ADVERSARIAL GRADIENT FORCING FOR ROBUST FEW-STEP AUTOREGRESSIVE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive video generation models generate videos sequentially, with each new block conditioned on previously generated blocks. By training on the model's own rollouts, Self Forcing and its follow-up methods substantially mitigate exposure bias caused by the mismatch between training- and inference-time history distributions. However, these models still suffer from error accumulation during long-horizon generation. We identify a key underlying limitation: the model remains highly sensitive to errors in its generated history, allowing small deviations to be amplified through subsequent generations. We refer to this underexplored problem as the local history robustness gap. To address this gap, we introduce Adversarial Gradient Forcing (AGF), a framework designed to improve the robustness of few-step autoregressive video models to perturbations and accumulated errors in their own generated histories. AGF uses distribution matching gradients from future block predictions to select adversarial perturbations of self-generated histories. It then jointly trains the generator on the original and perturbed histories to improve the robustness of future generation. Extensive experiments show that AGF achieves the highest VBench Total score on 60-second video generation and the best visual quality on four-minute video generation. Notably, AGF achieves such long-horizon robustness while requiring only 5-second rollouts for training.

1 INTRODUCTION

Video generation has progressed from early diffusion-based generative models (Ho et al., 2022) to large-scale systems capable of synthesizing high-quality, temporally coherent videos (Blattmann et al., 2023; Chen et al., 2024b; Yang et al., 2024; Kong et al., 2024; Team Wan et al., 2025). Autoregressive approaches further extend video generation by producing successive video blocks conditioned on previously generated blocks (Henschel et al., 2025; Meituan LongCat Team et al., 2025; Gao et al., 2024). Recent methods further combine autogressive modeling with few-step denoising to reduce inference latency (Yin et al., 2025; Lin et al., 2025; Zhu et al., 2026), enabling streaming and interactive video generation (Yang et al., 2026; Shin et al., 2026; Savva et al., 2026). Among these methods, self forcing further narrows the training–inference gap by optimizing on self-generated rollouts with Distribution Matching Distillation (DMD) (Huang et al., 2025a; Yin et al., 2024b;a), and subsequent methods extend this paradigm through longer training trajectories and additional supervision (Cui et al., 2025; Yu et al., 2026; Zhuang et al., 2026; Liu et al., 2026). Together, these methods aim to make autoregressive video generation more efficient and reliable over long horizons.

However, current autoregressive video generation models still suffer from the notorious error accumulation problem during long-horizon generation (Yang et al., 2026; Liu et al., 2026). Since each generated block becomes part of the conditioning history, errors in earlier blocks can propagate through subsequent generation and progressively degrade visual quality (Huang et al., 2025a; Yang et al., 2026; Li et al., 2025). We identify a key cause of error accumulation: the model remains highly sensitive to errors in its self-generated history. In Section 3.1, we present details on the error sensitivity and refer to it as the *local history robustness gap*. We further show that improving robustness to these errors substantially mitigates error accumulation and stabilize long-horizon generation.

To address this gap, we introduce *Adversarial Gradient Forcing* (AGF), a novel training framework for improving long-horizon visual quality in few-step autoregressive video generation. Rather than

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.



Figure 1: **An example of four-minute video generation.** Self Forcing and AGF generate four-minute videos using the same ODE initialization. With Self Forcing, visual artifacts emerge and progressively worsen over time, while AGF maintains coherent subjects and scene structure.

merely training on the model's own rollouts, AGF deliberately constructs challenging histories according to how strongly they impair the subsequent generation. By explicitly exposing the model to such challenging histories during training, AGF improves the robustness against accumulated errors, and suppresses their propagation into subsequent blocks. More specifically, AGF backpropagates the DMD gradient from future rollout blocks to a randomly selected history block, and uses the resulting gradient to construct a bounded adversarial perturbation to the selected history block.

Given the adversarial perturbation, AGF then recomputes block predictions under both the original and perturbed histories and jointly optimizes the generator with DMD. The original-history branch preserves performance under the model's standard generation conditions, while the perturbed-history branch improves robustness to errors in the generated history. DMD is evaluated independently on the rollout produced by each branch, yielding the training gradients for joint optimization.

We further introduce a *prediction consistency regularizer* during early training to prevent adversarial perturbations from driving the generated trajectory too far from the local data manifold. This regularization term further improves consistency in viewpoint and scene structure.

We conduct extensive experiments to evaluate AGF. Despite being trained only on five-second rollouts, AGF achieves the highest 60-second VBench Total score and the best four-minute visual quality among all five evaluated baselines, while remaining competitive on five-second generation. More importantly, AGF sustains stable visual quality throughout four-minute generation, whereas all baselines progressively degrade as generation proceeds (Section 4.3), as illustrated in Figure 1.

Our main contributions are summarized as follow:

- We analyze how small errors in generated histories propagate to subsequent generation and identify the local history robustness gap in existing methods.
- We propose AGF, which selects adversarial history perturbations using gradients from future and incorporates them into on-policy distribution matching to improve robustness to history errors.
- Extensive experiments demonstrate that AGF improves visual quality and reduces degradation during long video generation. Ablation studies validate the effectiveness of each component.

2 Related Work

2.1 Training Few-Step Autoregressive Video Models

Early approaches differ primarily in how they construct the histories used for autoregressive generation. Teacher Forcing conditions on ground-truth histories (Bengio et al., 2015; Lamb et al., 2016), whereas Diffusion Forcing relaxes this dependency by independently corrupting video frames with noise (Chen et al., 2024a). CausVid then combines Diffusion Forcing with Distribution Matching Distillation (DMD) to enable few-step autogressive generation (Yin et al., 2025; 2024b;a). Self Forcing takes a novel direction by training the model on its own autoregressive rollouts, thereby reducing the mismatch between training- and inference-time history distributions (Huang et al., 2025a). AAPT further introduces discriminator-based supervision over generated histories (Lin et al., 2025). Subsequent methods built on this paradigm propose improved causal initialization (Zhu et al., 2026;

Zhao et al., 2026; Zheng et al., 2026), longer self-generated training trajectories (Yang et al., 2026; Cui et al., 2025; Chen et al., 2026; Luo et al., 2026), joint denoising across adjacent blocks (Liu et al., 2025; Li et al., 2026), and gradient propagation through recomputed history representations (Savva et al., 2026; Zhuang et al., 2026). Despite these improvements, robustness against errors already present in generated histories remains underexplored. The proposed AGF targets this problem by improving the model's tolerance to imperfect self-generated histories, thereby mitigating error accumulation over long-horizon generation while requiring only short training rollouts.

2.2 Training with Imperfect Histories

Recent studies have begun to investigate autoregressive video generation under imperfect history conditions. GameNGen injects noise into conditioning frames, while Helios applies predefined frame corruptions to improve long-horizon video generation (Valevski et al., 2024; Yuan et al., 2026). Resampling Forcing constructs imperfect histories by self-resampling real videos, Stable Video Infinity reuses prediction errors from previous generations, and BAgger builds training trajectories from progressively drifting rollouts (Guo et al., 2025; Li et al., 2025; Po et al., 2025). Salt further diversifies history quality through cache perturbations and feature alignment, whereas OPSD-V leverages real-video context to provide teacher supervision (Ge et al., 2026; Liu et al., 2026). Despite these efforts, none of these methods explicitly assess how individual history perturbations affect downstream generations and use this impact to select informative training histories.

Adversarial training improves robustness by constructing input perturbations that increase the training loss (Szegedy et al., 2014; Goodfellow et al., 2015; Madry et al., 2018). BadWorld studies adversarial perturbations to the conditioning context of fixed world models (Shen et al., 2026). In contrast, AGF introduces adversarial history selection into on-policy distribution matching, using the perturbed histories to train the model for improved robustness.

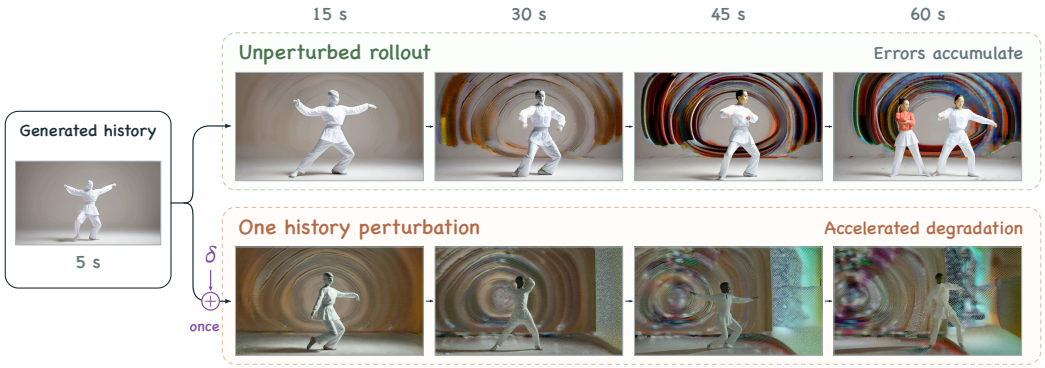


Figure 2: **A single perturbation leads to severe visual degradation in long-horizon autogressive video generation.** The top row shows an unperturbed Self-Forcing rollout, while the bottom row uses the same model and identical sampling randomness for future generation, differing only by a one-time perturbation applied to the generated history. Compared with the unperturbed rollout, concentric background artifacts emerge earlier and progressively intensify in the perturbed rollout, accompanied by increasingly severe subject distortion.

3 Method

In this section, we first introduce the local history robustness gap, where small errors in generated histories can lead to catastrophic visual degradation in subsequent generation. Building on this observation, we then present AGF, including its adversarial perturbation selection strategy and training pipeline. We further introduce a regularization term that encourages the adversarial perturbations to remain close to the generator's data manifold. Finally, we visualize the learned adversarial perturbations and analyze their effects on both generated histories and subsequent generation.

3.1 THE LOCAL HISTORY ROBUSTNESS GAP

Autoregressive video generation models condition each new block on previously generated ones, making generation quality highly sensitive to errors accumulated in the history. Even a small perturbation to a generated history block can propagate forward and substantially degrade subsequent generation. As illustrated in Figure 2, perturbing a single history block causes visual artifacts to emerge earlier and become progressively more severe compared with the unperturbed rollout. We refer to the model's vulnerability to small history errors that degrade future generation as the *local history robustness gap*.

Motivated by this observation, AGF aims to identify perturbations on the generated history that maximally impair subsequent generation, and trains the model to produce high-quality continuations from both clean and perturbed histories. In Appendix A.1, we provide a theoretical analysis of the error propagation mechanism, showing that perturbations introduced into the history and errors accumulated during autoregressive generation share the same underlying mechanism.

3.2 ADVERSARIAL GRADIENT FORCING

Rather than selecting the perturbation that most changes future predictions, AGF selects the one that most increases the DMD loss on future blocks. High sensitivity to history does not by itself imply a distribution-matching error: a generator can respond strongly to its history while its continuations remain correctly distributed (Appendix A.2).

DMD trains a few-step autoregressive video generator, parameterized by θ, to match a teacher-defined target distribution by minimizing the reverse KL divergence. At noise level τ, the frozen teacher estimates the score s_{real} of the noisy target distribution, while a fake-score model, trained online on samples from the generator, estimates the score s_{fake} of the noisy generator-induced distribution. Given a generated video $X_0 = (x_0^{(1)}, \ldots, x_0^{(K)})$ represented by K clean latent blocks, DMD perturbs it with Gaussian noise to obtain X_τ, evaluates both scores at X_τ, and uses their difference to construct the distribution-matching gradient $g_{\text{DM}}(X_0)$:

$$g_{\text{DM}}(X_0) \propto s_{\text{fake}}(X_\tau, \tau) - s_{\text{real}}(X_\tau, \tau) \propto \widehat{X}_{0,\text{fake}(X_\tau,\tau)} - \widehat{X}_{0,\text{real}}(X_\tau, \tau). \tag{1}$$

Here, $\widehat{X}_{0,\text{real}}$ and $\widehat{X}_{0,\text{fake}}$ are the two models' predictions. DMD backpropagates $g_{\text{DM}}(X_0)$ through the generator to update θ by gradient descent. Further details are available in Appendix B.1.

(a) Sample autoregressive rollout trajectories. To supervise different denoising stages, AGF first samples a generator timestep $t^\star$ for each rollout and uses the same timestep across all K blocks. It then performs an autoregressive rollout without retaining the computation graph. For each block k, the generator takes the noisy latent block $\bar{x}_{t^\star}^{(k)}$ as input and predicts the corresponding clean latent block $\bar{x}_0^{(k)}$. Rollout trajectories are recorded as $\bar{X}_{t^\star} = (\bar{x}_{t^\star}^{(1)}, \ldots, \bar{x}_{t^\star}^{(K)})$ and $\bar{X}_0 = (\bar{x}_0^{(1)}, \ldots, \bar{x}_0^{(K)})$ for use in the following stages. AGF then randomly selects a block $j < K$ to perturb.

(b) Select an adversarial history perturbation. To find a small perturbation to the selected block j that most adversely affects subsequent generation, AGF then adds a zero-initialized perturbation δ on the clean latent block $\bar{x}_0^{(j)}$ to probe how changes to this block affect subsequent generation.

Given the offline rollout trajectory $\bar{X}_{t^\star}$, we recompute the clean latent predictions in parallel for all blocks $k > j$ with computation graph enabled. Specifically, the generator first encodes the recorded clean latent blocks into key–value (K/V) representations, with the perturbation δ added to $\bar{x}_0^{(j)}$. Then, for each block $k > j$, the generator takes the recorded noisy latent $\bar{x}_{t^\star}^{(k)}$ as input and conditions on the K/V representations constructed from the preceding latent blocks, where block j is perturbed by δ. We denote the resulting predictions for all blocks after j by $\widehat{Y}_j(\delta)$. This forward pass preserves a differentiable path from $\widehat{Y}_j(\delta)$ back to δ (Appendix B.2).

Next, we compute the distribution–matching gradient on the full rollouts, and backpropagate only its entries $\bar{g}_j$ from blocks after j to δ, obtaining the history gradient u_j, along which the DMD loss

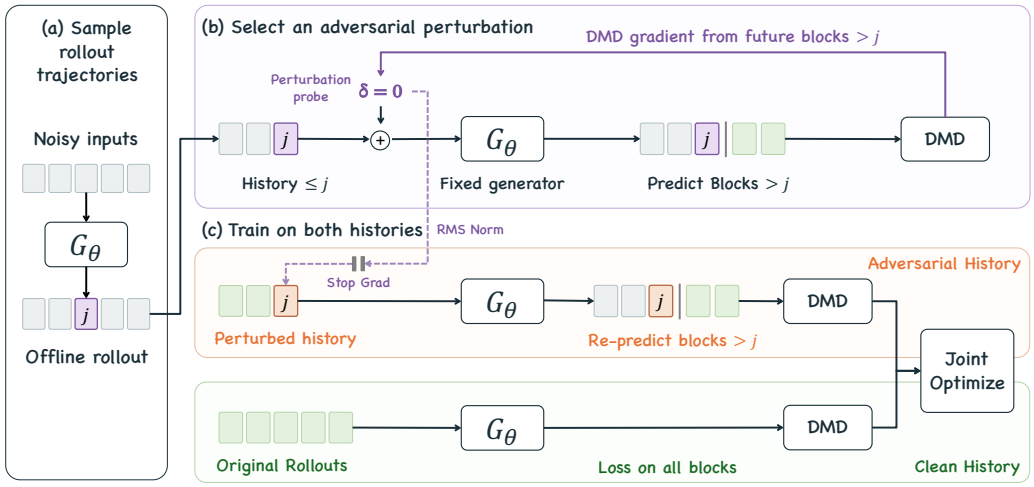


Figure 3: **Adversarial Gradient Forcing.** (a) Sample autoregressive rollout trajectories offline without gradient. (b) Use DMD gradients from blocks after j to select a history perturbation that most adversely affects subsequent generations. (c) With the perturbation fixed, train jointly on the clean and perturbed histories. G_θ denotes the generator in DMD parameterized by θ.

increases most rapidly. This backpropagation gives $u_j = J_j^\top \bar{g}_j$, where J_j is the Jacobian of the later predictions $\widehat{Y}_j(\delta)$ with respect to δ, evaluated at $\delta = 0$ (Appendix A.3).

The history gradient u_j may differ across the latent frames in block j. To obtain a shared direction $\widetilde{u}_j$, we average u_j across the latent frame dimension and use the resulting mean as the perturbation direction for every frame in block j. This shared direction $\widetilde{u}_j$ is then rescaled to a root-mean-square (RMS) magnitude of ϵ, giving $\delta_j^\star = \mathrm{sg}(\epsilon\, \widetilde{u}_j / \mathrm{RMS}(\widetilde{u}_j))$, where ϵ controls the perturbation strength. The resulting $\delta_j^\star$ is the adversarial history perturbation selected for training (Appendix A.4). The operator sg stops gradients through perturbations, keeping $\delta_j^\star$ fixed during generator training.

(c) Train on both histories. After adding the selected adversarial perturbation $\delta_j^\star$ to the recorded clean latent block $\bar{x}_0^{(j)}$, we then recompute denoise predictions from $\bar{X}_{t^\star}$ at $t^\star$ under both the clean and perturbed histories. The clean and adversarial branches use the original and perturbed histories to produce full video predictions X_0^{clean} and X_0^{adv}, respectively.

We evaluate DMD on each full rollout to obtain the corresponding gradients g^{clean} and g^{adv}. The surrogate loss $\ell_{\mathrm{DM}}(X_0; g)$ applies the fixed DMD gradient g to the prediction X_0 (Appendix B.1). We train the generator with the joint learning objective:

$$\mathcal{L}_{\mathrm{AGF}} = \ell_{\mathrm{DM}}(X_0^{\mathrm{clean}}; g^{\mathrm{clean}}) + \ell_{\mathrm{DM}}^{>j}(X_0^{\mathrm{adv}}; g^{\mathrm{adv}}). \tag{2}$$

The clean loss branch ℓ_{DM} averages over all blocks within one rollout, while the adversarial loss branch $\ell_{\mathrm{DM}}^{>j}$ only averages over blocks after the perturbed block j, focusing training on predictions conditioned on the perturbed history. The fake-score model in DMD is also trained on a weighted mixture of samples from both branches (Appendix B.3). Algorithm 1 in Appendix B.5 summarizes the complete training procedure.

3.3 Prediction Consistency Regularizer

To further improve consistency between the clean and adversarial branches, we introduce a *prediction consistency regularizer* that aligns velocity predictions of the generator at the first denoising step. Enforcing consistency at this initial stage encourages the two branches to preserve similar global scene structures, while allowing subsequent denoising steps to independently refine local appearance and fine-grained details.

Let $v_{>j}^{\text{clean}}$ and $v_{>j}^{\text{adv}}$ denote generator's predicted velocity for all blocks after j at the first denoising step. The regularization loss is written as:

$$\mathcal{L}_{\text{reg}} = \text{MSE}\big(v_{>j}^{\text{adv}}, \text{sg}(v_{>j}^{\text{clean}})\big)\,. \tag{3}$$

$\mathcal{L}_{\text{reg}}$ averages over all blocks after j with the prediction from the clean branch detached, such that the regularizer only affects the generator through the adversarial branch.

During the early training stage, we add this regularizer to the AGF learning objective to encourage consistency: $\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{AGF}} + \lambda_s \mathcal{L}_{\text{reg}}$, where λ_s is the regularization weight at training iteration s. We then remove the regularizer and continue optimizing $\mathcal{L}_{\text{AGF}}$ alone to further improve visual quality. Appendix B.4 provides details on the regularization weight scheduler. Section 4.6 conducts ablation studies on different choices for the regularizer.

3.4 Visualizing the Effects of History Perturbations in AGF

The perturbations selected by AGF tend to target vulnerable scene features in pixel space, including fine textures, subtle color variations, and complex structures. By training on these perturbed histories, AGF directs the model's attention toward fine-grained details and complex structures whose corruption has a particularly detrimental effect on subsequent generation, thereby improving the generator's ability to preserve these features over long rollouts.

Figure 4 compares the effects of AGF-selected perturbations with those of Gaussian noise in pixel space, with both applied to the same history block. As shown, AGF perturbations selectively affect vulnerable scene details, particularly fine hull textures, rigging around the masts, and sail boundaries. Moreover, the second-to-last subfigure shows that the influence of perturbations from AGF remains pronounced in later generated frames, indicating their persistent effect on subsequent generation. In contrast, Gaussian noise induces more spatially uniform changes in the history block and results in substantially weaker deviations in later frames. In addition to this example, Appendix C provides a more comprehensive analysis of this behavior using additional cases.



Figure 4: **Effects of selected and Gaussian history perturbations.** The leftmost clean history serves as the scene reference. *History difference* shows the pixel difference between corresponding frames decoded from the perturbed and original histories. *Later-frame difference* shows the pixel difference between corresponding later frames generated from two separate histories.

4 Experiments

4.1 Experimental Setup

Training. We train a four-step causal video generator based on Wan2.1-T2V-1.3B (Team Wan et al., 2025), initialized from the ODE initialization checkpoint released by Self-Forcing (Huang et al., 2025a). We use Wan2.1-T2V-14B as the frozen teacher and train on the filtered and LLM-extended VidProM (Wang & Yang, 2024) text prompts same as Self-Forcing. Training uses five-second rollouts with an effective batch size of 64, consisting of 200 iterations of AGF with prediction consistency regularization followed by 200 iterations without regularization. All experiments are conducted on eight NVIDIA H20 GPUs. Full training details are available in Appendix D.

Evaluation. 16 VBench dimensions (Huang et al., 2023) on the full 944-prompt suite at 5 seconds, and a randomly sampled 240-prompt subset at 60 seconds are evaluated. At 240 seconds, we further use 120 prompts randomly sampled from MovieGenVideoBench_extended (Polyak et al., 2024).

We report VBench Total at 5 seconds and Quality, Semantic, and Total at 60 seconds. At 240 seconds, we evaluate the six VBench-Long dimensions supported for custom prompts (Huang et al., 2025b). For visual quality, we follow LongLive (Yang et al., 2026) and use five VBench-Long dimensions: subject consistency, background consistency, motion smoothness, aesthetic quality, and imaging quality. We define **Five-metric Quality (Q5)** as their average results and use it as our primary quality score. Full protocols and checkpoint details appear in Appendix D.

Baselines. We compare against Self-Forcing (Huang et al., 2025a), Causal Forcing (Zhu et al., 2026), LongLive (Yang et al., 2026), Self Gradient Forcing (SGF) (Zhuang et al., 2026), and OPSD-V (Liu et al., 2026). For OPSD-V, we evaluate the official checkpoint post-trained from Self-Forcing. We reproduce SGF from the same ODE initialization as AGF, using SGF's official training settings. All methods adopt the same inference settings, with 3 sink and 9 rolling latent frames.

Table 1: **Quantitative comparison with relevant baselines.** We benchmark all methods from native 5-second generation to 240-second long-horizon generation. Except for Dynamic Degree, orange and blue indicate the best and second-best results, respectively; the best results are bolded. Higher Dynamic Degree may also arise from severe camera drift or visual corruption and does not necessarily indicate better generation quality.

	5 s	60 s			240 s						
Method	Total↑	Quality↑	Semantic↑	Total↑	Subj.↑	Back.↑	Motion↑	Aesth.↑	Imaging↑	Dyn.†	**Q5↑**
Self-Forcing	82.21	82.95	66.01	79.56	96.36	95.79	97.95	54.97	67.26	63.19	82.47
Causal Forcing	**82.94**	83.82	63.88	79.83	94.44	94.82	95.99	55.29	68.41	91.41	81.79
LongLive	80.34	82.11	**70.59**	79.81	97.08	96.24	**98.60**	58.29	67.86	49.53	83.62
SGF	82.08	83.56	64.09	79.67	97.11	96.10	98.18	58.32	70.63	51.08	84.07
OPSD-V	81.95	83.16	67.67	80.06	96.29	95.64	97.91	56.60	67.77	65.10	82.84
AGF (ours)	82.03	**84.67**	68.66	**81.47**	**97.32**	**96.38**	98.31	**61.20**	**70.91**	62.54	**84.82**

4.2 Main Long-Horizon Benchmark

Table 1 shows that AGF maintains comparable short-horizon performance while achieving state-of-the-art results at longer horizons. At five seconds, AGF performs comparably to Self-Forcing, indicating that adversarial training does not compromise short-horizon generation quality. By 60 seconds, AGF already outperforms the baselines, achieving the best Quality and Total scores among all methods. This advantage becomes more pronounced at 240 seconds: AGF ranks first in four of the five quality dimensions and achieves the highest Q5 score of 84.82. Complete per-dimension results are provided in Appendix E. These results indicate that AGF primarily improves the generator's ability to preserve visual quality as autoregressive errors accumulate over long rollouts.

Figure 5 presents a representative long-horizon comparison on a coastal church scene. As generation proceeds, the competing methods variously suffer from subject loss, compositional drift, or structured artifacts, whereas AGF preserves both the church and coastline through the final frame. Appendix F.4 provides additional comparisons on more diverse scenes.

Additional visual comparisons reveal failure modes not fully captured by benchmark scores. In Causal Forcing and several other methods, high Dynamic Degree often coincides with camera drift or accumulating artifacts, showing that stronger motion does not ensure better long-horizon quality (Appendix F.1). Meanwhile, AGF achieves high Dynamic Degree in genuinely dynamic scenes, such as clay sculpting, salad tossing, and character movement, while preserving the subjects and surrounding scenes (Appendix F.2). For Q5, in some scenes, methods with Q5 scores comparable to or higher than AGF's still exhibit subject duplication or background drift (Appendix F.3).

4.3 Stability over Rollout Time

We next evaluate the stability of visual quality throughout four-minute generation. Building on the evaluation in Section 4.2, we track Q5 over consecutive 20-second intervals and report its change relative to the first interval. Figure 6 shows that AGF maintains stable visual quality throughout the rollout, with a first-to-last change of $+0.03$ pp, whereas all five baselines decline by 0.96–2.61 pp. Detailed results for all methods are provided in Appendix G.1, while Appendix G.2 further demonstrates AGF's stronger robustness under two temporary history perturbations.



Figure 5: **Long-horizon qualitative comparison.** Six-method rollouts on the coastal church scene at 0, 48, 96, 144, 192, and 240 seconds. AGF better preserves the subject and scene composition.

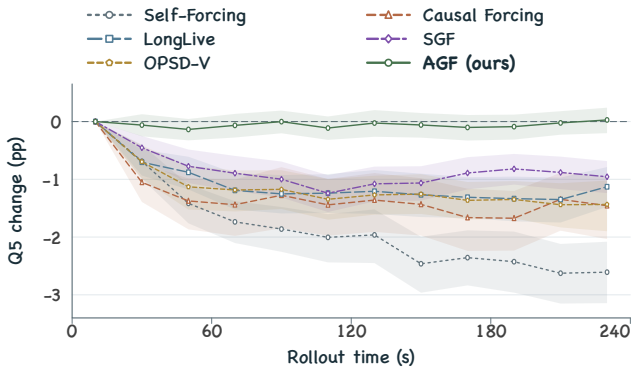


Figure 6: **Visual quality over time.** Local Q5 change from the first 20 seconds, averaged over 120 prompts; shading shows pointwise 95% bootstrap confidence intervals.

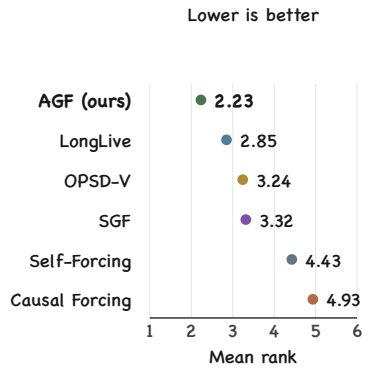


Figure 7: **Human ranking.** Mean ranks from 240 tasks covering 120 prompts (lower is better).

4.4 Human Evaluation

Twelve participants each ranked 20 sets of six anonymized four-minute videos according to their overall preference. AGF achieved the best mean rank of 2.23, followed by LongLive at 2.85 (Figure 7). AGF also received the highest proportion of first-place rankings (35.8%) and was ranked among the top three in 84.2% of evaluations. Appendix I provides a more detailed analysis together with the full rank distributions. Overall, these results show that AGF is consistently preferred by human evaluators for long-horizon video generation.

4.5 Training Cost

Although AGF adds perturbation selection and paired clean/adversarial training, its short schedule and five-second rollouts keep its training cost lower than most baselines. Starting from the ODE

initialization, AGF requires 400 outer iterations, including 80 generator updates, for a total of 364.4 GPU-hours on eight H20 GPUs.

For comparison, Self-Forcing requires 600 outer iterations and 431.6 GPU-hours. LongLive performs 3,000 iterations of long-video tuning with 60-second streaming sequences, requiring 2774.4 GPU-hours. OPSD-V uses approximately 45-second rollouts and supervises all four denoising steps after the warm-up blocks. Repeated student and teacher evaluations along these long rollouts raise its training cost to 944.4 GPU-hours. OPSD-V additionally requires real long-video data, whereas AGF post-training uses only text prompts and self-generated videos. Full training configurations and cost accounting are provided in Appendix D.

4.6 Ablation Studies

Table 2: **Ablation studies of AGF.** Panel (a) studies perturbation and regularization strategies, while panel (b) evaluates AGF under a different ODE initialization. Best scores in each row are bolded.

(a) Perturbation and regularization

	Default	Perturbation	Regularizer	Regularized steps			Reference model	
Duration	**AGF**	Gaussian	None	1–2	1–3	1–4	ODE init	Teacher
5 s	82.03	71.09	81.38	77.75	**82.06**	74.30	78.79	80.78
60 s	**81.47**	68.60	77.96	76.24	80.40	72.77	75.63	79.74
240 s	**84.82**	75.11	82.75	79.87	83.94	78.19	79.72	82.82

(b) ODE initialization

	Model	
Duration	CF	$\mathrm{AGF}_{\mathrm{CF}}$
5 s	**82.94**	81.74
60 s	79.83	**81.00**
240 s	81.79	**84.60**

Adversarial perturbations. We compare the selected history perturbations with Gaussian perturbations of the same RMS magnitude. Adversarial selection improves 240-second Q5 by 9.71 pp, demonstrating the benefit of using gradients to select adversarial perturbations (Table 2(a)).

Regularization strategies. The prediction consistency regularizer is an integral component of AGF rather than an optional add-on. Removing it lowers 240-second Q5 from 84.82 to 82.75 and 60-second Total from 81.47 to 77.96. We attribute this drop to that without an anchor to the clean-history prediction, the adversarial branch can shift the global scene layout. Nevertheless, the unregularized variant still improves 240-second Q5 over Self-Forcing (82.47) and matches OPSD-V (82.84) with only 80 generator updates. This indicates that adversarial history selection alone already benefits long-horizon quality, while the regularizer is needed to preserve scene consistency.

We then study effects of the regularizer beyond the first denoising step. Regularizing the first two, three, or all four steps lowers 60-second Total and 240-second Q5, supporting the choice to guide the initial prediction while leaving later steps free to refine details.

We then compare two alternative regularizers that use frozen reference models to guide the perturbed branch. Specifically, the first denoising step is regularized toward either the prediction of the fixed ODE-initialized generator or an equivalent first-step velocity target derived from the frozen diffusion teacher. Both regularizers underperform the proposed regularizer at all three durations (Table 2(a)). Appendix H presents details on their formulation.

ODE initialization. To verify the effectiveness of AGF across different initialization strategies, we train $\mathrm{AGF}_{\mathrm{CF}}$ from Causal Forcing's Stage 2 ODE checkpoint. It outperforms the released Causal Forcing model by 2.81 pp in 240-second Q5 (Table 2(b)). Training details appear in Appendix H.2.

5 Conclusion

Our work investigates long-horizon degradation in few-step autoregressive video generation and identifies the model's sensitivity to errors in its self-generated history as a key source of cumulative quality decay. Building on this observation, we introduce Adversarial Gradient Forcing (AGF), which constructs history perturbations that are particularly detrimental to subsequent generations and explicitly trains the model to remain robust to them. Despite being trained only on five-second rollouts, AGF sustains stable visual quality throughout four-minute generation.

AI USE STATEMENT

Generative AI tools were used to assist with code development, drafting and polishing selected parts of the manuscript, and preparing preliminary versions of several illustrative figures. The authors independently conceived the research problem, core ideas, and main methodology of this work. All AI-assisted outputs were carefully reviewed, revised, and verified by the authors, who take full responsibility for the technical content, experimental results, and final manuscript.

REFERENCES

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv preprint arXiv:1506.03099*, 2015. URL https://arxiv.org/abs/1506.03099.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. URL https://arxiv.org/abs/2311.15127.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *arXiv preprint arXiv:2407.01392*, 2024a. URL https://arxiv.org/abs/2407.01392.

Haoxin Chen, Yong Zhang, Xiaodong Cun, Menghan Xia, Xintao Wang, Chao Weng, and Ying Shan. VideoCrafter2: Overcoming data limitations for high-quality video diffusion models. *arXiv preprint arXiv:2401.09047*, 2024b. URL https://arxiv.org/abs/2401.09047.

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.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-Forcing++: Towards minute-scale high-quality video generation. *arXiv preprint arXiv:2510.02283*, 2025. URL https://arxiv.org/abs/2510.02283.

Kaifeng Gao, Jiaxin Shi, Hanwang Zhang, Chunping Wang, Jun Xiao, and Long Chen. Ca2-VDM: Efficient autoregressive video diffusion model with causal generation and cache sharing. *arXiv preprint arXiv:2411.16375*, 2024. URL https://arxiv.org/abs/2411.16375.

Xingtong Ge, Yi Zhang, Yushi Huang, Dailan He, Xiahong Wang, Bingqi Ma, Guanglu Song, Yu Liu, and Jun Zhang. Salt: Self-consistent distribution matching with cache-aware training for fast video generation. *arXiv preprint arXiv:2604.03118*, 2026. URL https://arxiv.org/abs/2604.03118.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2015. URL https://arxiv.org/abs/1412.6572.

Yuwei Guo, Ceyuan Yang, Hao He, Yang Zhao, Meng Wei, Zhenheng Yang, Weilin Huang, and Dahua Lin. End-to-end training for autoregressive video diffusion via self-resampling. *arXiv preprint arXiv:2512.15702*, 2025. URL https://arxiv.org/abs/2512.15702.

Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. StreamingT2V: Consistent, dynamic, and extendable long video generation from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Henschel_StreamingT2V_Consistent_Dynamic_and_Extendable_Long_Video_Generation_from_Text_CVPR_2025_paper.html.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. URL https://arxiv.org/abs/2204.03458.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025a. URL https://arxiv.org/abs/2506.08009.

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. *arXiv preprint arXiv:2311.17982*, 2023. URL https://arxiv.org/abs/2311.17982.

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. URL https://arxiv.org/abs/2411.13503.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

Alex M. Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. *arXiv preprint arXiv:1610.09038*, 2016. URL https://arxiv.org/abs/1610.09038.

Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. *arXiv preprint arXiv:2510.09212*, 2025. URL https://arxiv.org/abs/2510.09212.

Zekun Li, Xiaoyan Cong, Hongyu Li, Zhiyang Dou, Chuan Guo, Abhay Mittal, Sizhe An, and Srinath Sridhar. Ms. Forcing: Efficient streaming video generation with multi-scale patchification and attention. *arXiv preprint arXiv:2607.20940*, 2026. URL https://arxiv.org/abs/2607.20940.

Shanchuan Lin, Ceyuan Yang, Hao He, Jianwen Jiang, Yuxi Ren, Xin Xia, Yang Zhao, Xuefeng Xiao, and Lu Jiang. Autoregressive adversarial post-training for real-time interactive video generation. *arXiv preprint arXiv:2506.09350*, 2025. URL https://arxiv.org/abs/2506.09350.

Hongyu Liu, Chun Wang, Feng Gao, Xuanhua He, Yue Ma, Ziyu Wan, Yong Zhang, Xiaoming Wei, and Qifeng Chen. OPSD-V: On-policy self-distillation for post-training few-step autoregressive video generators. *arXiv preprint arXiv:2607.08766*, 2026. URL https://arxiv.org/abs/2607.08766.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. *arXiv preprint arXiv:2509.25161*, 2025. URL https://arxiv.org/abs/2509.25161.

Yawen Luo, Xiaoyu Shi, Junhao Zhuang, Yutian Chen, Quande Liu, Xintao Wang, Pengfei Wan, and Tianfan Xue. ShotStream: Streaming multi-shot video generation for interactive storytelling. *arXiv preprint arXiv:2603.25746*, 2026. URL https://arxiv.org/abs/2603.25746.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2018. URL https://arxiv.org/abs/1706.06083.

Meituan LongCat Team, Xunliang Cai, Qilong Huang, Zhuoliang Kang, Hongyu Li, Shijun Liang, Liya Ma, Siyu Ren, Xiaoming Wei, Rixu Xie, and Tong Zhang. LongCat-Video technical report. *arXiv preprint arXiv:2510.22200*, 2025. URL https://arxiv.org/abs/2510.22200.

Ryan Po, Eric Ryan Chan, Changan Chen, and Gordon Wetzstein. BAgger: Backwards aggregation for mitigating drift in autoregressive video diffusion models. *arXiv preprint arXiv:2512.12080*, 2025. URL https://arxiv.org/abs/2512.12080.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024. URL https://arxiv.org/abs/2410.13720.

Georgy Savva, Oscar Michel, Daohan Lu, Suppakit Waiwitlikhit, Timothy Meehan, Dhairya Mishra, Srivats Poddar, Jack Lu, and Saining Xie. Solaris: Building a multiplayer video world model in minecraft. *arXiv preprint arXiv:2602.22208*, 2026. URL https://arxiv.org/abs/2602.22208.

Linghui Shen, Mingyue Cui, and Xingyi Yang. BadWorld: Adversarial attacks on world models. *arXiv preprint arXiv:2606.16519*, 2026. URL https://arxiv.org/abs/2606.16519.

Joonghyuk Shin, Zhengqi Li, Richard Zhang, Jun-Yan Zhu, Jaesik Park, Eli Shechtman, and Xun Huang. MotionStream: Real-time video generation with interactive motion controls. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0cece806cd3d1dfad4a893f016ad3d7d-Abstract-Conference.html.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2014. URL https://arxiv.org/abs/1312.6199.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. doi: 10.48550/arXiv.2503.20314. URL https://arxiv.org/abs/2503.20314.

Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024. URL https://arxiv.org/abs/2408.14837.

Wenhao Wang and Yi Yang. VidProM: A million-scale real prompt-gallery dataset for text-to-video diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/78b3e7836e3b7dea79d809b0c99cb097-Abstract-Datasets_and_Benchmarks_Track.html.

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. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/91a1610c6ed9e02d33f826b46f472b92-Abstract-Conference.html.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models

with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. URL https://arxiv.org/abs/2408.06072.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. *arXiv preprint arXiv:2405.14867*, 2024a. URL https://arxiv.org/abs/2405.14867.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. *arXiv preprint arXiv:2311.18828*, 2024b. URL https://arxiv.org/abs/2311.18828.

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. *arXiv preprint arXiv:2412.07772*, 2025. URL https://arxiv.org/abs/2412.07772.

Yonghao Yu, Lang Huang, Runyi Li, Zerun Wang, and Toshihiko Yamasaki. Video-mirai: Autoregressive video diffusion models need foresight. *arXiv preprint arXiv:2606.03971*, 2026. URL https://arxiv.org/abs/2606.03971.

Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real real-time long video generation model. *arXiv preprint arXiv:2603.04379*, 2026. URL https://arxiv.org/abs/2603.04379.

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. *arXiv preprint arXiv:2602.02214*, 2026. URL https://arxiv.org/abs/2602.02214.

Junhao Zhuang, Shiyi Zhang, Yuxuan Bian, Yaowei Li, Yawen Luo, Yijun Liu, Weiyang Jin, Songchun Zhang, Xianglong He, Xuying Zhang, Haoran Li, Haoyang Huang, Zeyue Xue, and Nan Duan. Self gradient forcing: Native long video extrapolation. *arXiv preprint arXiv:2607.20368*, 2026. URL https://arxiv.org/abs/2607.20368.

A Local History Robustness and Adversarial Selection

This appendix explains how a history error affects later generated blocks and derives the perturbation used by AGF. We begin by separating newly introduced errors from errors carried forward through generation. This lets us relate a controlled history perturbation to the same transitions. We then explain why the size of a video difference alone is insufficient for choosing an adverse perturbation, before deriving the direction selected by the DMD gradient under the frame-sharing and RMS constraints.

A.1 Causal Error Propagation

At each generation step, the model can both carry forward an existing error and introduce a new one. To express these contributions separately, let S_k be a reference state after block k and $\widetilde{S}_k$ the state produced by the generator. Each state contains the history, cache, and other variables needed

to continue generation. With the sampler state and randomness ω_{k+1} fixed, write the generator's transition as $\widetilde{S}_{k+1} = \mathcal{T}_k(\widetilde{S}_k, \omega_{k+1})$. Define the new error at the reference state by

$$e_{k+1} = \mathcal{T}_k(S_k, \omega_{k+1}) - S_{k+1}. \tag{4}$$

Thus, e_{k+1} is the error the model would introduce even if its input were the reference state S_k. Its actual input may already differ from that reference by $\Delta S_k = \widetilde{S}_k - S_k$, so we also need to account for the effect of this existing difference.

Assume that the transition is continuously differentiable along the segment joining the two states. Averaging its Jacobian along this segment gives A_k^{sec}, which maps the existing difference to its effect one step later. Products of these operators describe propagation over several steps:

$$A_k^{\mathrm{sec}} = \int_0^1 D_S\mathcal{T}_k(S_k + \tau\Delta S_k, \omega_{k+1})\, d\tau, \qquad \Phi_{K,m}^{\mathrm{sec}} = A_{K-1}^{\mathrm{sec}} \cdots A_m^{\mathrm{sec}}, \quad \Phi_{m,m}^{\mathrm{sec}} = I. \tag{5}$$

Here $D_S\mathcal{T}_k$ is the Jacobian of the transition with respect to its state input. The fundamental theorem of calculus gives the exact recursion $\Delta S_{k+1} = A_k^{\mathrm{sec}}\Delta S_k + e_{k+1}$. Repeated substitution then yields

$$\Delta S_K = \Phi_{K,j}^{\mathrm{sec}}\Delta S_j + \sum_{m=j+1}^{K} \Phi_{K,m}^{\mathrm{sec}} e_m. \tag{6}$$

The first term is the effect of the history error already present at block j. Each term in the sum is the effect of an error introduced at a later step. In both cases, the transition product determines how that error contributes to the final state difference.

We are interested in later video blocks, so we extract them from the full state using $P_{>j,K}$, which selects blocks $j+1, \ldots, K$ at horizon K. Define the gain from a state difference at step m to these later blocks as $\Lambda_{K,m}^{\mathrm{sec}} = \|P_{>j,K}\Phi_{K,m}^{\mathrm{sec}}\|_2$. Projecting the exact recursion gives

$$\|P_{>j,K}\Delta S_K\|_2 \le \Lambda_{K,j}^{\mathrm{sec}}\|\Delta S_j\|_2 + \sum_{m=j+1}^{K} \Lambda_{K,m}^{\mathrm{sec}}\|e_m\|_2. \tag{7}$$

Each contribution is therefore bounded by the size of the error multiplied by its propagation gain. This separates the accuracy of an individual generation step from the effect its error can have later.

Let $Y_j(\delta)$ denote the continuation obtained by autoregressive generation after adding δ to history block j. We write

$$\Delta Y_j(\delta) = Y_j(\delta) - Y_j(0) \tag{8}$$

for the difference between the perturbed and original continuations with model parameters and sampling randomness fixed. To express this comparison in terms of the state transitions, choose the model's unperturbed rollout as the reference trajectory. Both branches then follow the same transition with the same future randomness. Consequently, $e_m = 0$ for $m > j$, and the perturbation supplies the initial difference $\Delta S_j = S_j(\delta) - S_j(0)$. Projecting the resulting state difference onto future outputs gives

$$\Delta Y_j(\delta) = P_{>j,K}\Phi_{K,j}^{\mathrm{sec}}\big[S_j(\delta) - S_j(0)\big].$$

Here every later difference comes from the single history intervention. The reference is the model's own rollout, so setting these comparison residuals to zero does not assume that the reference video is free of artifacts.

A.2 Why Video Differences Need a Quality Criterion

The preceding analysis describes how much later outputs change. To select training perturbations, however, we also need to assess whether those changes are adverse to distribution matching. The following example shows why these are different questions: a generator can match its target distribution exactly while its outputs remain highly sensitive to history under shared noise.

Consider a target conditional distribution $\mathcal{N}(0, I_2)$ at every history $h \in \mathbb{R}$. Let $z \sim \mathcal{N}(0, I_2)$, let $\mathrm{Rot}(\alpha)$ be a two-dimensional rotation, and define the smooth family of generators

$$Q_\kappa(h, z) = \mathrm{Rot}(\kappa h)z. \tag{9}$$

Rotational invariance gives $\mathrm{Law}(Q_\kappa(h,z) \mid h) = \mathcal{N}(0, I_2)$ for every h and κ, so the conditional distribution is matched independently of κ. With the same z at two neighboring histories,

$$\mathbb{E}\,\|\partial_h Q_\kappa(h,z)\|_2^2 = 2\kappa^2, \qquad \mathbb{E}\,\|Q_\kappa(h+\delta,z) - Q_\kappa(h,z)\|_2^2 = 4\big(1-\cos(\kappa\delta)\big). \tag{10}$$

Increasing κ makes the local derivative arbitrarily large without changing the conditional output distribution. Thus, large sensitivity under shared randomness alone does not establish a distribution-matching error. With exact score estimates, the target and generated scores in this example coincide, so the DMD gradient is zero for every κ. This is why AGF uses the DMD gradient to assess a history perturbation's effect on later blocks, rather than selecting perturbations solely by the size of the resulting video difference.

A.3 Backpropagating the Future DMD Gradient

AGF selects the perturbation using the parallel forward pass described in Section 3.2(b). This pass uses the recorded noisy inputs $\bar{X}_{t^\star}$ at $t^\star$ and the recorded clean latent blocks $\bar{X}_0$ as history, with δ added only to $\bar{x}_0^{(j)}$. Let $\widehat{y}_j^{(k)}(\delta)$ denote the clean latent prediction for block k, and collect the predictions for blocks after j as $\widehat{Y}_j(\delta) = (\widehat{y}_j^{(j+1)}(\delta), \ldots, \widehat{y}_j^{(K)}(\delta))$. The recorded inputs, all other history blocks, and model parameters remain fixed. Let $J_{j,k}$ and J_j denote the Jacobians of $\widehat{y}_j^{(k)}(\delta)$ and $\widehat{Y}_j(\delta)$ with respect to δ, evaluated at $\delta = 0$. Writing $\bar{g}_{j,k}$ for the entries of the fixed DMD gradient $\bar{g}_j$ corresponding to block k, backpropagation gives

$$u_j = \sum_{k=j+1}^{K} J_{j,k}^\top \bar{g}_{j,k} = J_j^\top \bar{g}_j. \tag{11}$$

Each term maps the DMD gradient of a later block to a gradient on history block j through the differentiable history K/V representations. A single vector–Jacobian product computes their sum without forming the Jacobians. These derivatives describe predictions computed from recorded histories. Appendix A.1 describes error propagation during autoregressive generation, where newly generated blocks become history for subsequent blocks.

A.4 Selecting a Perturbation under Sharing and RMS Constraints

Once we have a gradient with respect to the history block, we need to choose a perturbation that satisfies the frame-sharing and RMS constraints. Using the Jacobian J_j defined above, let $\Delta\widehat{Y}_j(\delta) = \widehat{Y}_j(\delta) - \widehat{Y}_j(0)$. The criterion $\phi_j(\delta) = \left\langle \bar{g}_j, \Delta\widehat{Y}_j(\delta) \right\rangle$ measures the prediction change along the fixed DMD gradient. Holding $\bar{g}_j$ fixed gives

$$\phi_j(\delta) = \langle \bar{g}_j, J_j\delta \rangle + o(\|\delta\|) = \left\langle J_j^\top \bar{g}_j, \delta \right\rangle + o(\|\delta\|). \tag{12}$$

The linear term is therefore maximized by aligning the perturbation with $u_j = J_j^\top \bar{g}_j$, subject to those constraints. For T_j latent frames and flattened spatial-channel dimension D, the allowed set is

$$\begin{aligned} \mathcal{S}_{\mathrm{share},j} &= \left\{ \delta \in \mathbb{R}^{T_j \times D} : \delta_1 = \cdots = \delta_{T_j} \right\}, \\ \mathrm{RMS}(\delta) &= \sqrt{\frac{1}{T_j D} \sum_{\tau=1}^{T_j} \sum_{d=1}^{D} \delta_{\tau,d}^2}, \\ \mathcal{B}_{\epsilon,j} &= \left\{ \delta \in \mathcal{S}_{\mathrm{share},j} : \mathrm{RMS}(\delta) \le \epsilon \right\}. \end{aligned} \tag{13}$$

Here $\mathcal{S}_{\mathrm{share},j}$ requires the same perturbation in every frame, and $\mathcal{B}_{\epsilon,j}$ also bounds its RMS magnitude. To enforce sharing, the orthogonal projection averages the history gradient over frames and assigns the same average to every frame. This gives the shared direction $\widetilde{u}_j = \Pi_{\mathrm{share}}(u_j)$ used in the main text:

$$[\widetilde{u}_j]_\tau = [\Pi_{\mathrm{share}}(u_j)]_\tau = \frac{1}{T_j} \sum_{\ell=1}^{T_j} [u_j]_\ell, \qquad \tau = 1, \ldots, T_j. \tag{14}$$

For every $\delta \in \mathcal{S}_{\text{share},j}$, $\langle u_j, \delta \rangle = \langle \widetilde{u}_j, \delta \rangle$. Cauchy–Schwarz therefore gives

$$\arg\max_{\delta \in \mathcal{B}_{\epsilon,j}} \langle u_j, \delta \rangle = \epsilon \, \frac{\widetilde{u}_j}{\mathrm{RMS}(\widetilde{u}_j)} \quad \text{when } \widetilde{u}_j \neq 0. \tag{15}$$

Stopping gradients through this normalized direction gives $\delta_j^\star$ in Section 3.2(b). When $\widetilde{u}_j = 0$, every allowed perturbation has the same linearized value, and AGF chooses $\delta_j^\star = 0$. The optimality here applies to the first-order criterion with $\bar{g}_j$ fixed; the finite perturbation is subsequently used to recompute predictions under the perturbed history for generator training.

B Paired Training and Gradient Computation

AGF first differentiates with respect to a history perturbation, then fixes that perturbation and differentiates with respect to the generator parameters. This appendix specifies the DMD gradients and the computation paths for these two operations. It also gives the fake-score update, regularizer weighting, and complete training algorithm.

B.1 Applying DMD Gradients to the Generator

Let $X_0 = (x_0^{(1)}, \ldots, x_0^{(K)})$ denote a full video represented by its clean latent blocks, and let c be its text prompt. We use $q(X_0 \mid c)$ for the distribution of videos generated from the original and perturbed histories, weighted according to the two branch losses, and $p_{\text{real}}(X_0 \mid c)$ for the target video distribution. DMD compares these distributions after adding Gaussian noise to the videos (Yin et al., 2024b;a). At diffusion time τ, the noisy video is $X_\tau = \alpha_\tau X_0 + \sigma_\tau \varepsilon_{\text{diff}}$, where $\varepsilon_{\text{diff}} \sim \mathcal{N}(0, I)$ and α_τ, σ_τ specify the noise schedule. Writing the corresponding noisy distributions as q_τ and $p_{\text{real},\tau}$, the DMD objective is

$$\mathcal{L}_{\text{DM}}(q) = \mathbb{E}_{c,\tau}[D_{\text{KL}}(q_\tau(\cdot \mid c) \,\|\, p_{\text{real},\tau}(\cdot \mid c))]\,. \tag{16}$$

To make the sampling explicit, we write the DMD gradient introduced in Section 3.2 as $g_{\text{DM}}(X_0; \xi)$ in this appendix, where $\xi = (\tau, \varepsilon_{\text{diff}})$ contains the sampled diffusion time and noise. These are sampled separately from the randomness used to generate the video, including the denoising step $t^\star$ selected for the generator's recorded inputs $\bar{X}_{t^\star}$. Overbars denote values held fixed during differentiation. Equation 16 defines the distributional objective. To derive the gradient used to train the generator, consider a differentiable sampler $X_{0,\theta} \sim q_\theta(\cdot \mid c)$ and define the scores of the noisy generated and target distributions by

$$s_{q_\theta}(X_\tau, \tau, c) = \nabla_{X_\tau} \log q_{\theta,\tau}(X_\tau \mid c), \qquad s_{\text{real}}(X_\tau, \tau, c) = \nabla_{X_\tau} \log p_{\text{real},\tau}(X_\tau \mid c).$$

Using exact scores, the distribution-matching gradient is (Yin et al., 2024b;a)

$$\nabla_\theta \mathcal{L}_{\text{DM}}(q_\theta) = \mathbb{E}\big[(D_\theta X_{0,\theta})^\top \alpha_\tau \big(s_{q_\theta}(X_\tau, \tau, c) - s_{\text{real}}(X_\tau, \tau, c)\big)\big]\,. \tag{17}$$

Here $D_\theta X_{0,\theta}$ is the Jacobian of the generated video with respect to the generator parameters. The expectation includes prompts, generator sampling randomness, and the diffusion time and noise. The factor α_τ comes from differentiating $X_\tau = \alpha_\tau X_{0,\theta} + \sigma_\tau \varepsilon_{\text{diff}}$ with respect to the generated video. Thus, the difference between the two scores supplies an output gradient that is backpropagated through the generator.

For the Gaussian noising process, a denoised video prediction and its score estimate satisfy

$$\widehat{s}_a(X_\tau, \tau, c) = \frac{\alpha_\tau \widehat{X}_{0,a}(X_\tau, \tau, c) - X_\tau}{\sigma_\tau^2}, \qquad a \in \{\text{real}, \text{fake}\}.$$

Substituting these estimates into the score difference gives $\alpha_\tau(\widehat{s}_{\text{fake}} - \widehat{s}_{\text{real}}) = \frac{\alpha_\tau^2}{\sigma_\tau^2}(\widehat{X}_{0,\text{fake}} - \widehat{X}_{0,\text{real}})$. The practical DMD gradient uses estimated scores and an additional normalization:

$$g_{\text{DM}}(X_0; \xi) = \gamma_{\text{DM}}(X_0, \tau) \left[\widehat{X}_{0,\text{fake}}(X_\tau, \tau, c) - \widehat{X}_{0,\text{real}}(X_\tau, \tau, c)\right], \tag{18}$$

where $\gamma_{\text{DM}}(X_0, \tau) > 0$ includes diffusion scaling and normalization. When this normalization depends on the sample, it changes the weighting of samples relative to the unscaled KL gradient in Eq. 17.

To backpropagate a fixed DMD gradient g through the generator without differentiating through either score model, we construct the mean squared error surrogate loss

$$\ell_{\mathrm{DM}}(X_0; g) = \frac{1}{2N} \left\| X_0 - \mathrm{sg}(X_0 - g) \right\|_2^2, \qquad \nabla_{X_0} \ell_{\mathrm{DM}}(X_0; g) = \frac{g}{N}. \tag{19}$$

Here N is the number of scalar latent coordinates in X_0. The target $\mathrm{sg}(X_0 - g)$ is treated as constant, so gradients pass only through the first occurrence of X_0. Differentiation therefore gives $(X_0 - \mathrm{sg}(X_0 - g))/N = g/N$, with $1/N$ implementing the mean reduction. The loss thus applies the fixed DMD gradient, with a factor of $1/N$ from averaging over latent coordinates; its numerical value is not an estimate of the KL divergence.

During perturbation selection, we use the DMD gradient before this averaging and keep only its entries for blocks after j:

$$\bar{g}_j = \mathrm{sg}\big([g_{\mathrm{DM}}(X^{\mathrm{in}}_{0,j}(0); \xi^{\mathrm{clean}})]_{>j}\big), \tag{20}$$

Here $X^{\mathrm{in}}_{0,j}(0)$ is the full video prediction recomputed under the original history from the recorded inputs, as defined in Appendix B.2, and $[\cdot]_{>j}$ extracts its later blocks. During generator training, each branch $b \in \{\mathrm{clean}, \mathrm{adv}\}$ uses $g^b = \mathrm{sg}(g_{\mathrm{DM}}(X_0^b; \xi^b))$ evaluated on its own full prediction X_0^b. The clean branch reuses the DMD gradient computed for selection; the adversarial branch draws its own diffusion time and noise. These DMD draws are separate from the recorded noisy inputs used for the parallel forward pass.

The adversarial loss averages the same squared error only over blocks after j. Let $N_{>j}$ be their number of scalar latent coordinates. Then

$$\ell^{>j}_{\mathrm{DM}}(X_0; g) = \frac{1}{2N_{>j}} \left\| [X_0]_{>j} - \mathrm{sg}([X_0]_{>j} - [g]_{>j}) \right\|_2^2.$$

Its gradient with respect to X_0 is zero on blocks up to j and equals $[g]_{>j}/N_{>j}$ on later blocks. The two branch losses therefore average over different coordinate sets; their weights in Eq. 2 apply to these mean losses. For perturbation selection, multiplying the mean-reduced inner loss by $N_{>j}$ recovers the unaveraged gradient $J_j^\top \bar{g}_j$. At fixed j, this positive scaling does not change the direction after RMS normalization. The mask determines where gradients are applied; DMD itself is evaluated on the full video in both selection and training.

B.2 Parallel Forward Passes for Selection and Training

The autoregressive rollout in Section 3.2(a) records the noisy inputs $\bar{X}_{t^\star}$ at a sampled denoising step $t^\star$ shared across all blocks and their corresponding clean predictions $\bar{X}_0$, without retaining the computation graph. These inputs are recorded from the generator's denoising trajectory; they are not constructed by adding fresh Gaussian noise to $\bar{X}_0$. The independent forward-noising relation in Appendix B.1 instead applies to a clean video prediction X_0 and its DMD input X_τ. The text prompt is denoted by c. Let $\bar{X}_0(\delta)$ be $\bar{X}_0$ with δ added only to $\bar{x}_0^{(j)}$. We write $\mathcal{R}_\theta(\bar{X}_{t^\star}, t^\star; \bar{X}_0(\delta), c)$ for the full video prediction produced by the generator's parallel forward pass.

The generator encodes the recorded clean latent blocks and noisy inputs together under causal attention. Each noisy block uses the K/V representations of the preceding recorded blocks as history. Since these history blocks are already available, all block predictions can be computed in parallel. Newly computed predictions are not inserted into the history during this pass. The serial K/V cache is released after the rollout; each parallel forward pass recomputes differentiable history K/V representations from the recorded clean latent blocks.

During selection, the parameters $\bar{\theta} = \mathrm{sg}(\theta)$ are fixed, and the zero-initialized δ is the variable being differentiated:

$$X^{\mathrm{in}}_{0,j}(\delta) = \mathcal{R}_{\bar{\theta}}(\bar{X}_{t^\star}, t^\star; \bar{X}_0(\delta), c), \qquad \widehat{Y}_j(\delta) = [X^{\mathrm{in}}_{0,j}(\delta)]_{>j}. \tag{21}$$

DMD is evaluated on the complete prediction $X^{\mathrm{in}}_{0,j}(0)$. Its later-block entries are backpropagated through the parallel forward pass to δ, producing the gradient used to construct $\delta_j^\star$.

We then detach the selected perturbation and discard the selection graph. For $b \in \{\mathrm{clean}, \mathrm{adv}\}$, let $\delta^{\mathrm{clean}} = 0$ and $\delta^{\mathrm{adv}} = \delta_j^\star$. The fixed clean latent blocks used as history in each branch are

$C^b = \mathrm{sg}(\bar{X}_0(\delta^b))$. Using the same parallel computation, the clean and adversarial branches produce

$$X_0^b = \mathcal{R}_\theta\big(\bar{X}_{t^\star}, t^\star; C^b, c\big)\,,$$
$$g^b = \mathrm{sg}(g_{\mathrm{DM}}(X_0^b; \xi^b)). \tag{22}$$

The recorded noisy inputs and history blocks are fixed, but the history K/V representations are recomputed with trainable parameters. Gradients therefore update both history encoding and denoising. Each X_0^b contains predictions for all video blocks. Before the parameter update, the clean branch uses the same recorded inputs, history, and parameters as perturbation selection, so

$$X_0^{\mathrm{clean}} = X_{0,j}^{\mathrm{in}}(0). \tag{23}$$

This permits the clean branch to reuse the DMD gradient computed during selection. Table 3 summarizes the two gradient computations.

Table 3: Gradient paths for perturbation selection and generator training.

	Perturbation selection	Generator training
Differentiated variable	history perturbation δ	generator parameters θ
Active path	history K/V computation and later predictions	history K/V computation and predictions
Fixed values	parameters, recorded inputs and history, DMD gradient	perturbation, recorded inputs and history, DMD gradient
Output	history perturbation	generator update

B.3 Training the Fake-Score Model

The generator learns from full video predictions recomputed under the original and perturbed histories. We train one fake-score model on their weighted mixture. With branch loss weights λ_{clean} and λ_{adv}, define

$$\pi_{\mathrm{clean}} = \frac{\lambda_{\mathrm{clean}}}{\lambda_{\mathrm{clean}} + \lambda_{\mathrm{adv}}}, \qquad q_{\mathrm{mix}}^{(s)} = \pi_{\mathrm{clean}} q_{\mathrm{clean}}^{(s)} + (1 - \pi_{\mathrm{clean}}) q_{\mathrm{adv}}^{(s)}. \tag{24}$$

Here $q_{\mathrm{clean}}^{(s)}$ and $q_{\mathrm{adv}}^{(s)}$ are the video distributions from the two branches at training iteration s. They use the same sampling law over prompts, generator noise, history positions, and DMD draws (c, ω, j, ξ). Both branch weights are 0.5 in the default configuration. The fake-score model is shared by the clean and adversarial branches. For each branch, the same teacher and fake-score model are evaluated on the Gaussian-noised version of that branch's full video prediction.

Training also reuses generated videos stored in a replay buffer. If N_{replay} is the maximum age and β_ℓ is the sampling weight for videos of age ℓ, their distribution is

$$q_{\mathrm{replay}}^{(s)} = \sum_{\ell=0}^{N_{\mathrm{replay}}} \beta_\ell q_{\mathrm{mix}}^{(s-\ell)}, \qquad \beta_\ell \geq 0, \quad \sum_{\ell=0}^{N_{\mathrm{replay}}} \beta_\ell = 1. \tag{25}$$

The default replay scheme preserves the branch proportions at every age. All videos are detached from the generator before the fake-score model is trained on them with the diffusion regression objective used by DMD. Thus, this update trains the score estimate without updating the video generator.

B.4 Prediction Consistency Regularization and Weighting

The regularizer compares first-step velocity predictions under the two histories used in Eq. 22. Let V_θ denote parallel velocity prediction at the highest-noise training timestep and $Z_{\max}$ the initial noisy latents recorded during sampling. Writing C^b for the fixed clean latent blocks used as history in branch b, the predictions for blocks after j are

$$v_{j,1}^b = [V_\theta(Z_{\max}; C^b, c)]_{>j}, \qquad b \in \{\mathrm{clean}, \mathrm{adv}\}.$$

Both calls use the current generator parameters and the same noisy inputs. We recompute the clean prediction for each update and then detach it to provide the reference in Eq. 3. The operator $\mathrm{MSE}_{>j}$

averages squared differences over coordinates after block j. Gradients pass through the adversarial prediction and its history K/V computation; the recorded history blocks and clean reference remain fixed.

The gradients from DMD and the regularizer can change scale during training. To control their relative strength, we weight the regularizer using the ratio of estimated gradient RMS values at the model outputs. At iteration s, denote these estimates by $r_{\mathrm{DM}}^{(s)}$ and $r_{\mathrm{reg}}^{(s)}$, and let ρ_s be the desired relative strength of the regularizer. The loss weight is

$$\lambda_s = \mathrm{sg}\left(\min\left\{\lambda_{\max}, \rho_s \frac{r_{\mathrm{DM}}^{(s)}}{r_{\mathrm{reg}}^{(s)} + \epsilon_{\mathrm{num}}}\right\}\right), \qquad \mathcal{L}_{\mathrm{train}} = \mathcal{L}_{\mathrm{AGF}} + \lambda_s \mathcal{L}_{\mathrm{reg}}. \tag{26}$$

The ratio adjusts the regularization coefficient as the two gradient scales change; the cap $\lambda_{\max}$ and stabilizer ϵ_{num} limit large weights when the regularizer gradient estimate is small. These are estimates at the outputs used for calibration, rather than norms of the full generator parameter gradients. The coefficient itself is detached during the update.

The evaluated configuration uses $\epsilon_{\mathrm{num}} = 10^{-12}$ and $\lambda_{\max} = 1000$. The target ratio ρ_s rises from zero at iteration 20 to 0.05 at iteration 80, stays there through iteration 200, and is zero for iterations 201–400. Thus, 0.05 is the target gradient ratio, not a fixed loss coefficient. After the regularizer is removed, training continues with Eq. 2. Table 4 records the complete evaluated configuration.

B.5 Complete Training Update

Algorithm 1 collects the computations for a generator update. The fake-score model is trained separately using videos from both branches, detached from the generator and stored in replay. In our experiments, there are five fake-score updates per generator update; the 400 outer iterations therefore contain 80 generator updates and 400 fake-score updates. The regularization schedule uses the outer iteration index s.

Algorithm 1 AGF generator update

Require: Generator θ, prompt c, perturbation budget ϵ, learning rate η_θ, branch weights $\lambda_{\mathrm{clean}}, \lambda_{\mathrm{adv}}$, iteration s.

1: Perform an autoregressive rollout without retaining its graph. Record noisy inputs $\bar{X}_{t^\star}$ at a sampled denoising step $t^\star$ shared across all blocks and corresponding clean predictions $\bar{X}_0$; retain the initial noise $Z_{\max}$ for regularization. Select $j < K$.
2: With θ fixed, recompute the full video prediction $X_{0,j}^{\mathrm{in}}(\delta)$ at $\delta = 0$ with a parallel forward pass (Eq. 21).
3: Sample ξ^{clean} and evaluate the full-video DMD gradient $g^{\mathrm{clean}} \leftarrow \mathrm{sg}(g_{\mathrm{DM}}(X_{0,j}^{\mathrm{in}}(0); \xi^{\mathrm{clean}}))$. Set $\bar{g}_j \leftarrow [g^{\mathrm{clean}}]_{>j}$.
4: Backpropagate to history: $u_j \leftarrow J_j^\top \bar{g}_j$ (Eq. 11); form the shared direction $\widetilde{u}_j \leftarrow \Pi_{\mathrm{share}}(u_j)$ (Eq. 14).
5: Set $\delta_j^\star \leftarrow \mathrm{sg}(\epsilon \widetilde{u}_j / \mathrm{RMS}(\widetilde{u}_j))$, or zero if $\widetilde{u}_j = 0$. Discard the selection graph.
6: Add $\delta_j^\star$ to $\bar{x}_0^{(j)}$ to form the perturbed history. Recompute X_0^{clean} and X_0^{adv} under the original and perturbed histories with θ trainable and recorded inputs and history blocks fixed (Eq. 22).
7: Reuse g^{clean}, sample ξ^{adv}, and evaluate $g^{\mathrm{adv}} \leftarrow \mathrm{sg}(g_{\mathrm{DM}}(X_0^{\mathrm{adv}}; \xi^{\mathrm{adv}}))$. Form $\mathcal{L}_{\mathrm{AGF}}$ (Eq. 2).
8: During the regularization phase, compute the paired first-step velocity predictions and $\mathcal{L}_{\mathrm{reg}}$ (Eq. 3); set λ_s using Eq. 26. Otherwise set $\lambda_s = 0$.
9: Update the generator through history K/V computation and denoising: $\theta \leftarrow \theta - \eta_\theta \nabla_\theta (\mathcal{L}_{\mathrm{AGF}} + \lambda_s \mathcal{L}_{\mathrm{reg}})$.
10: Store detached branch videos in replay for fake-score training with proportions $\lambda_{\mathrm{clean}} : \lambda_{\mathrm{adv}}$ (Appendix B.3).

C Visualizing Changes in History and Continuations

The figures below show where a history perturbation changes the decoded video. For each example, we decode the clean and perturbed videos separately and subtract their RGB pixels. We show this

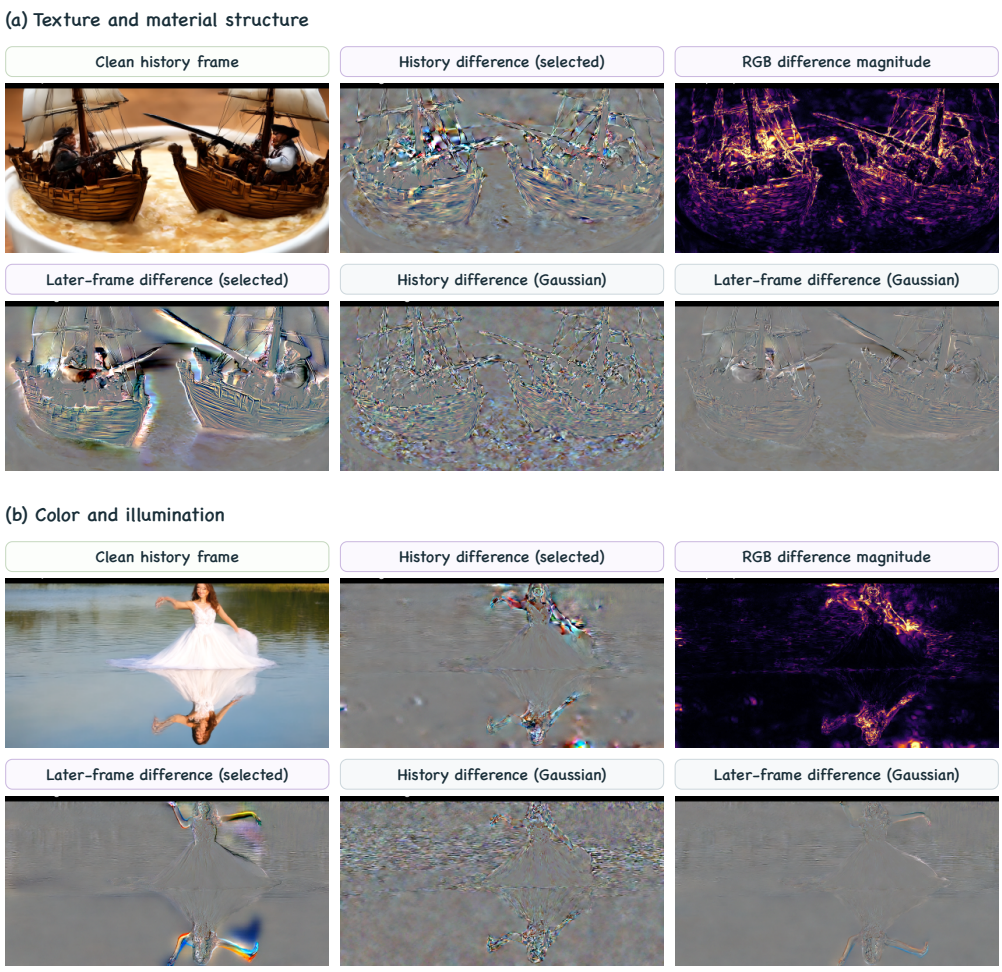


Figure 8: **Pixel changes around objects and illumination regions.** In the ship scene, the selected perturbation changes pixels along the hulls, masts, and material boundaries. In the dancer scene, changes appear around the subject, her reflection, and regions of similar brightness. Similar patterns appear in the differences between later frames. The matched Gaussian perturbation produces weaker and more diffuse later-frame differences in these examples.

difference in the history block and in a later generated frame, so the reader can compare the initial change with its effect on subsequent generation. The clean frame provides a scene reference.

Figure 4 uses an archived diagnostic checkpoint to inspect the perturbation-selection procedure. Its panels show the same scene and perturbations as the ship example below. For each comparison, the model parameters, original history, and future sampling randomness are fixed; only the inserted perturbation changes.

History panels use frame 27, within the perturbed block, and later-frame panels use frame 48; indices are zero-based. We compare the selected perturbation with Gaussian noise at the same latent RMS budget, with each pattern shared across three latent frames.

Let $\Delta I = I_{\text{perturbed}} - I_{\text{clean}}$, where both decoded RGB images have values in $[0, 1]$. To make small differences visible, all signed maps use the display transform $\text{clip}_{[0,1]}(0.5 + 4\Delta I)$; mid-gray denotes zero difference. Magnitude maps average the absolute RGB differences over channels, amplify them $8\times$, and clip them before applying the Inferno colormap. All panels use these same scales. The RMS budget is matched in latent space before decoding; the maps show differences between decoded videos.

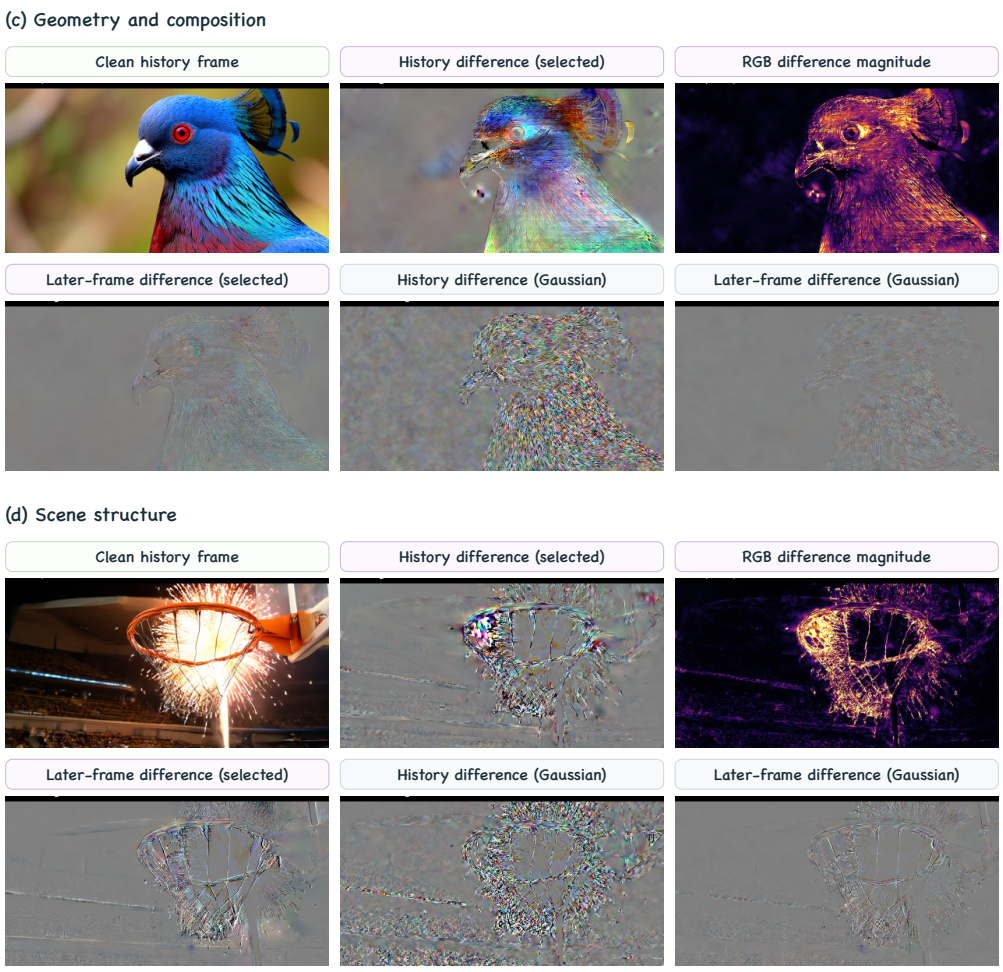


Figure 9: **Pixel changes along object outlines and scene structures.** The selected perturbations change pixels along the pigeon's head and body outline, or around the hoop, net, and surrounding basketball scene. Similar contours appear in the differences between later frames, while the matched Gaussian perturbation produces more diffuse changes.

D Training and Evaluation Details

D.1 Training Configuration

Table 4 summarizes the configuration of the model evaluated in Section 4, using its EMA weights at iteration 400. Training resumes at iteration 200 with the regularizer disabled, retaining the generator, fake-score model, optimizer, EMA, replay buffer, random-number generator, and data-loader states.

Training-cost accounting. Table 5 lists the training configuration, video-data requirements, and timing measurements for the stage included in each method's cost estimate. All timings use eight H20 GPUs. Iterations count outer training loops: AGF's 400 iterations comprise 80 generator updates and 400 fake-score updates.

For methods whose costs are extrapolated, the budget is $8N\bar{t}(B/B_{\text{timed}})/3600$ GPU-hours, where N and B are the iterations and global batch in Table 5(a), and $\bar{t}$ is the mean seconds per iteration at batch B_{timed} in (b). This assumes linear workload scaling with batch. Self-Forcing uses historical rank-0 log timestamps; OPSD-V and LongLive use synchronized timings maximized over all eight ranks. For DMD-based methods, timings are averaged over complete cycles of generator and fake-score updates. OPSD-V timing uses synthetic inputs with 180 latent frames; its target batch of 24 follows the reported 24 devices with one sample each and assumes no extra accumulation.

Table 4: Training configuration for the evaluated AGF model. Optimizer and perturbation settings correspond to the pure-AGF stage.

Field	Configuration
Generator / fake-score model	Wan2.1-T2V-1.3B / one Wan2.1-T2V-1.3B fake-score model
Frozen target score model	Wan2.1-T2V-14B
Initialization	Self-Forcing ODE checkpoint
Prompt source	Self-Forcing's filtered, LLM-extended VidProM text prompts; sampled line-wise
Training rollout	21 latent frames; 3 frames per block; 7 latent blocks ($\sim$5 s)
Steps 1–200	AGF with regularization applied to the first denoising prediction
Regularization schedule	$\rho_s = 0$ at step 20, linear ramp to 0.05 at step 80, hold through step 200
Resuming training	Full training state resumed at step 200; optimizer and replay retained
Steps 201–400	Pure AGF; the regularizer disabled; step-400 EMA evaluated
Perturbation	One gradient step using DMD gradients on later blocks; candidate block indices $[1, 2, 3]$; RMS budget 0.05; same pattern in every latent frame; no random initialization
Branch weights	Clean/adversarial generator loss $0.5/0.5$
Fake-score updates	One fake-score model; 5:1 ratio of fake-score to generator updates
Replay	Enabled; 8 batches per rank on CPU in bf16; clean/adversarial $0.5/0.5$
Batch	Microbatch 1 per rank; 8 ranks; accumulation 8; effective global batch 64
Optimizer	AdamW, $\beta = (0, 0.999)$; generator/fake-score LR $2\times10^{-6}/4\times10^{-7}$; generator weight decay 0.01
Sampler in training	Nodes $[1000, 750, 500, 250]$ mapped through the scheduler with shift 5.0; same sampled exit step across blocks; guidance 3.0
Gradient / precision	Max generator/fake-score norm 10/10; bf16; gradient checkpointing
Distributed training	FSDP `hybrid_full`; 8$\times$ NVIDIA H20; seed 20260731
EMA	Weight 0.99; EMA restored at step 200 and evaluated at step 400

AGF's recorded elapsed time is the sum of the regularized and pure-AGF runs ($23.6 + 21.9$ hours). Each interval extends from a recorded resource snapshot to the corresponding checkpoint timestamp. It includes some setup and checkpoint saving, but excludes idle time between runs and training beyond step 400. We use the same accounting for our SGF reproduction from the ODE initialization. Estimates based on short timing runs exclude loading, warmup, and checkpointing. These costs cover different training stages and use different timing procedures; they do not measure throughput under a common workload.

D.2 Long-Horizon Inference Protocol

At 60 seconds, Temporal Flickering applies an additional static-video filter and retains at most five selected segments per prompt; consequently its valid-prompt counts are 17/17/15/19/19/18 for Self-Forcing/OPSD-V/SGF/LongLive/Causal Forcing/AGF rather than all 20 prompts assigned to that dimension. Tables 8 and 9 report the valid counts at each horizon.

The 5-second evaluation set contains 944 unique VBench prompts. The 60-second set is a randomly sampled 240-prompt subset. The 240-second set contains 120 prompts randomly sampled from the 1,003-prompt MovieGenVideoBench_extended pool (Polyak et al., 2024). Generation uses noise seed $100000 + i$ for prompt index i. Across methods, inference uses Wan2.1-T2V-1.3B at 832$\times$480 and 16 fps, denoising nodes $[1000, 750, 500, 250]$, guidance 3.0, timestep shift 5.0, three frames per block, context noise 0, and local attention over 12 latent frames: three persistent sink frames and nine rolling frames. Relative sink positions are enabled, and no ground-truth replacement is used.

D.3 Metrics and Statistical Protocol

At 60 seconds, aesthetic quality, imaging quality, human action, temporal style, and overall consistency use nominal 10-second clips; the other dimensions use two-second clips, except appearance style, which uses two-second clips for 18 prompts and 10-second clips for two prompts. Each dimension averages its valid segment scores before the normalization below. Temporal flickering instead averages the evaluator's per-video aggregates over the videos retained by its static-content filter. The valid counts for each dimension appear in Table 9.

Table 5: **Training requirements and cost accounting on eight H20 GPUs.** Only the listed stages are counted; upstream initialization and data preparation are excluded. Check marks indicate yes and dashes indicate no.

(a) Training configuration and video-data requirements

Method	Training stage	Iterations	Global batch	Training horizon	Extra video data
Self-Forcing	DMD	600	64	5 s	–
Causal Forcing	Asymmetric DMD	750	64	5 s	–
LongLive	Long tuning	3,000	64	60 s stream (5 s windows)	–
SGF	ODE-init reproduction	1,200	8	5 s	–
OPSD-V	LoRA post-training	200	24	45 s	✓
AGF	Regularized + pure AGF	200 + 200	64	5 s	–

(b) Timing evidence and resulting budget

Method	Timing sample	Timed batch	Timing basis	Extrap.	GPU-hours
Self-Forcing	35 / 5	8	40.5 s/iter	✓	431.6
Causal Forcing	SF timing proxy	8	40.5 s/iter	✓	539.6
LongLive	25 / 5	8	52.0 s/iter	✓	2774.4
SGF	Full run	8	19.3 h	–	154.8
OPSD-V	4 / 2	8	708 s/iter	✓	944.4
AGF	Full run	64	45.5 h	–	**364.4**

Extra video data refers to the training stage listed in (a): OPSD-V uses 3,800 real long videos (Liu et al., 2026); LongLive uses generated text-prompt pairs, without video data (Yang et al., 2026). Timing sample gives retained / excluded warmup iterations. Full run denotes a recorded operating interval; SF timing proxy reuses Self-Forcing's measurements, without a separate Causal Forcing timing run. Extrap. marks scaling to the full iteration count and global batch in (a).

Table 6: Evaluation sets for each video duration. Each method produces one video per prompt, using the same mapping from prompt index to noise seed.

Horizon	Prompts	Frames	Seconds	Segments/video	Evaluated dimensions
5 s	944	81	5.0625	1	16 VBench dimensions
60 s	240	957	59.8125	6 (10 s) or 30 (2 s)	16 VBench dimensions
240 s	120	3837	239.8125	120	6 dimensions

Table 7: Model variants used in the controlled comparison. All six methods share the generation and evaluation protocol described above.

Method	Evaluated weights
Self-Forcing	Self-Forcing generator checkpoint; no additional EMA selection at inference
OPSD-V	Rank-256, $\alpha = 256$ LoRA (dropout 0); non-EMA weights
SGF	Our reproduction using the official hyperparameters and ODE initialization; generator EMA at iteration 1200
LongLive	Released generator base and generator LoRA; non-EMA weights
Causal Forcing	Released final Stage-3 chunkwise generator; non-EMA weights
AGF (ours)	Step-400 generator EMA after 200 regularized and 200 pure-AGF iterations

To compute the reported 5- and 60-second scores, we normalize each raw dimension as $(x - l)/(u - l)$. The VBench normalization bounds used here are: subject $[0.1462, 1]$, background $[0.2615, 1]$, flickering $[0.6293, 1]$, motion smoothness $[0.706, 0.9975]$, scene $[0, 0.8222]$, appearance style $[0.0009, 0.2855]$, and temporal style and overall consistency $[0, 0.364]$; all remaining dimensions use $[0, 1]$. Dynamic Degree has weight 0.5 in the quality mean and every other dimension has weight 1. Quality and semantic subscores are combined as $0.8Q + 0.2S$.

At 240 seconds, we define two summaries of the six custom-prompt dimensions in VBench-Long (Huang et al., 2025b). Q6 is the unweighted mean of subject consistency, background consis-

tency, motion smoothness, aesthetic quality, imaging quality, and Dynamic Degree. Q5 averages the first five dimensions, which are also used in LongLive's interactive evaluation (Yang et al., 2026). Whole-video Q5/Q6 average the per-video dimension scores and then average across prompts. Neither composite is an official VBench total. For the temporal versions, subject and background use their local `inclip_score`, whereas the whole-video scores also include clip-to-clip consistency. The remaining dimensions use their local segment score. Local Q5 therefore tracks quality within successive intervals; it does not by itself establish identity or scene correspondence to the opening of the video.

For Figure 6, ten consecutive two-second positions form each non-overlapping 20-second bin. We subtract each prompt's first-bin Q5 from its later bins before averaging over prompts. Pointwise 95% percentile bootstrap intervals use 10,000 resamples of the 120 prompt indices (seed 20260907). Each resample retains complete trajectories and uses the same indices across methods, preserving temporal and cross-method pairing. These bands quantify prompt-level uncertainty in this evaluation setting; they are not simultaneous confidence bands or per-video prediction intervals. The comparison does not estimate variation across generation seeds, training checkpoints, or evaluator versions.

E Complete Multi-Length Results

Table 8: Complete raw dimension scores at 5 seconds. Each n entry gives valid prompt counts in Self-Forcing/OPSD-V/SGF/LongLive/Causal Forcing/AGF order. Scores are percentages; the reported summary scores are computed after normalization.

Dimension	n	Self-Forcing	OPSD-V	SGF	LongLive	Causal Forcing	AGF
Subject consistency	72/72/72/72/72/72	96.32	96.08	96.73	97.47	95.68	96.52
Background consistency	86/86/86/86/86/86	95.84	95.24	95.45	97.07	95.70	95.05
Temporal flickering	75/75/75/75/75/75	99.00	98.55	97.97	99.31	98.81	98.85
Motion smoothness	72/72/72/72/72/72	98.65	98.43	98.23	98.94	98.21	98.52
Dynamic degree	72/72/72/72/72/72	72.22	73.61	73.61	31.94	84.72	70.83
Aesthetic quality	93/93/93/93/93/93	62.29	62.62	64.69	63.71	64.65	63.92
Imaging quality	93/93/93/93/93/93	70.16	70.35	70.74	69.39	71.27	70.55
Object class	79/79/79/79/79/79	81.41	81.65	84.02	81.88	83.62	82.28
Multiple objects	82/82/82/82/82/82	67.15	63.41	72.87	70.12	66.62	67.23
Human action	100/100/100/100/100/100	82.00	78.00	77.00	83.00	80.00	80.00
Color	81/79/81/80/82/82	92.70	92.04	92.36	92.68	89.06	90.90
Spatial relationship	84/84/84/84/84/84	79.84	83.72	80.87	81.08	78.38	79.21
Scene	86/86/86/86/86/86	32.63	36.56	27.91	27.76	30.31	30.38
Appearance style	90/90/90/90/90/90	19.15	18.92	19.32	19.05	19.35	19.39
Temporal style	100/100/100/100/100/100	23.41	23.30	23.27	22.87	24.31	23.29
Overall consistency	93/93/93/93/93/93	24.49	24.26	24.16	24.15	24.58	24.34

Table 9: Complete raw dimension scores at 60 seconds. The count and method order follow Table 8.

Dimension	n	Self-Forcing	OPSD-V	SGF	LongLive	Causal Forcing	AGF
Subject consistency	20/20/20/20/20/20	97.51	97.46	97.83	98.36	95.07	97.91
Background consistency	42/42/42/42/42/42	96.32	95.82	95.92	96.78	94.87	96.56
Temporal flickering	17/17/15/19/19/18	99.06	98.53	98.54	99.43	99.02	99.14
Motion smoothness	20/20/20/20/20/20	98.53	98.39	98.49	98.78	97.05	98.58
Dynamic degree	20/20/20/20/20/20	61.33	61.67	57.50	35.83	87.83	68.50
Aesthetic quality	20/20/20/20/20/20	55.08	57.07	59.92	58.17	56.18	58.48
Imaging quality	20/20/20/20/20/20	68.06	69.87	70.80	68.79	69.38	71.05
Object class	20/20/20/20/20/20	79.53	79.95	75.33	81.22	73.07	84.05
Multiple objects	20/20/20/20/20/20	63.90	69.75	66.98	85.67	53.95	72.75
Human action	20/20/20/20/20/20	77.50	79.17	76.67	80.83	76.67	77.50
Color	20/20/20/20/20/20	88.87	86.31	80.14	92.51	83.84	93.77
Spatial relationship	20/20/20/20/20/20	70.48	80.03	74.72	71.65	76.36	71.37
Scene	42/42/42/42/42/42	19.44	21.37	16.59	29.93	18.87	24.46
Appearance style	20/20/20/20/20/20	20.06	19.44	19.47	19.50	20.06	19.78
Temporal style	20/20/20/20/20/20	20.81	21.12	19.80	20.96	20.93	20.72
Overall consistency	20/20/20/20/20/20	22.88	22.50	21.95	22.29	21.98	22.79

Table 10: All six dimensions evaluated at 240 seconds on the same 120 prompts. Values are percentages.

Dimension	Self-Forcing	OPSD-V	SGF	LongLive	Causal Forcing	AGF
Subject consistency	96.36	96.29	97.11	97.08	94.44	**97.32**
Background consistency	95.79	95.64	96.10	96.24	94.82	**96.38**
Motion smoothness	97.95	97.91	98.18	**98.60**	95.99	98.31
Aesthetic quality	54.97	56.60	58.32	58.29	55.29	**61.20**
Imaging quality	67.26	67.77	70.63	67.86	68.41	**70.91**
Dynamic degree	63.19	65.10	51.08	49.53	**91.41**	62.54
Q6 (all six dimensions)	79.25	79.88	78.57	77.94	**83.39**	81.11
Q5 (excludes Dynamic Degree)	82.47	82.84	84.07	83.62	81.79	**84.82**

F Additional Long-Horizon Examples

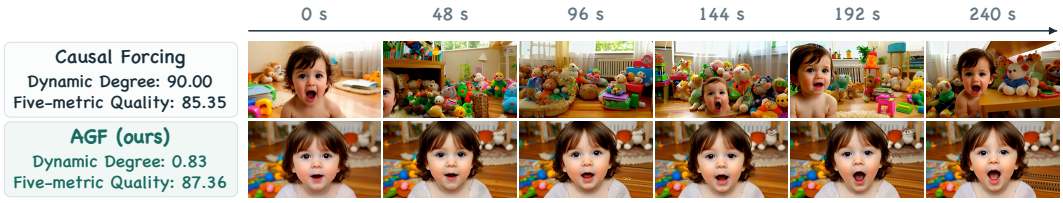
These 240-second rollouts show how scene composition, dynamic interactions, and subject and background structure evolve over time. Every row shows the same six time points: 0, 48, 96, 144, 192, and 240 seconds. Labels report whole-video scores in percent: *Dynamic Degree* and *Five-metric Quality*, the unweighted mean of subject consistency, background consistency, motion smoothness, aesthetic quality, and imaging quality (Q5). Original prompt excerpts appear below each panel.

F.1 Camera Drift and Loss of Scene Structure

Higher Dynamic Degree does not necessarily indicate better visual quality: a larger motion score can accompany camera drift or structural degradation. The following cases show AGF preserving clearer content and obtaining higher Q5 at lower Dynamic Degree.



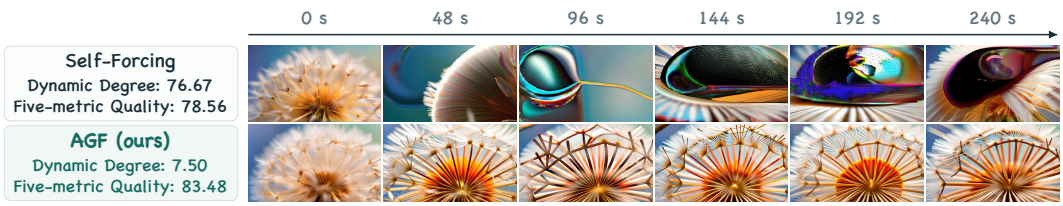
(a) Prompt (excerpt): A dramatic and surreal photograph in a realistic style, capturing an abandoned warehouse where vibrant flowers are blooming from the cracked concrete walls. ...



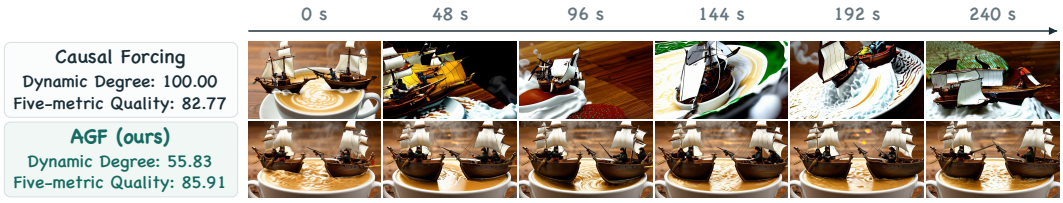
(b) Prompt (excerpt): A candid moment captured in a soft, natural light photograph of a little child yawning widely. ...

Figure 10: **Camera drift changes the scene composition. (a) Warehouse flowers:** SGF shifts into close-ups that lose the industrial surroundings, while AGF keeps the flowers and warehouse visible together. **(b) Yawning child:** Causal Forcing moves the view toward the toys and partly or fully excludes the child; AGF retains a clear view of the child and the changing expression.

In Figure 10(a), SGF alternates between warehouse views and extreme flower close-ups, repeatedly changing how the flowers are framed within the building. AGF keeps the flowers and industrial interior visible together in a consistent composition. In Figure 10(b), Causal Forcing redirects the view from the child toward the toys: the child is partly cropped at 48 seconds and absent at 96 seconds. AGF keeps the child's face centered as the mouth and expression change.



(a) Prompt (excerpt): A hyper-realistic macro photograph capturing the intricate details of a dandelion, ...



(b) Prompt (excerpt): A photorealistic closeup video of two pirate ships battling each other as they sail inside a steaming cup of coffee. ...

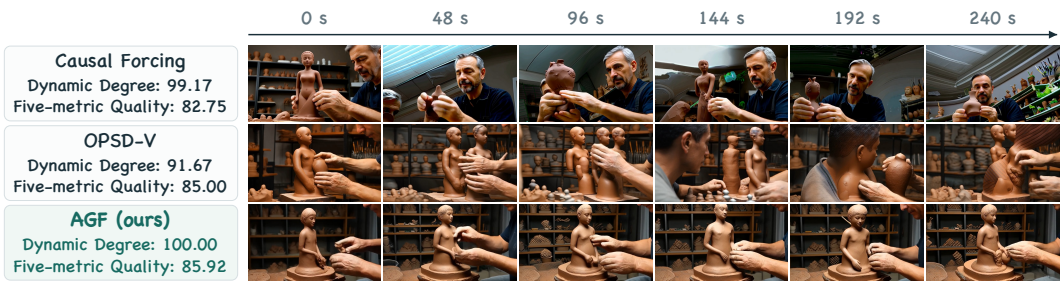
Figure 11: **Loss of fine detail and object structure. (a) Dandelion:** distorted surfaces and colored artifacts replace the fine seed structure in Self-Forcing, while AGF retains the radial organization and visible filaments. **(b) Pirate ships:** Causal Forcing distorts the foam, cup, and ship placement; AGF preserves two recognizable ships within a clearly defined cup.

Figure 11 examines fine detail and object layout. In the dandelion sequence, Self-Forcing's later frames replace the fine seed structure with large distorted surfaces and colored artifacts. AGF retains the recognizable radial structure and fine filaments through the sampled close-ups. In the coffee-cup sequence, Causal Forcing increasingly warps the foam and cup geometry, while the ships lose their stable placement within the cup. AGF keeps both ships, the foam, and the cup rim recognizable as parts of the same miniature scene.

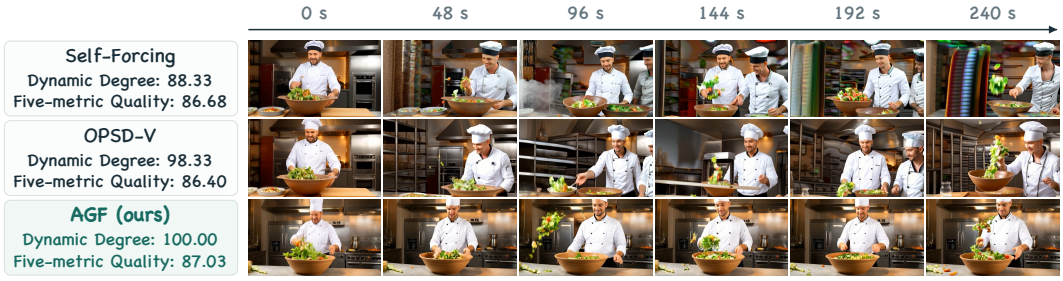
F.2 Coherent Dynamic Scenes

AGF can retain high visual quality while generating dynamic scenes. The examples below pair its high Dynamic Degree with coherent object manipulation and character movement.

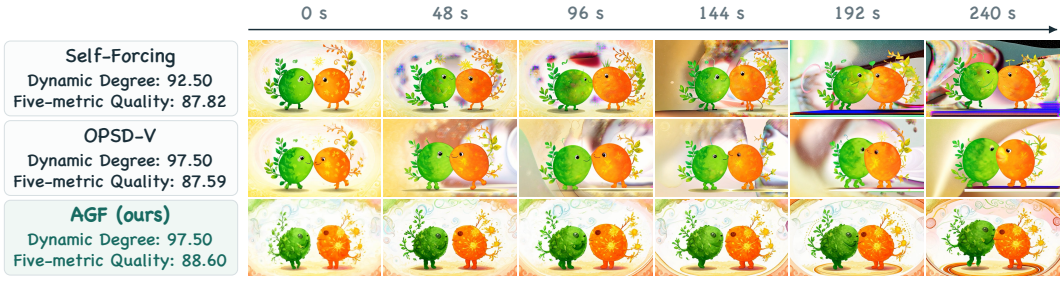
Figure 12 shows hands reshaping clay, a chef tossing salad, and two colored characters changing pose and position. AGF preserves the objects and spatial relations involved in each action: the hands work on one clay figure, tossed ingredients appear above the bowl, and the two characters remain distinct in their pastel setting. The studio, kitchen, and illustrated background remain recognizable through the sampled views.



(a) Prompt (excerpt): A clay model being slowly deformed as it is pressed and molded into a new shape by hand. ...



(b) Prompt (excerpt): A dynamic scene captured in the style of a vibrant food photography shoot, showcasing a chef expertly tossing a salad in a large ceramic bowl. ...



(c) Prompt (excerpt): A whimsical illustration in a vibrant watercolor style depicting two blobs in a passionate dance of love. ...

Figure 12: **Maintaining subjects and settings during dynamic actions. (a) Sculpting:** OPSD-V introduces adjacent copies of the clay figure; Causal Forcing repeatedly reframes the interaction around the sculptor's face, taking the work surface out of view. AGF keeps one figure on the work surface as the hands shape it. **(b) Salad tossing:** Self-Forcing develops repeated chefs and background striping; OPSD-V also duplicates the chef. AGF retains one chef, the bowl, and the kitchen. **(c) Dancing characters:** the alternatives develop background deformations and colored bands, while AGF preserves the green and orange pair and their pastel setting as they move.

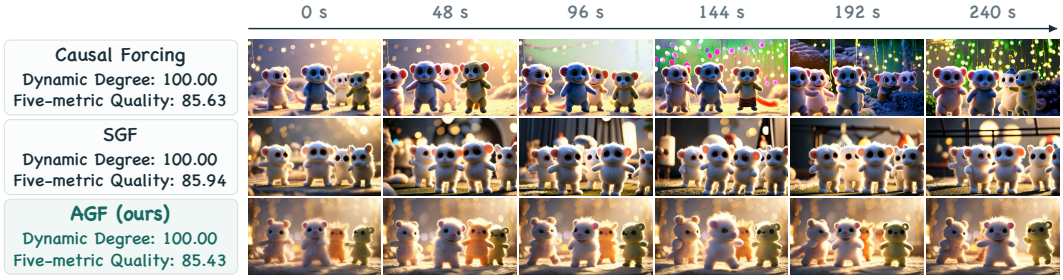
F.3 Subject Duplication and Background Corruption

Similar Q5 scores can conceal substantial differences in visual coherence. In the following cases, other methods achieve Q5 close to or above AGF's while introducing subject copies or background corruption. The aggregate score does not fully reflect these visible differences.

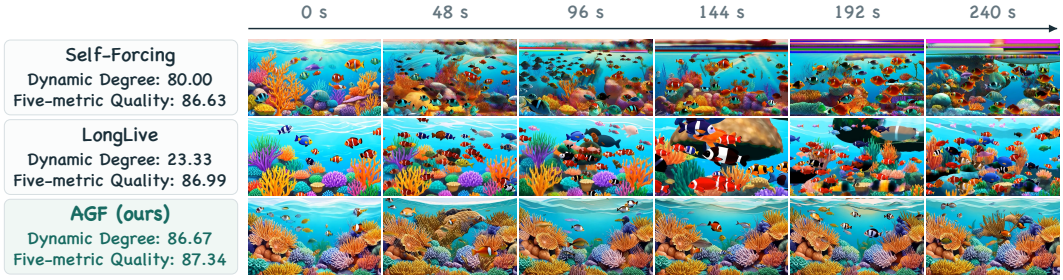
In Figure 13(a), Self-Forcing and OPSD-V introduce additional copies of the woman and alter the room, while AGF retains one main subject and the holiday setting. In Figure 13(b), Causal Forcing replaces the soft bokeh with bright patterned scenery, and SGF introduces curtains and windows. AGF retains the felt appearance of the monsters and their softly blurred surroundings. In Figure 13(c), Self-Forcing develops horizontal colored stripes, while LongLive introduces oversized dark forms that interrupt the reef composition. AGF preserves a recognizable coral environment and layered papercraft texture as the fish change position.



(a) Prompt (excerpt): A vibrant and dynamic digital illustration in the style of a modern holiday card, featuring a young woman with sparkling eyes and a radiant smile as she opens a gift. ...



(b) Prompt (excerpt): A miniature 3D render in an octane engine style depicting adorable wool and felt monsters dancing together in a dreamy, bokeh-filled setting. ...



(c) Prompt (excerpt): A beautifully detailed papercraft illustration of a vibrant coral reef teeming with colorful fish and sea creatures. ...

Figure 13: **Subject copies and background artifacts in long rollouts. (a) Gift:** Self-Forcing and OPSD-V add copies of the woman and alter the room; AGF retains one main subject and the holiday setting. **(b) Felt monsters:** Causal Forcing and SGF replace the softly blurred background with patterned scenery or indoor structures, while AGF retains the soft bokeh. **(c) Papercraft reef:** Self-Forcing adds colored stripes and LongLive introduces oversized dark forms; AGF keeps the coral environment and papercraft texture recognizable.

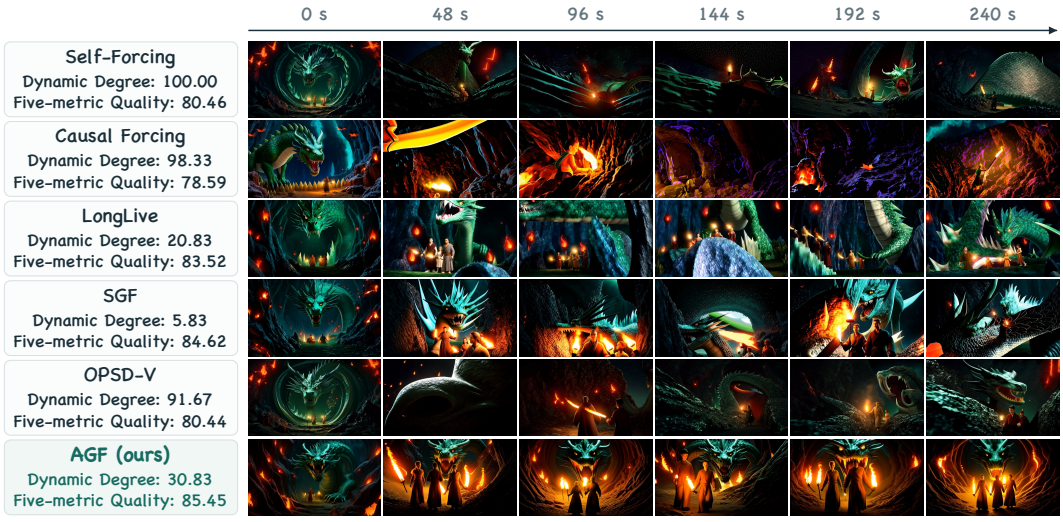
F.4 Complete Six-Method Comparisons

The full six-method comparisons in Figure 14 complement the main example of a church on the coast with two further scenes. In both, AGF combines the highest Q5 with clearer preservation of the main subjects and their spatial relationships.

In the bookshelf scene, close-ups, repeated structures, and colored bands obscure the mushrooms and their surroundings in several other methods. AGF keeps the mushrooms and their placement on the shelf recognizable. In the underworld scene, viewpoint and geometry changes disrupt the composition around the dragon and torch-bearing people; AGF keeps both visible within a recognizable cavern.



(a) Prompt (excerpt): A surreal and haunting digital art piece in a dreamlike style, featuring mushrooms sprouting from the base of a decaying bookshelf. ...



(b) Prompt (excerpt): A dramatic digital painting in the style of an epic fantasy, depicting humans walking into a dragon's open jaws as they descend into the underworld. ...

Figure 14: **Two additional scenes across all six methods. (a) Bookshelf mushrooms:** several alternatives develop repetitive geometry or striping, or shift away from the shelf. AGF preserves a clear view of the mushrooms on the shelf. **(b) Dragon and underworld:** close-ups and distorted geometry disrupt the relation between the dragon and people in several alternatives; AGF keeps the dragon, people, and cavern recognizable together.

G Additional Temporal Stability Results

G.1 Quality over Unperturbed Rollouts

Table 11 gives the absolute local Q5 scores underlying Figure 6. Each entry averages a consecutive 20-second interval over the same 120 prompts, with one video per prompt. AGF stays near its initial quality throughout the four-minute rollout, while all five baselines finish below their initial scores.

Table 11: Local Q5 (%) over unperturbed four-minute generation. Q5 averages the five quality dimensions and excludes Dynamic Degree. Bold marks the highest mean in each interval.

Time (s)	Self-Forcing	OPSD-V	SGF	LongLive	Causal Forcing	AGF (ours)
0–20	84.19	83.75	**84.85**	84.58	82.73	84.71
20–40	83.50	83.06	84.39	83.87	81.68	**84.65**
40–60	82.77	82.62	84.07	83.70	81.35	**84.58**
60–80	82.45	82.57	83.95	83.38	81.29	**84.65**
80–100	82.33	82.58	83.85	83.32	81.45	**84.71**
100–120	82.18	82.41	83.61	83.34	81.29	**84.60**
120–140	82.23	82.48	83.77	83.37	81.37	**84.69**
140–160	81.73	82.50	83.78	83.31	81.29	**84.66**
160–180	81.83	82.39	83.96	83.27	81.07	**84.61**
180–200	81.76	82.41	84.03	83.24	81.05	**84.63**
200–220	81.56	82.31	83.97	83.23	81.38	**84.69**
220–240	81.58	82.32	83.89	83.45	81.27	**84.74**

G.2 Stability after Temporary History Perturbations

We evaluate the same six methods on the 120 prompts with two inference-time perturbations. Both act from 30 to 60 seconds on each newly generated latent block before it is encoded into the history cache; saved outputs are not edited. At 60 seconds, we stop injecting errors and continue generation from the accumulated history until 240 seconds.

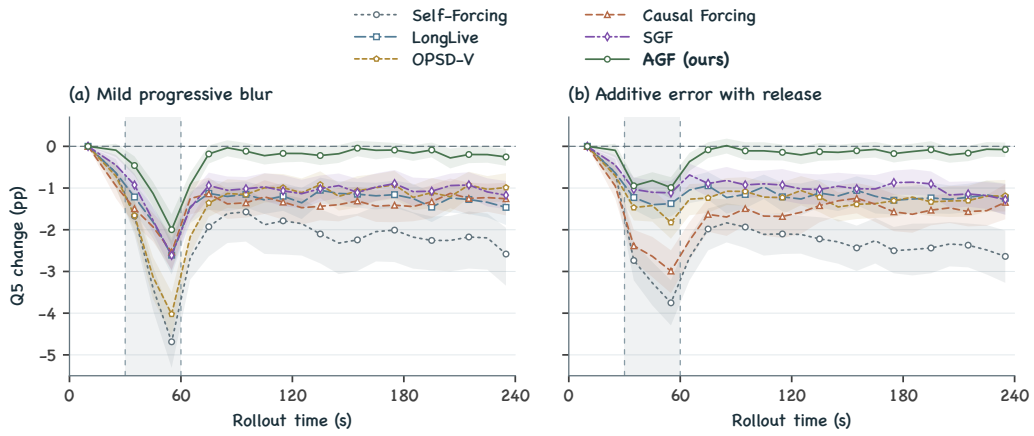


Figure 15: **Generation stability under temporary history perturbations.** Mean local Q5 change from each video's first 20 seconds, over 120 prompts. The first point summarizes 0–20 seconds; subsequent points use 10-second bins. Gray shading marks the 30–60-second intervention. Colored bands show pointwise 95% bootstrap intervals from 2,000 prompt resamples. These trajectories include natural quality drift as well as the effects of the intervention.

Mild progressive blur. For a history block h, we write $\tilde{h} = (1-\alpha)h + \alpha\,\mathrm{AvgPool}_{5\times5}(h)$ to the cache. The spatial mean filter uses reflection padding, and α increases linearly from 0.0625 to 0.25 across the intervened blocks.

Additive error with release. At the onset, we compute a history perturbation that increases the squared error between the next block's prediction and a fixed denoised target from the frozen 14B teacher. We differentiate through all four denoising steps of that next block, sharing the perturbation across the history block's three latent frames. The resulting gradient direction is scaled to a nominal RMS of 0.30 and held fixed for each method–prompt pair; the same error field is added to every new history block during the intervention.

All six methods lose quality during the intervention and recover after it ends (Figure 15). AGF returns closest to its initial level: its mean Q5 change over 90–240 seconds is -0.17 pp after blur

and -0.13 pp after additive error, compared with -2.10 to -1.03 pp and -2.31 to -1.01 pp, respectively, for the five baselines.

H Additional Ablations

The ablations below use the configuration in Table 4 with only the specified setting changed. They use Self-Forcing ODE initialization, batch 64, five-second rollouts, and 200 regularized + 200 pure-AGF iterations. We evaluate the generator's EMA weights at iteration 400 on the same 944/240/120 prompts at 5/60/240 seconds; default AGF scores reuse the main benchmark. All variants use one training run and one video per prompt, with ordinary, unperturbed inference.

H.1 Regularization Variants

Without regularization. The variant without regularization (None in Table 2) uses the generator's EMA weights at iteration 400 and is evaluated on the same 944/240/120 prompts with the common inference settings. It obtains 81.38/77.96 Total at 5/60 seconds and 82.75 Q5 at 240 seconds.

Regularized steps. For regularization over multiple denoising steps, both history conditions use shared noisy inputs from the clean trajectory. We average the per-step MSEs against detached clean predictions. We weight this combined loss using the same 5% target ratio, based on gradient estimates at the model outputs.

Reference models. The default prediction target is the current generator's velocity prediction at the first denoising step, conditioned on the unperturbed history. This prediction is detached from the computation graph. The *Frozen* variant uses the same ODE-initialized generator held fixed throughout training. The *Teacher* variant uses a frozen 14B teacher to generate a full video from the shared initial noise with 25 sampling steps and guidance 3.5. Its output is converted to an equivalent first-step velocity at the actual scheduler noise level; the teacher does not condition on the student's generated history. All three variants apply the regularizer only to the continuation under the perturbed history. They use the same schedule for the target gradient ratio, which reaches 5%, as described in Appendix B.4.

H.2 Alternative ODE Initialization

$\mathrm{AGF}_{\mathrm{CF}}$ starts from Causal Forcing's chunkwise Stage 2 ODE checkpoint. Given the different starting point and optimization characteristics of this initialization, we slightly adjust the training schedule and train for 1,400 iterations in total. The regularization strength ramps from zero at iteration 20 to a 3% output-gradient ratio at iteration 80, holds through iteration 400, and is disabled for iterations 401–1,400. EMA starts at iteration 100 with decay 0.99 and is evaluated at iteration 1,400. Other settings follow Table 4.

Compared with the released Stage 3 Causal Forcing model, all five Q5 dimensions increase (Table 12). Lower Dynamic Degree makes Q6 fall despite higher Q5, reinforcing the need to report visual quality separately from the extent of motion.

Table 12: **Alternative initialization at 240 seconds (%).**

Model	Subject consistency	Background consistency	Motion smoothness	Aesthetic quality	Imaging quality	Dynamic Degree	Q5
CF	94.44	94.82	95.99	55.29	68.41	91.41	81.79
$\mathrm{AGF}_{\mathrm{CF}}$	**97.37**	**96.44**	**98.33**	**59.87**	**71.01**	79.95	**84.60**

I HUMAN EVALUATION DETAILS

The study comprises 240 ranking tasks completed by 12 participants, covering 120 prompts. Each task presents six four-minute videos generated from the same prompt, with method identities hidden. Participants rank all six candidates in order of preference, assigning each rank from 1 to 6 exactly once. Each participant completes 20 tasks.

For each method, we compute its mean rank as the sum of its assigned ranks divided by 240. Figure 16 shows the percentage of assignments at each rank, using the same denominator. This distribution complements the mean by showing how often each method appears near the top or bottom of the ordering.

AGF leads the human evaluation with the best mean rank (2.23), the largest share of first-place rankings (35.8%), and a top-three placement in 84.2% of ranking tasks. This preference trend is broadly consistent with the four-minute quality evaluation in Table 1. Together, these results show that AGF's advantage in automatic quality metrics is also reflected in human judgments of four-minute videos.

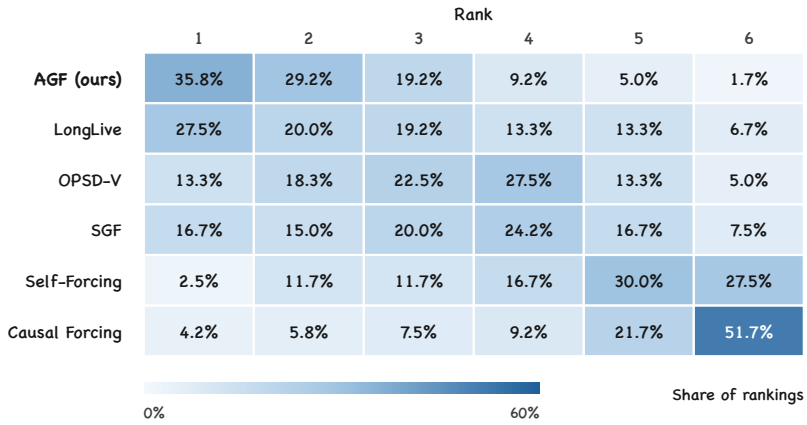


Figure 16: **Distribution of human-assigned ranks.** Each cell shows the percentage of 240 ranking tasks assigning that rank to the method. Rows follow the mean-rank ordering in Figure 7; darker cells indicate a larger share. Percentages are rounded to one decimal place.