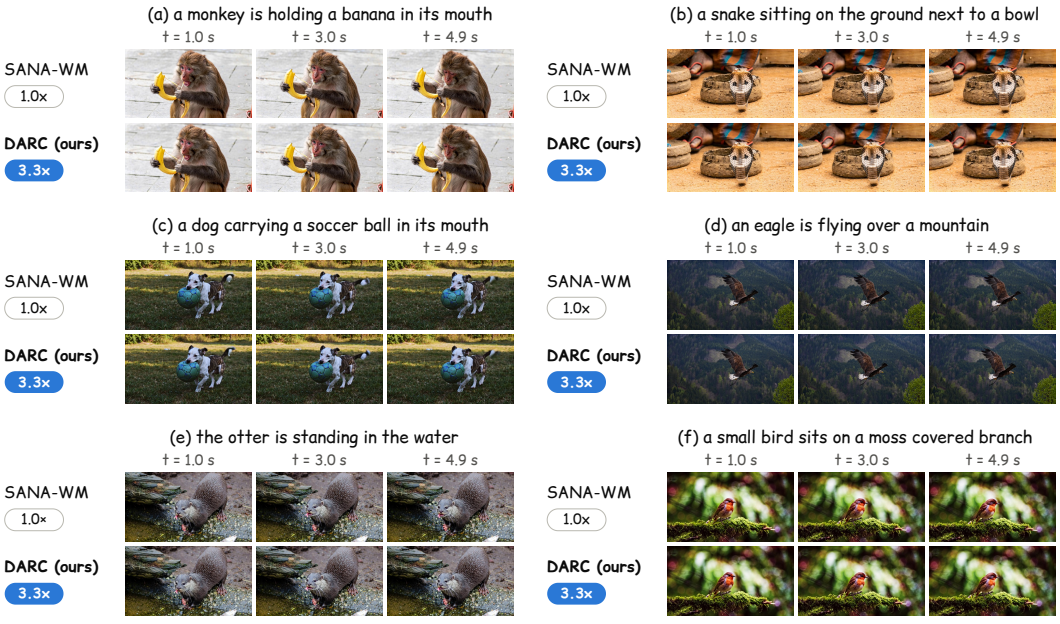


Figure 9: **Qualitative results on VBench-I2V.** Six image-to-video prompts, with frames at 1.0, 3.0 and 4.9 s. In each pair, the top row is the unaccelerated SANA-WM and the bottom row is DARC. DARC preserves each subject, its appearance and its consistency with the conditioning image.

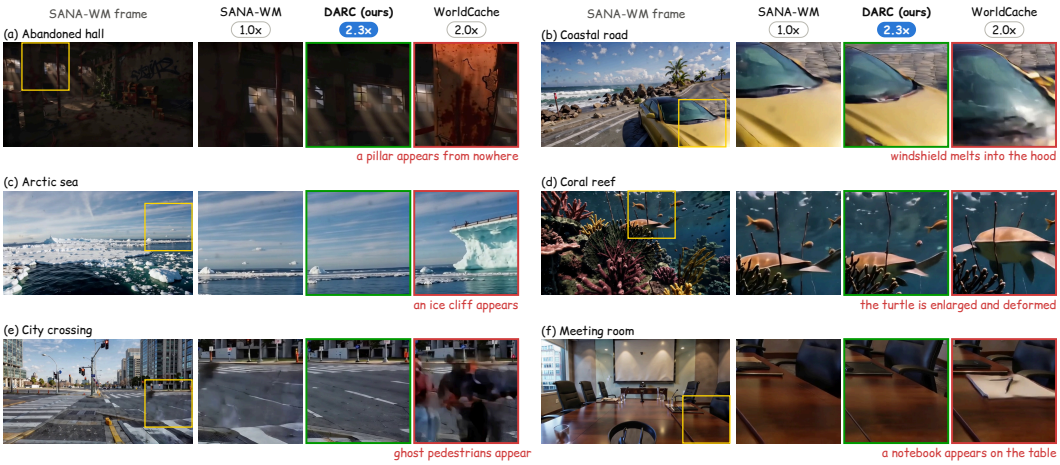


Figure 10: **A failure mode of the fastest prior cache on SANA-Bench.** Six SANA-WM scenes selected to illustrate this failure mode. For each scene, the left image is a full frame of the unaccelerated model with a region of interest (yellow box), and the three crops show this region for SANA-WM, DARC and WorldCache at the same time step. WorldCache (released threshold, $2.0\times$) inserts content that the reference never contains (a, c, e, f) or deforms existing objects (b, d), whereas DARC ($2.3\times$) introduces no such content in these scenes (green boxes). Best viewed zoomed in.

motion, and the terrain and sky structure consistent with the unaccelerated model, while running 2.3–$2.5\times$ faster end-to-end.

Image-to-video. Figure 9 shows six VBench-I2V prompts. DARC preserves each subject, its appearance and its consistency with the conditioning image across the clip; visible differences from the reference are confined to fine texture and small details.

A failure mode of WorldCache. Figure 10 contrasts DARC with WorldCache, the fastest prior cache on SANA-Bench, on six scenes selected to illustrate one characteristic failure. WorldCache inserts content that the unaccelerated model never generates, such as a pillar in an empty hall, an ice cliff on the open sea, ghost pedestrians on a crossing and a notebook on a meeting table, and it deforms existing objects, such as a car's windshield and a turtle. DARC, although faster ($2.3\times$ against $2.0\times$), introduces no such content in these scenes. These examples do not reflect average fidelity: at its released threshold, WorldCache stays closer to EXACT than DARC does on average (paired LPIPS 0.193 against 0.242; Table 5). They matter because inserted objects are local changes of content, which frame-averaged metrics barely register, yet an object that appears from nowhere breaks the persistence that a user or agent relies on when returning to a place.

Few-step generation. Figure 11 shows DARC-FS on the distilled four-step SANA-WM. Over the first seconds, its rollout closely matches EXACT, preserving the room's layout, materials, and lighting. Over the full minute, the two rollouts diverge, as expected for autoregressive generation: the four-step EXACT rollout degrades into speckle artifacts, while DARC-FS stays sharp and coherent, raising VBench from 79.42 to 80.57 at $1.43\times$ end-to-end.

L DARC PSEUDO ALGORITHM

Algorithm 1 gives the complete execution order of DARC for one descending trajectory, combining the transport, calibration and exposure definitions above with the protected calls. Every routing decision uses only information available before the current model call, and both branches apply the same native update Ψ_ℓ.



Figure 11: **DARC-FS on distilled four-step SANA-WM (Jetson AGX Thor).** Frames at 0 s (input), 15 s and 60 s for EXACT (top) and DARC-FS (bottom) on one bathroom scene. The two rollouts match closely at 15 s. By 60 s, the four-step EXACT rollout has degraded into speckle artifacts, while DARC-FS continues along a different but coherent trajectory. On this scene, DARC-FS raises VBench from 79.42 to 80.57 at a $1.43\times$ end-to-end speedup.

Algorithm 1 DARC at the model–solver boundary

Require: model F_θ, native updates $\{\Psi_\ell\}_{\ell=0}^{N-1}$, adapter G, budget ε, maximum span m

initialize empty anchors, $\boldsymbol{\beta} \leftarrow \mathbf{0}$ (hold forecast), no defect, $R_{\text{seg}} \leftarrow 0$
for $\ell = 0, \dots, N-1$ **do**
  obtain the current reached state $\mathbf{x}_\ell^d = \pi_x(\mathbf{s}_\ell^d)$
  if fewer than two anchors exist **then**
    $a_\ell \leftarrow 1$; no pre-call forecast is available
  else
    construct geometry and $\widehat{\mathbf{Y}}_{\ell,\text{pre}}$ via Eqs. (16)–(19)
    if $\ell = 2$, $\ell = N-1$, or the previous defect is unavailable **then**
      $a_\ell \leftarrow 1$
    else
      compute R_ℓ^{proj} using Eq. (25)
      $a_\ell \leftarrow 0$ iff Eq. (26) holds with $k = \ell$ and j_1 the latest anchor; otherwise $a_\ell \leftarrow 1$
    end if
  end if
  if $a_\ell = 1$ **then**
    $\widetilde{\mathbf{Y}}_\ell \leftarrow F_\theta(\mathbf{x}_\ell^d, t_\ell; \boldsymbol{\xi})$
    if a pre-call forecast exists **then**
      measure d_ℓ (Eq. (22)) and fit $\boldsymbol{\beta}_\ell$ (Eq. (21))
    end if
    $(j_0, j_1) \leftarrow (j_1, \ell)$ and $R_{\text{seg}} \leftarrow 0$
  else
    $\widetilde{\mathbf{Y}}_\ell \leftarrow \widehat{\mathbf{Y}}_{\ell,\text{pre}}$ and $R_{\text{seg}} \leftarrow R_\ell^{\text{proj}}$
  end if
  $\mathbf{s}_{\ell+1}^d \leftarrow \Psi_\ell(\mathbf{s}_\ell^d, G(\widetilde{\mathbf{Y}}_\ell))$
end for