STABILIZING LONG-HORIZON STREAMING DMD WITH DRIFT-CONTROLLED ADAPTIVE TTUR

Anonymous authors
Paper under double-blind review

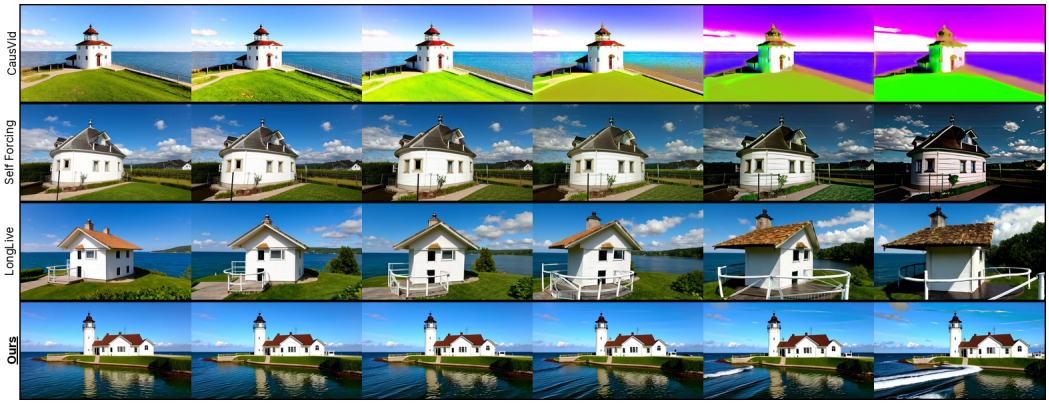

Figure 1: Qualitative comparison on the *lighthouse* prompt.

ABSTRACT

Distribution matching distillation (DMD) relies on an online fake-score model that tracks an evolving generator distribution; tracking errors can perturb generator updates. Fixed two-timescale update rules (TTUR) use constant critic-to-generator ratios that can be poorly suited to the varying tracking burden in long-horizon streaming DMD. We introduce Drift-Gated TTUR, which adapts *only optimizer timing*. A periodic fixed-paired probe holds prompts, latents, diffusion timesteps, and forward noise fixed while measuring temporal changes in the on-policy fake-score field as a one-sided indicator of elevated change in the coupled generator–critic dynamics. After a short calibration, exceeding a frozen threshold triggers a skipped generator update and additional critic recovery; otherwise, the original schedule is retained. DMD objectives and inference remain unchanged, with no gradient through the probe or controller. On the official 946-prompt VBench evaluation in LongLive, our method achieves Total and Quality scores of 81.05 and 83.59, compared with 80.16 and 82.93 for the uniform-schedule control. The Total gain is $+0.89$ with a paired 95% confidence interval of $[+0.60, +1.17]$ across seven matched inference seeds. A random intervention schedule scores 79.58, while removing critic recovery from the uniform schedule yields 80.03. These controls indicate that neither additional critic optimization nor schedule irregularity alone explains the gain, supporting drift-conditioned optimizer timing as an effective control variable for long-horizon streaming DMD.

1 INTRODUCTION

Diffusion models have enabled high-quality video generation, but their iterative denoising process remains expensive for low-latency generation. Distribution matching distillation (DMD) addresses this bottleneck by distilling a pretrained diffusion model into a few-step generator, making fast autoregressive generation practical. However, autoregressive generators introduce a second challenge: training trajectories can differ from the model-generated histories encountered at inference time. Self-Forcing (Huang et al., 2025a) addresses this exposure mismatch by applying DMD to the generator's own autoregressive rollouts, and LongLive (Yang et al., 2026) extends this paradigm to interactive,

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.

long-horizon streaming generation. In these DMD-based systems, the generator is optimized using a frozen diffusion teacher together with an online fake-score model that must continually track the distribution induced by the evolving generator.

This coupling makes fake-score estimation inherently a two-timescale optimization problem. Errors in the learned fake score perturb the practical DMD generator direction, while each generator update changes the distribution that the fake-score model must subsequently track. DMD2 (Yin et al., 2024a) addresses this interaction with a two-timescale update rule (TTUR), assigning multiple critic updates to each generator update; LongLive follows a fixed $5{:}1$ critic-to-generator schedule. Such a fixed ratio, however, allocates the same critic catch-up budget regardless of how the tracking difficulty changes over optimization. Our analysis makes this limitation explicit: fake-score tracking error directly contributes to error in the practical DMD generator direction, while a local tracking recursion separates residual tracking error from the additional burden introduced by moving the generator. This motivates adapting the relative generator and critic timescales to the observed training state rather than fixing them throughout training.

We introduce **Drift-Gated TTUR**, a lightweight controller that adapts optimizer timing using a *fixed-paired drift probe*. At each probe evaluation, the prompt, generator latent, diffusion timestep, and forward-noise draw are held fixed, while the sample is regenerated from the current generator. We measure the mean-squared change in the resulting fake-score outputs across successive evaluations, yielding a drift statistic D_k. Holding the probe bank fixed removes variation due to probe resampling, so D_k reflects temporal changes arising from the coupled generator–critic dynamics. Importantly, we do not treat D_k as a calibrated estimate of the unobservable tracking error: elevated drift is used as a one-sided trigger for intervention, whereas a small value is not interpreted as evidence that the critic is accurately tracking the generator.

After an initial calibration period, the drift threshold is frozen. Before each scheduled generator update, the most recent probe value determines whether the update is executed or skipped. When the threshold is exceeded, the generator update is suppressed and the critic receives K additional recovery steps; otherwise, the original schedule is retained. Normal critic updates continue in either case. The controller therefore changes only the timing and allocation of optimizer updates: the underlying DMD losses and inference procedure remain unchanged, and no gradient flows through the probe or gate.

We evaluate Drift-Gated TTUR in the LongLive streaming-DMD setting using the official VBench (Huang et al., 2024) protocol over 946 prompts and all 16 dimensions. At iteration 2,000, Drift-Gated TTUR improves VBench Total from 80.16 to 81.05 over the uniform-schedule control, a gain of $+0.89$ with a paired 95% interval of $[+0.60, +1.17]$. Additional random and skip-only controls indicate that the gain is not explained by schedule irregularity or additional critic optimization alone. Comparisons with official Self-Forcing and LongLive checkpoints provide further context, although these external comparisons are not controlled. Together, these results support the central claim of this work: *when* generator updates are applied is an important control variable in long-horizon streaming DMD.

Our contributions are:

- We connect **fake-score tracking error** to perturbations of the practical DMD generator direction and derive a local tracking recursion that exposes the limitation of a fixed two-timescale update schedule (Section 3).
- We introduce Drift-Gated TTUR, which combines a fixed-paired temporal probe with a **frozen-threshold controller** that adapts generator updates and critic recovery without modifying the DMD losses or inference procedure (Section 3.2).
- We evaluate **drift-conditioned scheduling** against fixed, uniform, random, and skip-only alternatives in streaming DMD. On the official 946-prompt VBench suite, Drift-Gated TTUR improves over the uniform control, while additional controls show that the gain cannot be explained by recovery computation or schedule irregularity alone (Section 5).

2 Related Work

Distribution matching distillation. DMD (Yin et al., 2024b) aligns generator and teacher distributions using an online fake-score model. DMD2 (Yin et al., 2024a), SGMD (Wu et al., 2026d),

and SenseFlow (Ge et al., 2026) improve optimization or score-tracking stability, while FSF-DMD (Kim et al., 2026) replaces the auxiliary score network for flow-map generators. DP-DMD (Wu et al., 2026b) and CrossDistill (Liu et al., 2026b) assign complementary objectives across denoising stages to balance quality and diversity; GDMD (Dong et al., 2026) introduces gradient-level reward guidance.

Autoregressive video distillation. Self-Forcing (Huang et al., 2025a) and LongLive (Yang et al., 2026) establish on-policy distillation for autoregressive and streaming generation. Causal Forcing (Zhu et al., 2026), Causal Forcing++ (Zhao et al., 2026), and Causal-rCM (Zheng et al., 2026) improve causal student initialization, while Context Forcing (Chen et al., 2026) and Diagonal Distillation (Liu et al., 2026a) modify temporal conditioning. One-Forcing (Feng et al., 2026), DistillAlign (Li et al., 2026), and DynaForcing (Huang et al., 2026) explore adversarial, consistency, and rollout constraints, respectively. Reward Forcing (Lu et al., 2025) combines reward-weighted DMD (Re-DMD) with EMA-Sink to improve motion dynamics and maintain streaming context. Stream-R1 (Wu et al., 2026a) and Delta Forcing (Wu et al., 2026c) use reliability or trajectory-based signals to guide training. We focus on optimizer scheduling, retaining the DMD objectives and inference procedure while using temporal fake-score drift to schedule generator skipping and critic recovery without external rewards.

3 DRIFT-GATED TTUR

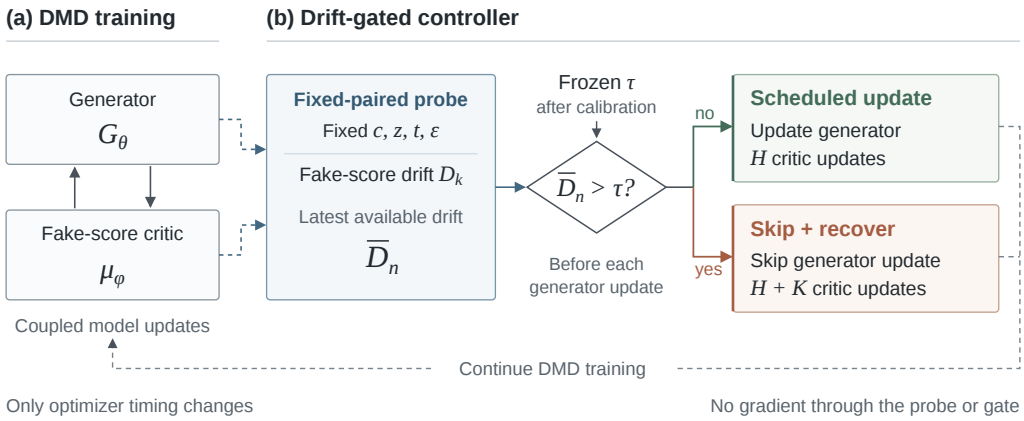

Figure 2: Drift-Gated TTUR couples a measurement loop to the two-optimizer training schedule. A fixed-paired probe holds the prompt, latent seed, diffusion timestep, and added-noise draw constant. It yields a drift statistic D_k for the current generator and fake-score model (critic). After calibration, a frozen threshold τ converts each high-drift slot into one skipped generator update and K immediate critic recovery steps; no gradient flows through D_k. The cadence is one normal critic update every outer step, with the probe evaluated every $H{=}5$ outer steps. The DMD losses and inference path are unchanged: only the timing of the two optimizers changes.

3.1 MOTIVATION: A CHANGING TRACKING BURDEN

The critic is trained to approximate the score of the current generator distribution, while each generator update changes the distribution that the critic must track. Existing DMD methods (Yin et al., 2024a) address this interaction with a *fixed two-timescale update rule (TTUR)*, typically performing multiple critic updates for each generator update. In our streaming baseline, this corresponds to a fixed critic-to-generator update ratio of 5:1.

The TTUR framework of Heusel et al. (2017) uses separate learning rates to coordinate generator and discriminator dynamics in GANs. We draw on its two-timescale perspective: the score estimator should have time to adapt as the generator distribution changes. Here, this motivates drift-conditioned generator skipping and critic recovery while retaining the original learning rates and DMD objectives.

Because the generator distribution evolves throughout training, the generator score appearing in the DMD update is itself a moving target. Thus, all else being equal, poor tracking of the current

generator score can directly translate into error in the generator update. Appendix A.6 formalizes this connection through an exact generator-direction error identity and its upper bound (Eqs. (23)–(24)).

The limitation of a fixed schedule is that the tracking difficulty need not remain constant throughout training. The local analysis in Appendix B separates residual tracking error from the additional burden introduced by moving the generator, making explicit why the required critic catch-up can vary across updates. This limitation is particularly relevant in long-horizon DMD training, where the coupled generator–critic dynamics may change substantially over the course of optimization, so that a fixed ratio cannot remain appropriate throughout. This motivates replacing the fixed update ratio with a controller that adapts the relative generator and critic timescales to the observed training state. Appendix B provides the full derivation of Eqs. (31)–(32). This tracking problem motivates the adaptive two-timescale formulation developed below.

3.2 Drift measurement and update scheduling

Neither the current generator score nor the resulting tracking error is observable during training. We therefore introduce a lightweight probe that monitors temporal changes in the online fake denoiser on a matched set of on-policy generator samples.

We construct a *fixed probe bank* $\mathcal{P} = \{(c^j, z^j, t^j, \epsilon^j)\}_{j=1}^m$, where the conditioning c^j, generator latent z^j, diffusion timestep t^j, and forward noise ϵ^j are fixed throughout training. At probe evaluation k, the current generator produces the corresponding noised sample

$$x_k^j = \alpha_{t^j} G_{\theta_k}(z^j, c^j) + \sigma_{t^j} \epsilon^j, \tag{1}$$

and the current critic is evaluated as $F_k^j = \mu_{\phi_k}(x_k^j, t^j, c^j) \in \mathbb{R}^d$. We define the mean-squared temporal change

$$D_k = \frac{1}{md} \sum_{j=1}^{m} \left\| F_k^j - F_{k-1}^j \right\|_2^2. \tag{2}$$

Fixing the probe bank removes variation due to resampling, while the probe inputs themselves remain on-policy: x_k^j is regenerated from the current generator at every evaluation. Consequently, D_k reflects changes arising from both the generator and the critic. Appendix B decomposes these two contributions in Eq. (36).

Eq. (36) also clarifies the role of the probe. The statistic D_k is *not a calibrated estimator* of the latent tracking error r_n: a moving generator may yield $D_k > 0$ even under accurate tracking, while a biased but nearly stationary critic may yield a small D_k. Instead, we use D_k as an observable, one-sided indicator of elevated change in the coupled generator–critic dynamics. We therefore treat a large D_k as a trigger for recovery, while a small value is not interpreted as a certificate of accurate tracking. The effectiveness of conditioning the update schedule on this signal, beyond changing the aggregate optimizer counts, is tested directly with matched controls in Section 4.

We apply the controller at generator opportunities indexed by n. Let $\bar{D}_n$ denote the most recent probe value available at the beginning of cycle n. The generator gate and critic budget are

$$u_n = \mathbf{1}\{\bar{D}_n \le \tau\}, \qquad c_n = H + K(1 - u_n), \tag{3}$$

where H is the default critic budget between consecutive generator opportunities, K is the additional recovery budget, and τ is the change threshold. Because the baseline performs one normal critic update per outer step and generator opportunities are H outer steps apart, the default critic budget per cycle is also H. When $\bar{D}_n \le \tau$, the ordinary schedule is retained. When $\bar{D}_n > \tau$, the **offered generator update is skipped** ($u_n = 0$) and K **additional critic updates** are allocated during the resulting recovery cycle. Under the local assumptions in Appendix B, Eq. (35) explains the two effects of this action: skipping avoids another generator-induced target shift, while additional critic updates reduce the modeled residual tracking bound.

The threshold is calibrated from the first M non-null probe observations,

$$\tau = \kappa \ \mathrm{median}\{D_1, \ldots, D_M\}, \tag{4}$$

and is *frozen thereafter*. Before calibration is complete, all scheduled generator updates are executed normally. No gradient flows through D_k, the threshold, or the binary gate; the underlying DMD objectives remain unchanged.

The intervention is deliberately conservative. Reacting to a benign high-change event delays one generator update and incurs a bounded amount of additional critic computation, whereas continuing to move the generator in a rapidly changing regime introduces another source of tracking burden. Because the probe is evaluated periodically, the decision is based on the most recent available measurement rather than an instantaneous observation. We therefore describe the method as a *delayed event-triggered TTUR controller*. We refer to the resulting method as *Drift-Gated TTUR*.

Algorithm 1 gives the complete training loop, and Figure 2 summarizes the controller.

4 Experimental Setup

4.1 Training and controller settings

Training setting. We build on the LongLive (Yang et al., 2026) streaming-DMD training pipeline. Both the generator and the online critic are adapted from the ODE-initialized checkpoint using rank-256 LoRA modules. All primary training configurations run for 2,000 outer steps from the same initialization and with the *same training seed*. Each trained checkpoint is evaluated under seven matched inference seeds. Training uses two nodes with eight GPUs each and one video per GPU (global batch size 16). The generator and critic learning rates are 10^{-5} and 2×10^{-6}, respectively. Following the default LongLive schedule, each outer step performs one standard critic update, while a generator update is scheduled every five outer steps.

Controller settings. For Drift-Gated TTUR, we evaluate the fixed-paired probe every $H = 5$ outer steps using $m = 2$ probe items, each containing 21 streamed frames. The drift threshold is calibrated from the first $M = 40$ valid drift measurements as described in Section 3.2, with $\kappa = 3$, and is then fixed for the remainder of training. When the threshold is exceeded, the controller skips the scheduled generator update and performs $K = 5$ additional critic updates.

To ensure that probe evaluation does not perturb the underlying training trajectory, we preserve and restore both the global random-number-generator state and the streaming key–value caches around each probe evaluation. The Fixed TTUR baseline evaluates the same probe only for logging, whereas Drift-Gated TTUR uses the resulting drift measurements to control the update schedule. We additionally ablate the drift-threshold multiplier κ over $\{2, 3, 4\}$ in Section 5.4; the default $\kappa = 3$ provides the best quality–intervention trade-off among the tested values.

4.2 Matched scheduling controls

Because Drift-Gated TTUR changes both the timing and the allocation of optimizer updates (gradient steps of either the generator or the critic), comparison with Fixed TTUR alone cannot isolate the effect of drift-conditioned scheduling. We therefore introduce matched controls that remove the drift signal while preserving different components of the training schedule.

Count-Matched Uniform (Unif). Unif uses *exactly the same optimizer budget* as Drift-Gated TTUR, consisting of 223 generator updates and 885 recovery critic updates. It places the generator skips uniformly over training rather than conditioning them on the measured drift D_k. The uniform mask distributes 177 skips across the 400 scheduled generator opportunities with a Bresenham placement, leaves the first 42 opportunities skip-free to match the calibration window, and is audited to agree with the drift-conditioned gate in only $162/400$ (40.5%) of slots, so the control re-times rather than copies the interventions. Each skip is followed by the same $K = 5$ critic recovery updates. The budget matching is *retrospective*: the intervention count measured in the Drift-Gated TTUR run is hard-coded into the uniform mask, and the matched totals are verified by a post-training optimizer-count audit. Thus, Unif isolates whether performance depends on *when* the same update budget is allocated.

Uniform Skip-Only (SkipOnly). SkipOnly uses the same uniform generator skip schedule as Unif but removes the additional critic recovery updates. Comparing SkipOnly with Unif isolates the effect of recovery optimization from that of reducing the generator update frequency.

Random-mask control (Random). As an additional control, we randomly place the same number of generator skips as in Unif, while preserving the same recovery budget. This tests whether the drift signal provides useful scheduling information beyond simply using a non-uniform skip pattern.

5 RESULTS

5.1 DRIFT-CONDITIONED TIMING MATTERS BEYOND UPDATE COUNTS

We evaluate generation quality using the official VBench (Huang et al., 2024) protocol over 946 prompts and all 16 dimensions. We first compare the matched scheduling controls to disentangle drift-conditioned timing from additional critic recovery.

The comparison with Unif is our **primary controlled result**. The two configurations share the same training setup and matched optimizer budget, differing in whether update placement is conditioned on the measured drift signal. As shown in Table 1, Drift-Gated TTUR achieves a VBench Total of 81.05, compared with 80.16 for Unif, and obtains the highest Quality score among the evaluated configurations. The Total gain is $+0.89$, with a paired 95% confidence interval of $[+0.60, +1.17]$ (Table 3). Because the two schedules use identical optimization budgets but place their updates differently, this result shows that drift-conditioned placement is beneficial even when the overall update budget is held fixed: the controller's benefit **cannot be explained by update counts alone**.

The interval is paired across the seven matched inference seeds; with one training run per configuration, it quantifies *inference-time stochasticity rather than variability across training runs*.

Table 1: VBench scores for the control configurations.

Configuration	Quality ↑	Semantic ↑	Total ↑
Fixed TTUR	82.56 ± 0.22	70.48 ± 0.49	80.14 ± 0.26
Unif	82.93 ± 0.08	69.08 ± 0.51	80.16 ± 0.11
Random	82.41 ± 0.18	68.25 ± 0.56	79.58 ± 0.20
Drift-Gated TTUR	$\mathbf{83.59 \pm 0.29}$	$\mathbf{70.90 \pm 0.87}$	$\mathbf{81.05 \pm 0.30}$
SkipOnly	82.47 ± 0.30	70.30 ± 0.76	80.03 ± 0.19

All configurations are trained for 2,000 iterations and evaluated at iteration 2,000; values are mean $\pm$ one sample standard deviation over 7 inference seeds. Fixed TTUR uses 400 generator and 2,000 critic updates. SkipOnly uses 223 generator and 2,000 critic updates. Unif, Random, and Drift-Gated TTUR each use 223 generator, 2,000 normal critic, and 885 recovery critic updates.

Recovery computation alone does not explain the improvement. Removing recovery updates from Unif (Unif $\rightarrow$ SkipOnly) changes VBench Total only from 80.16 to 80.03 (-0.13, 95% interval $[-0.36, +0.09]$).

Likewise, Random uses exactly the same update counts as Unif but reaches only 79.58. Thus, neither additional critic optimization nor arbitrary non-uniform placement is sufficient; the measured drift signal provides useful information for deciding when to intervene.

5.2 COMPARISON WITH EXTERNAL CHECKPOINTS

For additional context, we also evaluate the official Self-Forcing, LongLive, and CausVid (Yin et al., 2025) checkpoints using the same VBench protocol. Drift-Gated TTUR achieves a Total score of 81.05, compared with 80.32 for Self-Forcing, 80.21 for LongLive, and 74.05 for CausVid. The point estimates over Self-Forcing and LongLive are $+0.73$ and $+0.84$, respectively, although their paired 95% confidence intervals, $[-0.14, +1.58]$ and $[-0.03, +1.75]$, include zero. The substantially larger margin over CausVid ($+7.00$) should be interpreted in the context of its more aggressive speed–quality trade-off.

On VBench-Long, Drift-Gated TTUR obtains Quality and Total scores of 0.848 and 0.787, respectively, compared with 0.824/0.768 for Self-Forcing and 0.801/0.753 for LongLive (Table 2).

Table 2: Official VBench evaluation over 946 prompts and 16 dimensions, together with VBench-Long (Huang et al., 2025b).

	Official VBench			VBench-Long		
Configuration	Quality ↑	Semantic ↑	Total ↑	Quality ↑	Semantic ↑	Total ↑
Drift-Gated TTUR	**83.59**	70.90	**81.05**	**0.848**	0.546	**0.787**
Self-Forcing	82.65	71.01	80.32	0.824	0.544	0.768
LongLive	82.18	**72.30**	80.21	0.801	**0.559**	0.753
CausVid	76.21	65.42	74.05	0.770	0.467	0.709

Drift-Gated TTUR is evaluated at iteration 2,000 and reported as a mean over seven matched inference seeds. Self-Forcing, LongLive, and CausVid use their official checkpoints under the same VBench evaluation protocol; all methods generate videos of 20–30 seconds. The last three columns report VBench-Long (part of VBench++ (Huang et al., 2025b)). For VBench-Long, we report Total as $(4 \times \text{Quality} + \text{Semantic})/5$, following the evaluation protocol used by LongLive (Yang et al., 2026); scores are on a 0–1 scale. Six of the seven inference seeds are shared across all methods on VBench-Long.

Table 3: Paired differences in VBench Total.

Comparison	Δ Total	95% CI
Drift-Gated TTUR − Unif	+0.89	[+0.60, +1.17]
Drift-Gated TTUR − Self-Forcing	+0.73	[−0.14, +1.58]
Drift-Gated TTUR − LongLive	+0.84	[−0.03, +1.75]
Drift-Gated TTUR − CausVid	+7.00	[+6.07, +7.97]

The Drift-Gated TTUR–Unif interval is a paired t interval over the seven matched inference seeds; with one training run per configuration, it measures inference-time stochasticity conditional on the trained checkpoints. Δ Total equals the corresponding difference of Total scores in Tables 1 and 2. Intervals for the external checkpoints are computed by paired bootstrap over the official VBench prompt subsets. External baselines are contextual comparisons because their training data, schedules, and optimization budgets are not controlled.

5.3 Branch value of drift-gated updates

We further test the local decision value of the gate through branch interventions. At selected generator opportunities, we fork the same training state into two trajectories: one executes the scheduled generator update, while the other applies the gate intervention by skipping the update and allowing critic recovery. We then compare the subsequent probe drift over a fixed horizon. As shown in Figure 3, gating consistently reduces future drift, with a larger average reduction at high-drift states than at low-drift states. These branch results provide direct evidence that the gate preferentially intervenes at states where delaying the generator update reduces subsequent probe drift.

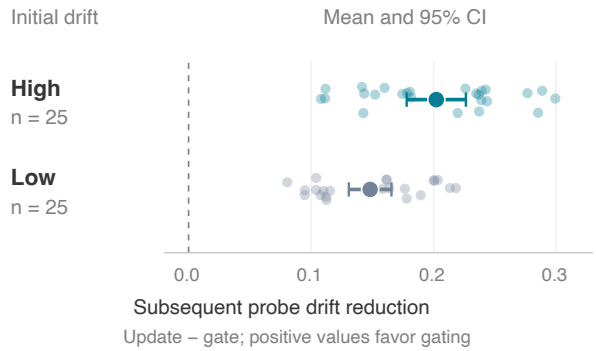

Figure 3: Branch intervention analysis. From the same training state, we compare executing the scheduled generator update (A) against applying the gate intervention (B: skip + critic recovery). Positive $D_k^{\mathrm{A}} - D_k^{\mathrm{B}}$ indicates lower subsequent probe drift under gating. Points show individual branches (n=25 per group); bars denote mean and 95% CI. The reduction is larger for states with high initial drift.

5.4 SENSITIVITY TO THE DRIFT THRESHOLD

We examine the sensitivity of Drift-Gated TTUR to the threshold multiplier κ, which controls how strongly the measured probe drift must exceed its calibration baseline before the controller intervenes. We compare $\kappa \in \{2, 3, 4\}$ and evaluate each trained checkpoint using the same seven inference seeds as in Table 1. The default $\kappa = 3$ achieves the best mean VBench Total in the tested range, with a score of 81.05, compared with 80.70 for $\kappa = 2$ and 80.29 for $\kappa = 4$ (Table 4). Relative to $\kappa = 3$, the mean paired differences across the seven matched inference seeds are -0.35 for $\kappa = 2$ and -0.76 for $\kappa = 4$.

Table 4: Sensitivity to the drift-threshold multiplier κ. Each trained checkpoint is evaluated with the same seven inference seeds as in Table 1. Reported VBench scores are mean $\pm$ one sample standard deviation over these seven evaluations; the κ=3 row corresponds to the Drift-Gated TTUR row of Table 1. Δ Total is computed by pairing the corresponding seeds against the default $\kappa = 3$ setting.

κ	Quality ↑	Semantic ↑	Total ↑	Δ Total vs. κ=3
2	82.81 ± 0.28	$\mathbf{72.25 \pm 0.68}$	80.70 ± 0.26	-0.35
3	$\mathbf{83.59 \pm 0.29}$	70.90 ± 0.87	$\mathbf{81.05 \pm 0.30}$	–
4	82.50 ± 0.35	71.46 ± 0.97	80.29 ± 0.34	-0.76

Changing κ also changes the intervention pattern during training. The fractions of post-calibration decisions above threshold are 49.3%, 45.1%, and 26.7% for $\kappa = 2, 3, 4$, respectively. As expected, increasing κ makes the gate more conservative and substantially reduces the number of interventions. The intermediate setting $\kappa = 3$ gives the best VBench Total *among the tested thresholds*.

The difference is concentrated mainly in the Quality component. While the Semantic scores remain similar for $\kappa = 2$ and $\kappa = 4$ (72.25 and 71.46), their Quality scores are lower than that of the default setting (82.81 and 82.50, compared with 83.59 for $\kappa = 3$). This is consistent with the controller primarily affecting the generation-quality and stability behavior targeted in this work.

5.5 DRIFT DYNAMICS DURING TRAINING

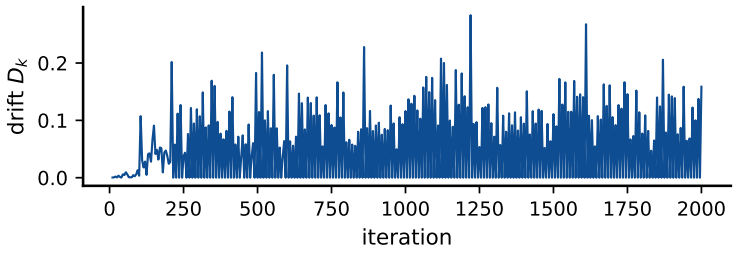

Figure 4: Temporal drift statistic D_k during Drift-Gated TTUR training. High-drift events continue to occur throughout the 2,000-iteration run.

Figure 4 shows the measured drift statistic D_k over the full 2,000-iteration run. The statistic remains intermittently elevated throughout training rather than concentrating in a single stage. After calibration, the frozen threshold is $\tau \approx 0.0927$, and the controller skips 177 of the 400 scheduled generator opportunities. This persistent variation motivates adapting the update schedule throughout training rather than using a fixed intervention pattern.

5.6 QUALITATIVE COMPARISONS

Figure 1 shows a qualitative comparison on the *lighthouse* prompt, with frames sampled at matched relative positions across the full 20–30 s rollout.

Figure 5 extends the qualitative comparison of Figure 1 to three further prompts: a camel walking through sand dunes, a fish swimming in the ocean, and an astronaut flying in space. The protocol is identical to Figure 1: each block shows one prompt, rows are CausVid, Self Forcing, LongLive, and Drift-Gated TTUR, and frames are sampled at matched relative positions (0%, 15%, 30%, 50%, 75%, and the final frame) across each method's full rollout.

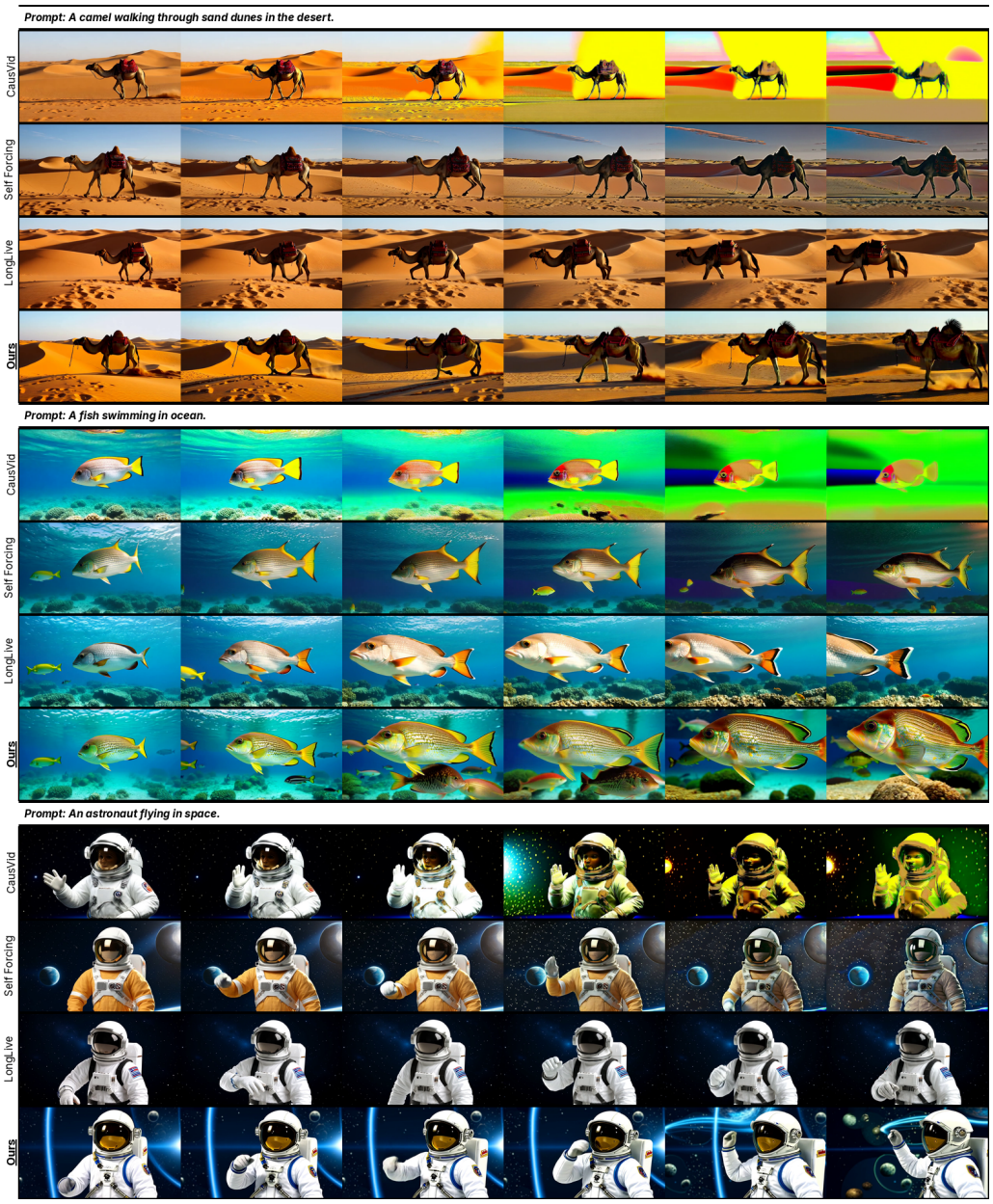

Figure 5: Qualitative comparison on three additional prompts. Within each block, rows are CausVid, Self Forcing, LongLive, and Drift-Gated TTUR (top to bottom), and columns show frames sampled at matched relative positions across each rollout, ending on the final frame.

6 Conclusion

We presented Drift-Gated TTUR, a simple adaptive controller for long-horizon streaming DMD. Rather than modifying the distribution-matching objective, the controller monitors temporal changes in the on-policy fake-score field and uses them to decide when the generator should advance and when the critic should be given additional time to recover. Across the official VBench evaluation and matched scheduling controls, drift-conditioned timing improves over the evaluated fixed and drift-agnostic alternatives. Our results highlight an overlooked aspect of DMD training: the **relative timing of generator and critic optimization** is itself an important degree of freedom, and adapting this timing to the evolving training dynamics can improve long-horizon generation without changing the model or inference pipeline.

AI USE STATEMENT

In this work, we used generative AI tools to help organize the manuscript, edit prose for readability, and assist with research code and figure-generation scripts. We have not used generative AI tools to generate synthetic evaluation datasets, invent numerical results, or replace author verification of claims or citations. Additionally, we used generative AI tools for literature search support and for drafting candidate wording that was subsequently rewritten by the authors. All experiment definitions, code changes, reported artifacts, numerical claims, and citations were reviewed by the authors against the source repository and primary references. AI-assisted code was subjected to static checks or executable tests before its outputs were treated as evidence. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies training stability for generative video models and does not involve human subjects or private data. Evaluation uses public prompt suites from VBench and VBench-Long. More reliable long-video generation can support creative and interactive applications, but can also increase the scale or consistency of misleading synthetic media. Any model or code release should retain the licenses and safety restrictions of the underlying LongLive models and clearly document intended use and limitations.

REPRODUCIBILITY STATEMENT

The appendix records the upstream LongLive snapshot, training hyperparameters, probe definition, controller semantics, optimizer counts, evaluation prompts, the matched-control and official-suite protocols, and the probe correction that separates the primary run from an earlier confounded run. The method itself is specified in the main text and appendix at the level of the probe definition, gating rule, and optimizer schedule, which is sufficient to reimplement the controller on top of the public LongLive training pipeline. Experiment definitions and compact audit outputs are retained in the project repository; large checkpoints, videos, and logs are stored in a versioned external artifact tree. Code and configuration will be shared upon request.

REFERENCES

Shuo Chen, Cong Wei, Sun Sun, Ping Nie, Kai Zhou, Ge Zhang, Ming-Hsuan Yang, and Wenhu Chen. Context Forcing: Consistent autoregressive video generation with long context. *arXiv preprint arXiv:2602.06028*, 2026. URL https://arxiv.org/abs/2602.06028.

Linwei Dong, Ruoyu Guo, Ge Bai, Zehuan Yuan, Yawei Luo, and Changqing Zou. Guiding distribution matching distillation with gradient-based reinforcement learning. *arXiv preprint arXiv:2604.19009*, 2026. URL https://arxiv.org/abs/2604.19009.

Jiaqi Feng, Justin Cui, Yuanhao Ban, and Cho-Jui Hsieh. One-Forcing: Towards stable one-step autoregressive video generation. *arXiv preprint arXiv:2605.23458*, 2026. URL https://arxiv.org/abs/2605.23458.

Xingtong Ge, Xin Zhang, Tongda Xu, Yi Zhang, Xinjie Zhang, Yan Wang, and Jun Zhang. SenseFlow: Scaling distribution matching for flow-based text-to-image distillation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/94a98f4338b6e5fba81344f961bac8e5-Abstract-Conference.html.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, 2025a.

Yubo Huang, Sirui Zhao, Xinchen Yao, Zhengye Zhang, Jinyang Huang, Fengqi Cui, Shiwei Wu, and Enhong Chen. DynaForcing: Overcoming dynamic collapse in self-forcing distillation for streaming avatar generation. In *ACM International Conference on Multimedia*, 2026. URL https://arxiv.org/abs/2608.17707.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and

Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b. doi: 10.1109/TPAMI.2025.3633890.

Youngjoong Kim, Deokyeong Lee, and Jaesik Park. Distribution matching distillation without fake score network. *arXiv preprint arXiv:2605.19256*, 2026.

Jiaxing Li, Kai Zou, Cindy Zhou, Kaichen Huang, Junyao Gao, Zile Wang, Yang Liu, Bin Liu, Bo An, and Yangguang Li. DistillAlign: Coordinating mode covering and mode seeking in autoregressive video distillation. *arXiv preprint arXiv:2607.26811*, 2026.

Jinxiu Liu, Xuanming Liu, Kangfu Mei, Yandong Wen, Ming-Hsuan Yang, and Weiyang Liu. Streaming autoregressive video generation via diagonal distillation. In *International Conference on Learning Representations*, 2026a. URL https://arxiv.org/abs/2603.09488.

Yuxi Liu, Haoyu Li, Yixiang Cai, Tengxu Sun, Zekun Zhang, Baole Ai, Ang Wang, Jiamang Wang, Lin Qu, Kun Yuan, and Kai Zhang. CrossDistill: Balancing quality and diversity via trajectory-level hybrid few-step distillation. *arXiv preprint arXiv:2609.14725*, 2026b. URL https://arxiv.org/abs/2609.14725.

Yunhong Lu, Yanhong Zeng, Haobo Li, Hao Ouyang, Qiuyu Wang, Ka Leong Cheng, Jiapeng Zhu, Hengyuan Cao, Zhipeng Zhang, Xing Zhu, Yujun Shen, and Min Zhang. Reward forcing: Efficient streaming video generation with rewarded distribution matching distillation. *arXiv preprint arXiv:2512.04678*, 2025.

Bin Wu, Mengqi Huang, Shaojin Wu, Weinan Jia, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Stream-R1: Reliability-perplexity aware reward distillation for streaming video generation. *arXiv preprint arXiv:2605.03849*, 2026a.

Tianhe Wu, Ruibin Li, Lei Zhang, and Kede Ma. Diversity-preserved distribution matching distillation for fast visual synthesis. *arXiv preprint arXiv:2602.03139*, 2026b. URL https://arxiv.org/abs/2602.03139.

Yuheng Wu, Xiangbo Gao, Tianhao Chen, Xinghao Chen, Qing Yin, Zhengzhong Tu, and Dongman Lee. Delta forcing: Trust region steering for interactive autoregressive video generation. *arXiv preprint arXiv:2605.14382*, 2026c.

Zhuguanyu Wu, Ruihao Gong, Yang Yong, Yushi Huang, Xiangyu Fan, Lei Yang, Dahua Lin, and Xianglong Liu. SGMD: Score gradient matching distillation for few-step video diffusion distillation. In *International Conference on Machine Learning*, 2026d. URL https://arxiv.org/abs/2605.30116.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time interactive long video generation. In *International Conference on Learning Representations*, 2026.

Tianwei Yin, Micha"el Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems*, 2024a.

Tianwei Yin, Micha"el Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Frédo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Min Zhao, Hongzhou Zhu, Kaiwen Zheng, Zihan Zhou, Bokai Yan, Xinyuan Li, Xiao Yang, Chongxuan Li, and Jun Zhu. Causal Forcing++: Scalable few-step autoregressive diffusion distillation for real-time interactive video generation. *arXiv preprint arXiv:2605.15141*, 2026. URL https://arxiv.org/abs/2605.15141.

Kaiwen Zheng, Guande He, Min Zhao, Jintao Zhang, Huayu Chen, Jianfei Chen, Chen-Hsuan Lin, Ming-Yu Liu, Jun Zhu, and Qianli Ma. Causal-rCM: A unified teacher-forcing and self-forcing open recipe for autoregressive diffusion distillation in streaming video generation and interactive world models. *arXiv preprint arXiv:2606.25473*, 2026. URL https://arxiv.org/abs/2606.25473.

Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal Forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.02214.

A DMD Derivation and Implementation Correspondence

This appendix expands the DMD review in Section 3. We retain the conditioning variable c in the definitions and suppress it only when it does not affect the calculation.

A.1 DMD as score matching between evolving distributions

We begin with the form of DMD needed for our analysis.

Let $p_{\text{data}}(x_0 \mid c)$ denote the target clean-data distribution represented by a frozen diffusion teacher, and let $q_\theta(x_0 \mid c)$ denote the distribution induced by the streaming generator $x_0 = G_\theta(z, c)$. At diffusion time t, samples from both distributions are perturbed by the same forward process,

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{5}$$

yielding the corresponding noised distributions $p_t(x_t \mid c)$ and $q_{\theta,t}(x_t \mid c)$. Distribution matching distillation (DMD) (Yin et al., 2024b) can be viewed as minimizing a time-averaged reverse-KL objective,

$$\mathcal{J}_{\text{DMD}}(\theta) = \mathbb{E}_{c,t}\big[\lambda(t)\, D_{\text{KL}}\big(q_{\theta,t}(\cdot \mid c) \,\|\, p_t(\cdot \mid c)\big)\big]\,. \tag{6}$$

Although this objective is defined over distributions, its generator update can be expressed through their scores. Denote the target and generator scores at time t by

$$s_p(x_t, t, c) = \nabla_{x_t} \log p_t(x_t \mid c), \qquad s_\theta(x_t, t, c) = \nabla_{x_t} \log q_{\theta,t}(x_t \mid c). \tag{7}$$

Both the population KL gradient and the implementation-weighted DMD update can be expressed through the difference between these two scores, pulled back through the generator. We write this common form as

$$g^*(\theta) = \mathbb{E}_\omega\big[a(\omega)\, J_\theta(\omega)^\top \big(s_\theta(x_t, t, c) - s_p(x_t, t, c)\big)\big]\,, \tag{8}$$

where $\omega = (z, c, t, \epsilon)$, $J_\theta(\omega) = \partial G_\theta(z, c)/\partial\theta$, and $a(\omega) \geq 0$ is a stop-gradient scalar coefficient. For the reverse-KL objective in Eq. (6), $a(\omega) = \lambda(t)\alpha_t$, recovering its exact pathwise gradient. Practical DMD implementations may additionally apply sample-dependent scalar normalization; we absorb such implementation-specific weighting into $a(\omega)$. The analysis below only requires that the same coefficient be used in the oracle and practical directions.

We call $g^*(\theta)$ the *oracle DMD direction* because it assumes access to the exact score s_θ of the current generator distribution. In practice, s_θ is unavailable and must be approximated by a learned fake-score model. Because $q_{\theta,t}$ changes whenever the generator is updated, this estimator must continually track a moving target. If it fails to keep pace, the practical DMD direction can deviate from its oracle counterpart. Section A.3 derives the reverse-KL special case of Eq. (8), and Section A.5 gives its exact correspondence to the practical DMD implementation.

A.2 Critic tracking error

The frozen diffusion teacher provides the target-side score, whereas the generator score s_θ in Eq. (8) is not directly available. DMD therefore introduces an auxiliary denoiser μ_ϕ, trained online on samples from the current generator, to estimate this score.

Under the x_0-prediction parameterization (Yin et al., 2024b), an x_0-prediction μ can be written in score form as

$$s_\mu(x_t, t, c) = -\frac{x_t - \alpha_t \mu(x_t, t, c)}{\sigma_t^2}. \tag{9}$$

We denote the frozen teacher prediction by μ_p. For the current generator distribution, the population-optimal denoiser is

$$\mu_\theta^*(x_t, t, c) = \mathbb{E}\big[x_0 \mid x_t, t, c; \theta\big], \tag{10}$$

where the expectation is taken under the joint distribution induced by the current generator and the forward noising process. Its score-form output is exactly the generator score s_θ.

In practice, μ_θ^* is approximated by the learned denoiser μ_ϕ, which is updated by denoising samples $x_0 = G_\theta(z, c)$ from the current generator, with x_0 treated as a stop-gradient target. We refer to μ_ϕ as the *critic* when discussing the generator–critic update schedule.

Because the generator distribution changes whenever θ is updated, the population target μ_θ^* changes as well. The critic generally receives only finitely many updates before the generator moves again and therefore need not reach this current optimum. Its estimated score

$$\hat{s}_\phi(x_t, t, c) = -\frac{x_t - \alpha_t \mu_\phi(x_t, t, c)}{\sigma_t^2}$$

can consequently differ from the current generator score. We define this discrepancy as the *tracking error*

$$e_{\theta,\phi}(x_t, t, c) = \hat{s}_\phi(x_t, t, c) - s_\theta(x_t, t, c). \tag{11}$$

Replacing the unavailable s_θ in Eq. (8) with $\hat{s}_\phi$, while keeping the same scalar weighting $a(\omega)$, yields the *practical DMD direction*

$$\hat{g}(\theta, \phi) = \mathbb{E}_\omega \left[a(\omega)\, J_\theta(\omega)^\top \left(\hat{s}_\phi(x_t, t, c) - s_p(x_t, t, c) \right) \right]. \tag{12}$$

The difference between the practical and oracle directions is derived in Section A.6, yielding the identity and bound in Eqs. (23)–(24). The score difference also gives the stop-gradient implementation of the DMD generator update, detailed in Section A.5.

A.3 First variation of the noised reverse KL

Fix t and consider an infinitesimal transport of a sample from $q_{\theta,t}$ by a vector field $v(x_t)$. If q_ε is the transported density, it obeys the continuity equation $\partial_\varepsilon q_\varepsilon|_{\varepsilon=0} = -\nabla \cdot (q_{\theta,t} v)$. Assuming sufficient smoothness and vanishing boundary terms, integration by parts gives

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\varepsilon} D_{\mathrm{KL}}(q_\varepsilon \| p_t)\Big|_{\varepsilon=0} &= -\int \nabla \cdot (q_{\theta,t} v) \left(\log \frac{q_{\theta,t}}{p_t} + 1 \right) \mathrm{d}x_t \\ &= \mathbb{E}_{x_t \sim q_{\theta,t}} \left[v(x_t)^\top \left(s_\theta(x_t, t) - s_p(x_t, t) \right) \right]. \end{aligned} \tag{13}$$

For the shared noising map $x_t = \alpha_t G_\theta(z, c) + \sigma_t \epsilon$, a parameter perturbation $\mathrm{d}\theta$ induces $\mathrm{d}x_t = \alpha_t J_\theta(z, c)\mathrm{d}\theta$. Substitution into Eq. (13), followed by averaging over c, t, z and ϵ, yields the standard DMD pathwise direction

$$\mathbb{E}\left[\lambda(t) \alpha_t J_\theta^\top (s_\theta - s_p) \right]. \tag{14}$$

This is the exact noised-KL case of Eq. (8), obtained with $a(\omega) = \lambda(t)\alpha_t$. The stopped normalization used in practice changes only this scalar coefficient; Section A.5 below gives the corresponding implementation formula.

A.4 Denoiser–score correspondence

Let $r_0(x_0 \mid c)$ be any clean distribution and let r_t be its convolution with the Gaussian kernel in Eq. (5). Differentiating under the integral gives

$$\begin{aligned} \nabla_{x_t} \log r_t(x_t \mid c) &= \mathbb{E}_{r_0(x_0 \mid x_t, c)} \left[\nabla_{x_t} \log \mathcal{N}(x_t; \alpha_t x_0, \sigma_t^2 I) \right] \\ &= -\frac{x_t - \alpha_t \mathbb{E}_{r_0}[x_0 \mid x_t, t, c]}{\sigma_t^2}. \end{aligned} \tag{15}$$

Hence the population x_0 predictor $\mu_r^*(x_t, t, c) = \mathbb{E}_{r_0}[x_0 \mid x_t, t, c]$ determines the score exactly. Applied to the current generator distribution, this gives μ_θ^* and s_θ in Eq. (9).

The online fake denoiser is trained on detached generator samples with an objective of the form

$$\mathcal{L}_{\mathrm{fake}}(\phi; \theta) = \mathbb{E}_{z,c,t,\epsilon} \left[\beta(t) \left\| \mu_\phi(x_t, t, c) - \mathrm{sg}(G_\theta(z, c)) \right\|_2^2 \right]. \tag{16}$$

For fixed θ, squared-error regression has conditional mean μ_θ^* as its population minimizer. The implementation can express the same denoising target in the equivalent flow parameterization; the network output used by the probe and DMD gradient is converted to x_0. Combining Eq. (15) for μ_ϕ and μ_θ^* shows

$$e_{\theta,\phi}(x_t, t, c) = \frac{\alpha_t}{\sigma_t^2} \left(\mu_\phi(x_t, t, c) - \mu_\theta^*(x_t, t, c) \right). \tag{17}$$

Thus critic tracking refers to the fake denoiser following a population regression target that moves with the generator, not to adversarial classification.

A.5 From score difference to the implemented pseudo-loss

Applying Eq. (9) to the fake and target distributions gives

$$\hat{s}_\phi - s_p = \frac{\alpha_t}{\sigma_t^2}(\mu_\phi - \mu_p). \tag{18}$$

LongLive absorbs the schedule-dependent scalar through gradient normalization. In simplified notation, the direction passed to the generator is

$$\Delta_\phi = \frac{\mu_\phi(x_t, t, c) - \mu_p(x_t, t, c)}{N(x_0, \mu_p)}, \qquad N(x_0, \mu_p) = \text{mean}\,|x_0 - \mu_p(x_t, t, c)|. \tag{19}$$

Equation (18) makes the corresponding implementation scalar in Eq. (8) explicit:

$$a_{\text{LL}}(\omega) = \frac{\sigma_t^2}{\alpha_t N(x_0, \mu_p)}, \qquad a_{\text{LL}}(\omega) J_\theta^\top(\hat{s}_\phi - s_p) = J_\theta^\top \Delta_\phi. \tag{20}$$

Here N and hence a_{LL} are treated as stop-gradient quantities. Shared scalar averaging factors can be absorbed into a_{LL}, while entry selection restricts J_θ and Δ_ϕ to the selected generator outputs. The implementation computes t, x_t, both denoiser predictions, N, and Δ_ϕ without gradient tracking. For d_G selected generator-output entries, its actual mean-squared pseudo-loss is

$$\widetilde{\mathcal{L}}_G = \frac{1}{2d_G} \left\| x_0 - \text{sg}(x_0 - \Delta_\phi) \right\|_2^2. \tag{21}$$

Although the forward value is merely a device for gradient injection, its derivative is exact. Treating the stopped target as constant,

$$\begin{aligned} \nabla_{x_0} \widetilde{\mathcal{L}}_G &= \frac{1}{d_G}\left(x_0 - \text{sg}(x_0 - \Delta_\phi)\right) = \frac{1}{d_G}\,\text{sg}(\Delta_\phi), \\ \nabla_\theta \widetilde{\mathcal{L}}_G &= \frac{1}{d_G} J_\theta^\top\,\text{sg}(\Delta_\phi). \end{aligned} \tag{22}$$

This is the code-level bridge from the DMD score field to a generator optimizer step. Classifier-free guidance and masking alter the particular denoiser combination or selected entries, but not the stop-gradient calculation.

A.6 Generator-direction error induced by critic tracking

Define the random linear map $A_\theta(\omega) = a(\omega) J_\theta(\omega)^\top$. At the same generator iterate, the target-score terms in Eqs. (8) and (12) cancel, giving

$$\hat{g} - g^* = \mathbb{E}_\omega[A_\theta(\omega) e_{\theta,\phi}(x_t, t, c)]. \tag{23}$$

Jensen's inequality, the definition of the operator norm, and Cauchy–Schwarz then give

$$\begin{aligned} \|\hat{g} - g^*\|_2 &\le \mathbb{E}\|A_\theta e_{\theta,\phi}\|_2 \le \mathbb{E}[\|A_\theta\|_{\text{op}} \|e_{\theta,\phi}\|_2] \\ &\le \left(\mathbb{E}\|A_\theta\|_{\text{op}}^2\right)^{1/2} \left(\mathbb{E}\|e_{\theta,\phi}\|_2^2\right)^{1/2}, \end{aligned} \tag{24}$$

This bound identifies critic tracking as one source of DMD direction error; it does not imply that the observable drift diagnostic directly estimates $e_{\theta,\phi}$. It also does not rule out teacher error, Monte Carlo noise, optimization error, or misspecification of the DMD surrogate.

B Local Tracking Analysis for Adaptive TTUR

We index training by generator opportunity n, the interval from one scheduled generator update to the next, and compare the fixed and adaptive schedules over one such cycle. The analysis is deliberately local and is *not intended as a global convergence guarantee*: neural DMD does not satisfy known global contraction conditions, and our experiments do not observe the population tracking error directly. We define

$$r_n = \left(\mathbb{E}\Big[\|e_{\theta_n,\phi_n}(x_t, t, c)\|_2^2\Big]\right)^{1/2} \tag{25}$$

as the root-mean-square tracking error at the beginning of cycle n.

B.1 Assumptions

Let

$$R(\theta, \phi) = \left(\mathbb{E}_{x_t \sim q_{\theta,t},t,c} \|\hat{s}_\phi(x_t, t, c) - s_\theta(x_t, t, c)\|_2^2\right)^{1/2}, \tag{26}$$

so $r_n = R(\theta_n, \phi_n)$. Let $\mathcal{C}_\theta^c(\phi)$ denote the critic parameters after c updates on samples from a fixed generator θ. We use three explicit assumptions.

A1: local critic contraction. For the iterates being compared, there is $\rho \in [0, 1)$ such that

$$R(\theta, \mathcal{C}_\theta^c(\phi)) \le \rho^c R(\theta, \phi) + \xi_n, \tag{27}$$

where $\xi_n \ge 0$ collects minibatch noise, finite optimization accuracy, and function-class approximation error over the interval.

A2: local target and measure stability. For fixed critic parameters and adjacent generator iterates,

$$R(\theta', \phi) \le R(\theta, \phi) + L_q \|\theta' - \theta\|_2. \tag{28}$$

This single inequality includes both movement of the oracle score and the change in the on-policy distribution under which the error is evaluated.

A3: bounded accepted generator move. Let $u_n \in \{0, 1\}$ indicate whether the offered generator update is executed, and let $\hat{g}_n = \hat{g}(\theta_n, \phi_n)$ denote the practical DMD direction at cycle n. At an offered generator update,

$$\|\delta\theta_n\|_2 \le \eta_g u_n \|\hat{g}_n\|_2, \qquad u_n \in \{0, 1\}. \tag{29}$$

This is a local abstraction of the optimizer step; clipping or preconditioning can be included by changing the right-hand norm and constant.

B.2 Tracking recursion and recovery analysis

Let the generator first move from θ_n to $\theta_{n+1} = \theta_n + \delta\theta_n$. Before the subsequent critic updates, A2 yields

$$R(\theta_{n+1}, \phi_n) \le r_n + L_q \|\delta\theta_n\|_2. \tag{30}$$

Now apply c_n critic steps to the new on-policy generator. A1 and Eq. (30) give

$$\begin{aligned} r_{n+1} &= R(\theta_{n+1}, \mathcal{C}_{\theta_{n+1}}^{c_n}(\phi_n)) \\ &\le \rho^{c_n}\left(r_n + L_q \|\delta\theta_n\|_2\right) + \xi_n, \end{aligned} \tag{31}$$

Substituting A3 gives

$$r_{n+1} \le \rho^{c_n}\left(r_n + L_q \eta_g u_n \|\hat{g}_n\|_2\right) + \xi_n. \tag{32}$$

This recursion separates two sources of tracking burden. The term $\rho^{c_n} r_n$ describes the residual error after the critic updates, whereas $\rho^{c_n} L_q \eta_g u_n \|\hat{g}_n\|_2$ captures the additional burden introduced by moving the generator. Thus, the amount of critic catch-up required to maintain a given tracking quality depends both on the current tracking state and on the size of the generator move.

A fixed TTUR keeps c_n constant regardless of either quantity. It therefore cannot allocate additional critic catch-up specifically when the tracking burden becomes large.

For the ordinary action, $u_n = 1$ and $c_n = H$, so the local bound is

$$B_n^{\text{ordinary}} = \rho^H\left(r_n + L_q \eta_g \|\hat{g}_n\|_2\right) + \xi_n. \tag{33}$$

For the recovery action, $u_n = 0$ and $c_n = H + K$, giving

$$B_n^{\text{recovery}} = \rho^{H+K} r_n + \xi_n. \tag{34}$$

Under A1–A3 and a common residual ξ_n, the recovery branch has a smaller modeled tracking bound because

$$B_n^{\text{ordinary}} - B_n^{\text{recovery}} = \rho^H(1 - \rho^K) r_n + \rho^H L_q \eta_g \|\hat{g}_n\|_2 \ge 0. \tag{35}$$

This is *not a claim that always recovering is optimal*: generator skipping also forgoes progress toward the target distribution, repeated decisions change the future trajectory, A1–A3 are only local, and the implemented trigger does not observe r_n. The comparison establishes the narrower mechanism claimed in the paper: skipping removes one source of target movement while additional critic steps increase the nominal catch-up factor.

B.3 Decomposition of the probe change

Suppressing (t^j, c^j) for readability, each temporal difference admits the exact decomposition

$$F_k^j - F_{k-1}^j = \underbrace{\mu_{\phi_k}(x_k^j) - \mu_{\phi_k}(x_{k-1}^j)}_{\text{generator-induced input change}} + \underbrace{\mu_{\phi_k}(x_{k-1}^j) - \mu_{\phi_{k-1}}(x_{k-1}^j)}_{\text{critic change}}. \tag{36}$$

B.4 Training algorithm

Algorithm 1 Drift-Gated TTUR

Require: Generator G_θ, critic μ_ϕ, probe bank $\mathcal{P}$, generator/probe interval H, calibration length M, threshold multiplier κ, recovery budget K

$\mathcal{D} \leftarrow \emptyset$, $\tau \leftarrow$ unset, $\bar{D} \leftarrow$ none
$\{F_{\text{prev}}^j\}_{j=1}^m \leftarrow$ none
for outer step $s = 1, 2, \ldots$ **do**
  recover $\leftarrow$ false
  if s is a scheduled generator opportunity **then**
    if τ is set **and** $\bar{D} > \tau$ **then**
      Skip generator update; recover $\leftarrow$ true
    else
      Update generator θ
    end if
  end if
  Perform one normal critic update
  if s is a probe step **then**
    Regenerate x^j using Eq. (1); $F^j \leftarrow \mu_\phi(x^j, t^j, c^j)$ for $j = 1, \ldots, m$
    if previous probe outputs exist **then**
      $D \leftarrow \frac{1}{md} \sum_{j=1}^m \|F^j - F_{\text{prev}}^j\|_2^2$; append D to $\mathcal{D}$; $\bar{D} \leftarrow D$
    end if
    $F_{\text{prev}}^j \leftarrow F^j$ for $j = 1, \ldots, m$
    if τ is unset **and** $|\mathcal{D}| = M$ **then**
      $\tau \leftarrow \kappa \,\text{median}(\mathcal{D})$; freeze τ
    end if
  end if
  if recover **then**
    Perform K additional critic recovery updates
  end if
end for