TRACE FORCING: ANCHOR TOKENS IN THE KV CACHE FOR TRAINING-FREE LONG VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive video diffusion models generate videos of arbitrary length by conditioning each frame on the preceding frames stored in a KV cache, yet their quality degrades as generation proceeds beyond the training horizon (the video length seen during training). Existing training-free methods attribute this degradation to RoPE temporal positions beyond the trained range and re-index the temporal positions of the cached tokens at inference, but the degradation persists. We reveal that the cached tokens themselves deviate from the tokens the model was trained on, and that the deviated tokens entering the cache induce artifacts in the regions that attend to these tokens. To address this, we introduce Trace Forcing, a training-free method that replaces the raw tokens in the KV cache with anchor tokens, which aggregate the corresponding tokens across frames by an exponential moving average along traces obtained from the similarity between adjacent frames. Each anchor token thus incorporates earlier representations that are less affected by the deviation while remaining consistent with the corresponding content, so that the model conditions on representations closer to the tokens observed during training. We further propose adaptive updating, which assigns each anchor an update rate according to how its content evolves, suppressing the accumulation of deviation on static content while allowing dynamic content to follow recent information. Experiments show that Trace Forcing exhibits less quality degradation than existing training-free methods when generating far beyond the training horizon.

1 INTRODUCTION

Video diffusion models (Blattmann et al., 2023; HaCohen et al., 2024; Ma et al., 2025; Zheng et al., 2025; Gao et al., 2025; Kong et al., 2024; Yang et al., 2025b) have achieved remarkable progress in high-quality video generation, escalating the demand for long videos in applications such as film production (Polyak et al., 2024; Wu et al., 2025; Xu et al., 2025), world simulation (Brooks et al., 2024; Feng et al., 2026; Sun et al., 2025; Agarwal et al., 2025; Wu et al., 2026), and interactive content creation (Hong et al., 2025; Xiong et al., 2026; Gao et al., 2026). However, conventional bidirectional DiT architectures (Team et al., 2025; Kong et al., 2024; Yang et al., 2025b) denoise all frames jointly, so that the video length is fixed by the input latents and extending it incurs a computational cost that grows quadratically with the number of frames. To address this limitation, autoregressive video diffusion (Gao et al., 2024; Yin et al., 2025; Teng et al., 2025; Chen et al., 2025; 2024) has emerged as the dominant paradigm for arbitrary-length video generation, where a causal DiT generates each frame conditioned only on the preceding frames (contexts) stored in a KV cache and extends the video frame by frame (rollout). Since the cache retains only the keys and values of the most recent frames within a fixed-length window, the cost of generating each frame remains constant regardless of video length, which in principle allows the generation of infinitely long videos.

However, autoregressive generation is prone to error accumulation (Huang et al., 2026; Yang et al., 2025a), where small prediction errors in each generated frame enter the cache and make the context progressively noisier, degrading video quality as the rollout proceeds. Existing training-based methods attribute this degradation to a train-test gap, as the model is conditioned on ground-truth contexts at training while on self-generated contexts at inference (Huang et al., 2026), and is trained on short clips while producing long videos at inference (Yang et al., 2025a; Cui et al., 2026a). These methods reduce the gap by training on self-generated contexts or longer videos, but only within the training horizon (the video length seen during training), so that degradation of video quality

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.

still occurs beyond the horizon, and extending the horizon requires retraining on longer videos at a considerable cost. Instead, recent training-free methods address the problem from the opposite direction, aligning the inference-time contexts beyond the training horizon with the contexts observed during training. Attributing the degradation to RoPE (Su et al., 2024) temporal positions beyond the trained range, these methods (Yesiltepe et al., 2026; Kim et al., 2026) re-index the temporal positions of the cached tokens into the trained range at inference. However, video quality still degrades even after the temporal positions are re-indexed, suggesting that the mismatch stems not only from the positions but also from the cached tokens themselves. To verify this, we take the tokens generated within the training horizon as a reference, since the model is trained on rollouts of this length, and measure how far the tokens generated beyond the horizon deviate from this reference while excluding the effect of temporal positions. We observe that the deviation increases with the video length (the orange line in Figure 1 (a)), and that once such deviated tokens enter the cache, the regions that attend strongly to these tokens exhibit artifacts (Figure 2). These observations indicate that *the cached tokens themselves deviate from the distribution the model was trained on*, and that this deviation is what degrades the quality of generated frames.

In this paper, we introduce Trace Forcing, a training-free method that mitigates the deviation of the tokens in the KV cache. The key insight from Figure 1 (a) is that the deviation increases over the rollout, so a representation that incorporates earlier tokens is less susceptible to the deviation than the raw token alone. We therefore propose the **anchor token**, which maintains an exponential moving average (EMA) of the cached tokens, gradually integrating each token that enters the cache into the aggregated representation. The anchor token is then used as the context in the cache instead of the raw token, providing a less deviated representation for the subsequent frames. Thus, the anchor token must also represent the raw token faithfully, which requires aggregating the tokens that correspond to the raw token across frames. We therefore trace tokens across frames based on the similarity between raw tokens of adjacent frames, such that each anchor token is continually updated along the resulting trace. Each anchor token thus becomes the aggregated representation of the corresponding content up to the latest raw token updated into it, and is used in the cache in place of that raw token. As a result, the cache holds representations that are less susceptible to the deviation while still reflecting the corresponding content, suppressing artifacts in the generated frames. The update rate of the EMA, however, involves a trade-off, as a small rate retains stale content whereas a large rate inherits the deviation, and we thus propose **adaptive updating**, which assigns each anchor an update rate according to how its content evolves. This design reflects how the deviation accumulates across anchors: static content persists in the same anchor the longest and thus requires a small rate, whereas dynamically evolving content changes faster than the deviation accumulates and thus admits a large rate to absorb recent information. Together, these two components enable Trace Forcing to condition the model on less deviated anchor tokens that evolve with the video content, thereby preserving the generation capability of autoregressive models over long rollouts.

To evaluate Trace Forcing on long video generation beyond the training horizon, we apply it to Self-Forcing (Huang et al., 2026) and LongLive (Yang et al., 2025a) as base models, which are trained on 5-second and 1-minute videos, respectively, and generate videos of up to 8 minutes. Under this setting, Trace Forcing achieves the highest VBench-Long average score and the lowest quality drift (change in quality over the rollout) among training-free methods on both base models. Our contributions are summarized as follows:

- We reveal that the tokens cached beyond the training horizon increasingly deviate from the tokens cached within it, and that the deviated tokens both degrade the quality of the generated frames and propagate the deviation to subsequently generated tokens.
- We propose Trace Forcing, a training-free method that uses *anchor tokens*, aggregated from the raw tokens along traces across frames and updated at rates adapted to how their content evolves (*adaptive updating*), as the context in the KV cache in place of the raw tokens.
- Trace Forcing achieves the highest VBench-Long average score and the lowest quality drift among training-free methods on both base models.

2 Related Works

Autoregressive Video Diffusion. Autoregressive video diffusion produces a video chunk by chunk with a causal transformer, where each chunk is conditioned on the KV cache of its predecessors.

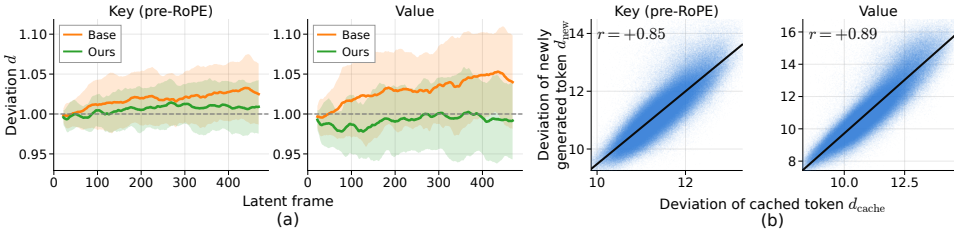

Figure 1: (a) Per-frame deviation of cached tokens from the distribution of tokens within the training horizon, normalized by the first window, for Self-Forcing (orange) and Trace Forcing (green). Shaded regions indicate one standard deviation across prompts. (b) Deviation d_{new} of the token generated at each query position against the deviation d_{cache} of the cached tokens attended by the query.

CausVid (Yin et al., 2025) distills a bidirectional DiT into a causal generator via DMD (Yin et al., 2024), and Self-Forcing (Huang et al., 2026) narrows the train-test gap by rolling the model out on its own samples during training. Subsequent work extends the reachable horizon through additional training, including teacher-guided error correction in Self-Forcing++ (Cui et al., 2026a), rolling-window denoising in Rolling Forcing (Liu et al., 2026), ODE-initialized distillation in Causal Forcing (Zhu et al., 2026), streaming long tuning in LongLive (Yang et al., 2025a), and long-context streaming tuning in Context Forcing (Chen et al., 2026b). In contrast, Trace Forcing requires no additional training and enables generation far beyond the bounded training horizon of the model while suppressing the degradation of video quality.

Training-Free Long-Horizon Generation. Since retraining a generator for every target length is costly, a parallel line of work extends the horizon of a frozen model at inference time, mostly by handling the positional indices beyond the trained range. ∞-RoPE (Yesiltepe et al., 2026) and MemRoPE (Kim et al., 2026) keep the indices within the trained range with block-relative coordinates, Deep Forcing (Yi et al., 2025) re-aligns the temporal phase of its sink tokens, Head Forcing (Tian et al., 2026) re-encodes positions per head, LoL (Cui et al., 2026b) jitters the RoPE frequencies across heads, and FLEX (Li et al., 2026b) modulates the under-trained low-frequency components of 3D-RoPE. Complementary to re-indexing, cache policies decide which tokens to keep: Deep Forcing selects cached tokens by their attention from recent frames, Head Forcing and Pyramid Forcing (Chen et al., 2026a) assign retention rules per head, and Rolling Sink (Li et al., 2026a) redesigns sink maintenance (Xiao et al., 2024) from a systematic analysis of the cache. MemRoPE goes beyond selection by compressing evicted frames into EMA tokens over unrotated keys. None of these methods, however, aggregates the cached tokens along traces of corresponding content across frames, as re-indexing and selection leave the token representations as generated, and the EMA in MemRoPE aggregates evicted frames position-wise. Trace Forcing instead uses such traced representations, which are less susceptible to the deviation, as the context in the cache in place of the raw tokens, thereby mitigating the artifacts that arise beyond the training horizon.

3 Preliminaries

Autoregressive Video Diffusion. Autoregressive video diffusion models (Huang et al., 2026; Yang et al., 2025a; Cui et al., 2026a) produce each latent frame conditioned on previously generated frames. In practice, frames are generated in chunks, each treated as a single frame hereafter. A causal diffusion transformer G_θ generates the f-th frame $\mathbf{x}^f$ by conditioning on the previously generated frames $\mathbf{x}^{<f} = (\mathbf{x}^1, \ldots, \mathbf{x}^{f-1})$, as follows:

$$\hat{\mathbf{x}}^f = G_\theta(\mathbf{x}^f_{\sigma_t}, \sigma_t, \mathbf{K}^{<f}, \mathbf{V}^{<f}), \tag{1}$$

where $\mathbf{x}^f_{\sigma_t}$ is the noisy frame at noise level σ_t of denoising step t, $\hat{\mathbf{x}}^f$ is the clean frame predicted at that step, which is re-noised to σ_{t-1} for the next step, and $\mathbf{K}^{<f}, \mathbf{V}^{<f}$ denote the keys and values of $\mathbf{x}^{<f}$ retained in the cache (text conditioning omitted for clarity). The f-th frame is iteratively denoised across a small number of noise levels $\sigma_T > \cdots > \sigma_1 > \sigma_0 = 0$, starting from $\mathbf{x}^f_{\sigma_T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

KV Cache Rollout. We follow the KV cache rollout of Self-Forcing (Huang et al., 2026). During generation, the cached keys and values $\mathbf{K}^{<f}, \mathbf{V}^{<f}$ are concatenated with the keys and values of the



Figure 2: For each generated frame (top), the maps visualize $D_{\text{cache}}(c)$, the deviation of the cached tokens weighted by the attention weights of each query at c, measured on the keys (middle) and values (bottom), where darker regions indicate queries that attend more to deviated tokens.

frame being generated at every self-attention layer, so that the queries of the current frame attend to the past context without recomputing it. To obtain the keys and values to be cached, the generated frame $\mathbf{x}^f$, the clean frame predicted at the final denoising step, is forwarded once more through $G_\theta(\mathbf{x}^f, \sigma_0, \mathbf{K}^{<f}, \mathbf{V}^{<f})$ at the clean noise level $\sigma_0 = 0$, and the resulting $\mathbf{K}^f, \mathbf{V}^f$ are cached as the context for the frames that follow. Since storing all past keys and values is infeasible for long videos, the cache is maintained as a sliding window that evicts the keys and values of the oldest frames in a first-in-first-out manner. To stabilize long rollouts, recent methods additionally retain the initial frames as attention sinks (Xiao et al., 2024; Han et al., 2023; Gu et al., 2025; Darcet et al., 2024; Barbero et al., 2025; Sun et al., 2024) that persist in the cache (Yang et al., 2025a; Liu et al., 2026; Yi et al., 2025; Kim et al., 2026).

4 Analysis of Token Deviation in the KV Cache

When autoregressive diffusion models generate videos beyond the training horizon, artifacts such as oversaturation, identity distortion, and repetitive textures are frequently observed. Existing methods (Yesiltepe et al., 2026; Kim et al., 2026) attribute this degradation to the RoPE temporal positions unseen during training, and re-index the temporal positions into the trained range. However, the degradation is still observed after re-indexing, indicating that re-indexing alone does not resolve the problem. We hypothesize that the degradation stems not only from the unseen temporal positions but also from the cached tokens themselves departing from the tokens the model was trained on.

Token Deviation beyond the Training Horizon. To examine this hypothesis, we regard the cached tokens generated within the training horizon as the tokens the model was trained on, and quantify how far the cached tokens generated beyond the horizon deviate from the cached tokens within the horizon. We use Self-Forcing (Huang et al., 2026) as the base model in this analysis. Since Self-Forcing is trained on 21 latent frames, the key and value tokens cached within the training horizon, collected from a separate set of prompts, serve as the reference. The deviation d of each token cached beyond the horizon is then measured as the Mahalanobis distance to the reference. The deviation is computed independently for each attention head across all layers. To exclude the effect of unseen RoPE temporal positions, the videos are generated with the block-relative indices of Kim et al. (2026), and the deviation is computed on pre-RoPE keys (values are not rotated). The details of the computation are provided in Appendix A. Figure 1 (a) plots the deviation of each latent frame, averaged over all heads and tokens, then normalized by the first window. As shown, the deviation of the baseline (orange line) grows as generation proceeds beyond the training horizon.

Propagation of the Deviation through the KV Cache. We then analyze how the deviated tokens in the KV cache affect the generation. For each query of the current frame, we compute the attention weights over the keys in the cache, and take the weighted sum of the deviation of each key and value with these weights. Figure 2 visualizes the result D_{cache} as a map, where the regions of high weighted deviation coincide with artifacts in the generated frame. We further measure how the deviation d_{cache} of the cached keys and values attended by the current query relates to the deviation d_{new} of the key and value newly generated at the query position (details in Appendix A). In Figure 1 (b), the two

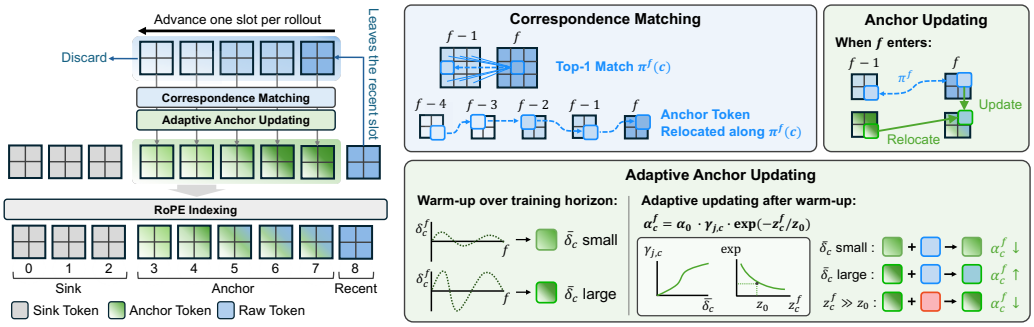

Figure 3: Overview of Trace Forcing. During the rollout, each raw token leaving the recent slot of the cache updates the anchor tokens, so that the model conditions on anchor tokens holding less deviated representations rather than on the deviated raw tokens. Correspondence matching between adjacent frames identifies the anchor token that each raw token updates, and adaptive updating assigns each anchor an update rate adapted to its content.

deviations are strongly correlated (Pearson $r > 0.8$), suggesting that the deviation propagates through the cache into the newly generated key and value, which in turn enter the cache.

These observations indicate that, beyond the training horizon, the cached tokens deviate from those observed during training, and that the deviated tokens both induce artifacts and propagate the deviation to the newly generated tokens that enter the cache. Trace Forcing is therefore designed to keep the cached tokens close to those observed during training, and the green line in Figure 1 (a) shows that the deviation is effectively reduced.

5 Trace Forcing

Trace Forcing consists of two components, the **anchor token** and **adaptive updating**, described in Section 5.1 and Section 5.2, respectively. Figure 3 provides an overview of Trace Forcing.

5.1 Anchor Token

As observed in Section 4, the deviation of the cached tokens tends to increase over the rollout, which suggests that a representation incorporating earlier tokens would be less susceptible to the deviation than the raw token alone. Trace Forcing therefore maintains such aggregated representations, which we call anchor tokens, in the KV cache in place of the raw tokens.

Anchor Tokens. We formulate each anchor token as an exponential moving average (EMA) (Hunter, 1986) of the cached tokens, which aggregates earlier tokens with exponentially decaying weights while emphasizing the latest one. Specifically, the cache holds A anchor slots indexed by $j \in \{1, \dots, A\}$ from the oldest to the most recent, and the j-th anchor holds a pair of key and value $\mathbf{k}_{j,c}^{\text{anc}}, \mathbf{v}_{j,c}^{\text{anc}} \in \mathbb{R}^{1\times D}$ at each spatial position $c \in \mathcal{I}^C = \{1, \dots, C\}$ of a frame, where D denotes the head dimension. We omit the anchor index j for simplicity when unambiguous. Given the raw token $\mathbf{k}_c^f, \mathbf{v}_c^f$ of frame f entering an anchor slot, each anchor token is updated as follows:

$$\mathbf{k}_c^{\text{anc}} \leftarrow (1-\alpha)\,\mathbf{k}_{\pi^f(c)}^{\text{anc}} + \alpha\,\mathbf{k}_c^f, \qquad \mathbf{v}_c^{\text{anc}} \leftarrow (1-\alpha)\,\mathbf{v}_{\pi^f(c)}^{\text{anc}} + \alpha\,\mathbf{v}_c^f, \tag{2}$$

where α denotes the update rate, and the correspondence map $\pi^f : \mathcal{I}^C \to \mathcal{I}^C$ determines which anchor is updated by each entering token $\mathbf{k}_c^f, \mathbf{v}_c^f$ of frame f, both shared between the key and value anchors since each key is aligned with its value. The correspondence map is required because the content moves across frames, so that updating the anchor at a fixed position would blend tokens of different contents into one anchor and compromise the representation. To obtain the correspondence map, we trace tokens across frames by matching each raw token of frame f to the raw tokens of frame $f-1$ based on the cosine similarity of the rotary-embedded keys $\tilde{\mathbf{k}}$, as follows:

$$\pi^f(c) = \arg\max_{c' \in \mathcal{I}^C} \cos\big(\tilde{\mathbf{k}}_c^f,\, \tilde{\mathbf{k}}_{c'}^{f-1}\big), \tag{3}$$

where c' ranges over the spatial positions of frame $f-1$, and the matching is performed independently per layer and head to account for the distinct roles of the heads. Since the deviation emerges gradually, adjacent frames remain closely aligned even after the tokens have deviated, ensuring reliable correspondence. Based on the correspondence map, each anchor token is updated along the matched tokens across frames, and represents the corresponding content of the entering token in the cache.

Anchor Tokens in the Cache. With the anchor token, the KV cache is composed of three components, *Sink*, *Anchor*, and *Recent*, with sizes of S, A, and R slots, respectively:

$$\mathbf{K} = \left[\, \mathbf{K}^{\text{sink}} \parallel \mathbf{K}^{\text{anc}} \parallel \mathbf{K}^{\text{rct}} \,\right], \tag{4}$$

where $[\cdot \parallel \cdot]$ denotes concatenation along the temporal axis, $\mathbf{K}^{\text{anc}} \in \mathbb{R}^{AC \times D}$ consists of the anchor tokens $\mathbf{k}^{\text{anc}}_{j,c}$ of the A anchor slots, and $\mathbf{V}$ is composed likewise. Whenever a frame enters the cache, the raw tokens evicted from the recent slot advance one anchor slot at a time from $j = A$ to $j = 1$, and the raw tokens passing through each anchor slot update the anchor token rather than being stored. Each anchor token is assigned a RoPE index by its slot and position: the temporal index is fixed by its slot j throughout the rollout, while the spatial index follows the relocated position in the frame that the anchor currently represents. Assigning RoPE to the anchor tokens, however, is not straightforward, since the EMA mixes tokens rotated at different positions, and updating rotary-embedded tokens as in Equation 2 yields representations that correspond to no valid position. We therefore adopt the indexing scheme of Kim et al. (2026), which stores all keys in their unrotated form and applies block-relative indices online at every attention call. Under this scheme, the sink, anchor, and recent slots occupy the temporal indices $[0, S)$, $[S, S+A)$, and $[S+A, S+A+R)$, respectively.

5.2 Adaptive Updating

The anchor tokens in Section 5.1 are updated with a uniform rate α across all positions. However, this involves a trade-off: a small rate preserves the earlier representations but holds stale content, whereas a large rate follows the content but inherits the deviation. We therefore propose adaptive updating, which assigns each anchor an update rate according to how its content evolves. This design reflects how the deviation accumulates across anchors: static content persists in the same anchor the longest and thus requires a small rate, whereas dynamically evolving content changes faster than the deviation accumulates and thus admits a large rate.

Anchor Tolerance. To adapt the rate to the content, each anchor first needs a reference of how much its content changes in the absence of the deviation. Since the model generates reliably within the training horizon, the change that each anchor token undergoes within this horizon reflects the genuine evolution of its content. We therefore update the anchor tokens with a uniform rate α_{warm} over the training horizon of N_{train} frames, and record the dissimilarity between each anchor and the tokens updated into it as the reference change $\bar{\delta}_c$:

$$\bar{\delta}_c = \sqrt{\frac{1}{|\mathcal{F}_{\text{train}}|} \sum_{f \in \mathcal{F}_{\text{train}}} \left(\delta_c^f\right)^2}, \qquad \delta_c^f = 1 - \max\Big(0,\ \frac{\mathbf{k}^{\text{anc}\top}_{\pi^f(c)}\, \mathbf{k}_c^f}{\|\mathbf{k}^{\text{anc}}_{\pi^f(c)}\|\, \|\mathbf{k}_c^f\|}\Big), \tag{5}$$

where $\mathcal{F}_{\text{train}}$ denotes the frames updated into the anchor within the training horizon N_{train}, and δ_c^f is the cosine distance between the anchor token $\mathbf{k}^{\text{anc}}_{\pi^f(c)}$ and the entering token $\mathbf{k}_c^f$, with negative cosine similarity truncated to zero. The reference $\bar{\delta}_c$ is relocated along π^f together with the anchor. As a root mean square, $\bar{\delta}_c$ grows with both the mean and the variance of δ_c^f, capturing both the mean degree of change in the content and its variability.

Adaptive Update Rate. With $\bar{\delta}_c$ as the degree of change that each anchor should tolerate thereafter, the update rate serves two purposes: a **content-adaptive gate** assigns a small rate to static content and a large rate to dynamically evolving content according to $\bar{\delta}_c$, and a **deviation rejection** suppresses the update from an entering token whose change far exceeds $\bar{\delta}_c$, which signals a deviated token rather than a genuine change. The update rate combines the two as follows:

$$\alpha_c^f = \alpha_0 \cdot \gamma_{j,c} \cdot \exp\big(-z_c^f / z_0\big), \qquad \gamma_{j,c} = \frac{(\bar{\delta}_c)^m}{(\bar{\delta}_c)^m + (\delta_{0,j})^m}, \qquad z_c^f = \frac{\delta_c^f}{\max(\bar{\delta}_c, \epsilon)}, \tag{6}$$

Table 1: Quantitative results on 240- and 480-second video generation, evaluated with VBench-Long. Each method is applied on top of two base models, Self-Forcing and LongLive. The best results are highlighted in **bold**, and the second-best results are underlined.

Method	Imaging Quality ↑	Aesthetic Quality ↑	Background Consistency ↑	Motion Smoothness ↑	Subject Consistency ↑	Dynamic Degree ↑	Average ↑
240 seconds							
Results on Self-Forcing							
Self-Forcing	57.61	44.13	**96.21**	97.75	95.61	28.02	69.89
Deep Forcing	64.32	56.10	95.70	97.80	96.36	46.32	76.10
∞-RoPE	67.93	54.98	95.60	97.53	96.27	62.02	79.05
MemRoPE	67.67	54.60	95.81	97.72	**96.45**	61.59	78.97
Ours	**68.57**	**56.62**	95.86	**98.23**	96.44	**62.81**	**79.75**
480 seconds							
Results on Self-Forcing							
∞-RoPE	69.70	55.29	96.03	98.27	97.24	45.96	77.08
MemRoPE	**69.99**	54.99	96.22	98.57	97.20	**47.43**	77.40
Ours	69.78	**57.52**	**96.35**	**98.67**	**97.32**	46.07	**77.62**
Results on LongLive							
∞-RoPE	66.50	56.96	96.38	**98.94**	97.51	45.94	77.04
MemRoPE	68.21	57.58	**96.50**	98.82	97.69	44.94	77.29
Ours	**69.54**	**58.32**	96.43	98.84	**97.77**	**54.90**	**79.30**

Table 2: Quality drift, measured by the fluctuation σ (standard deviation of scores across 2-second clips) and the cumulative degradation Δ (absolute score difference between the first and last 5 seconds). The best results are highlighted in **bold**, and the second-best results are underlined.

	Self-Forcing 240s				Self-Forcing 480s				LongLive 480s			
	Imaging Quality		Aesthetic Quality		Imaging Quality		Aesthetic Quality		Imaging Quality		Aesthetic Quality	
Method	$\sigma\downarrow$	$\Delta\downarrow$	$\sigma\downarrow$	$\Delta\downarrow$	$\sigma\downarrow$	$\Delta\downarrow$	$\sigma\downarrow$	$\Delta\downarrow$	$\sigma\downarrow$	$\Delta\downarrow$	$\sigma\downarrow$	$\Delta\downarrow$
Base	10.42	14.40	9.19	21.90	*not applicable*							
Deep Forcing	4.12	5.13	**3.35**	5.72								
∞-RoPE	4.16	5.13	4.30	7.25	4.36	5.54	4.78	10.39	4.80	6.61	4.08	6.53
MemRoPE	4.39	4.93	4.57	8.01	4.33	5.25	4.69	10.00	3.69	3.52	3.65	5.52
Ours	**3.89**	**4.55**	3.80	**5.24**	**3.43**	**2.63**	**3.61**	**5.65**	**2.93**	**2.04**	**3.06**	**5.10**

where α_0 is a global scale. The content-adaptive gate $\gamma_{j,c}$ increases with $\bar{\delta}_c$ from zero to one, where the threshold $\delta_{0,j}$ at which $\gamma_{j,c}$ reaches one half is set per anchor slot, decreasing from the oldest to the most recent anchor, and m controls the sharpness of the transition. The exponential term implements the deviation rejection, decaying with the normalized dissimilarity z_c^f of the entering token relative to $\bar{\delta}_c$, where z_0 sets the width of the decay and ϵ prevents division by a vanishing $\bar{\delta}_c$ (details of both terms in Appendix C). The resulting α_c^f is used in place of α in Equation 2.

6 Experiments

6.1 Experimental Settings

Implementation Details. We evaluate Trace Forcing on two autoregressive video diffusion models, Self-Forcing (Huang et al., 2026) and LongLive (Yang et al., 2025a). Following their default settings, videos are generated in chunks of 3 latent frames with a 4-step denoising schedule at timesteps $\{1000, 750, 500, 250\}$. The KV cache consists of $S = 3$ sink, $A = 5$ anchor, and $R = 1$ recent slots. The global scale α_0 is set to 0.2 for Self-Forcing and 0.02 for LongLive, and the per-anchor thresholds $\delta_{0,j}$ are spaced by 0.05 and 0.02, respectively. The remaining settings are listed in Appendix D.

Evaluation. To assess generation beyond the training horizon, we evaluate Self-Forcing (trained on 5-second videos) at 240 and 480 seconds, and LongLive (trained on 60-second videos) at 480 seconds. Following MemRoPE (Kim et al., 2026), we report VBench-Long (Huang et al., 2024) on MovieGenBench (Polyak et al., 2024) prompts refined by Qwen2.5-7B-Instruct, using 128 prompts for 240-second and 20 prompts for 480-second generation. We also measure quality drift with two measures: the cumulative degradation (Δ) as the absolute difference in aesthetic and imaging quality between the first and last five seconds following Liu et al. (2026) and Yi et al. (2025), and the

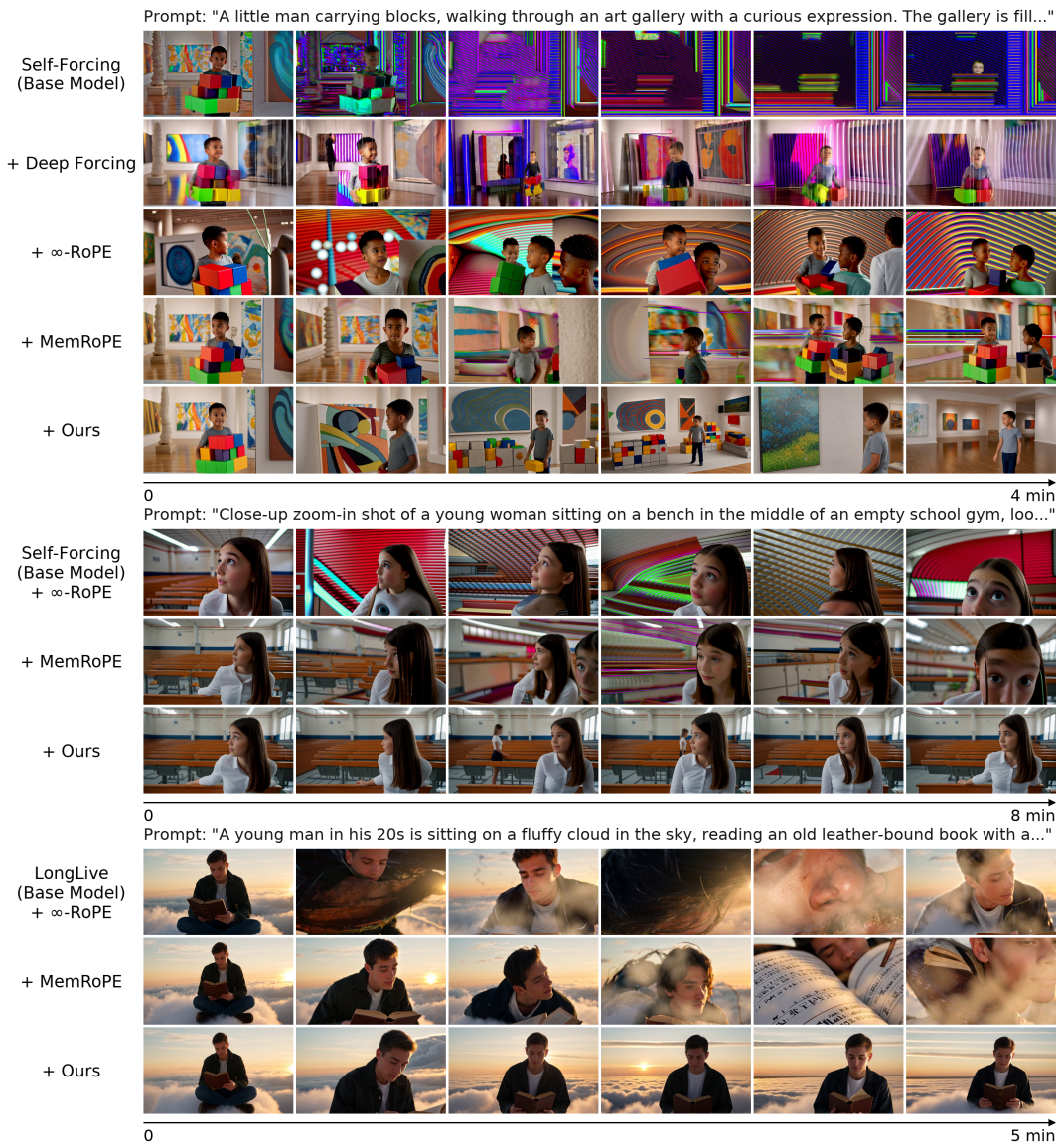

Figure 4: Qualitative comparison on Self-Forcing at 240 seconds (top), Self-Forcing at 480 seconds (middle), and LongLive at 480 seconds (bottom).

fluctuation (σ) as the standard deviation of scores across 2-second clips following Lu et al. (2026). All ablations are conducted on Self-Forcing at 240 seconds with 20 randomly sampled prompts.

6.2 Comparison with Existing Training-Free Methods

We compare Trace Forcing with existing training-free methods, Deep Forcing (Yi et al., 2025), ∞-RoPE (Yesiltepe et al., 2026), and MemRoPE (Kim et al., 2026), on top of the same base models. Since 480-second generation exceeds the maximum position of 1024 latent frames that the base models accept, the base models and Deep Forcing, which rely on absolute indices, are excluded from this setting, leaving ∞-RoPE and MemRoPE for comparison.

Quantitative Results on VBench-Long. Table 1 shows that Trace Forcing achieves the highest average score in every setting across base models and horizons, with the best aesthetic and imaging quality in most settings. The only exception is imaging quality on Self-Forcing at 480 seconds, slightly lower than MemRoPE by 0.21, where Trace Forcing nonetheless shows a lower imaging-quality Δ and σ in Table 2. Trace Forcing also restores the dynamic degree, which drops to 28.02 on

Table 3: Ablation study on each component, the anchor token and adaptive updating, on 20 prompts.

Method	Imaging Quality		Aesthetic Quality		Dynamic Degree↑
	Score↑	$\sigma\downarrow$	Score↑	$\sigma\downarrow$	
Self-Forcing	68.29	4.68	56.54	4.25	<u>50.83</u>
+ Anchor Token	<u>68.75</u>	<u>3.02</u>	<u>58.39</u>	<u>3.28</u>	49.35
+ Adaptive Updating (**Ours**)	**69.61**	**2.80**	**58.90**	**3.27**	**53.79**

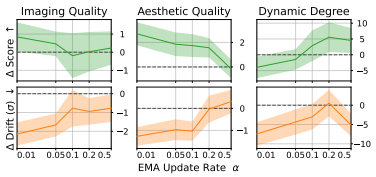

Figure 5: Effect of the update rate α without adaptive updating.

the base Self-Forcing at 240 seconds, and surpasses the baselines on LongLive by a large margin (54.90 vs. at most 45.94), indicating that our approach does not suppress dynamic motion. Trace Forcing is the best or second-best on the remaining metrics, background and subject consistency and motion smoothness, in every setting, so the gain in fidelity does not compromise temporal coherence.

Quantitative Results on Quality Drift. In Table 2, Trace Forcing achieves the lowest cumulative degradation (Δ) and fluctuation (σ) in nearly all settings. Notably, the quality drift of the baselines on Self-Forcing generally increases from 240 to 480 seconds, whereas that of Trace Forcing remains at a similar level, although the prompt sets differ, indicating that our approach scales to longer horizons without further degradation. This stability is consistent with the reduced deviation in Figure 1 (a): keeping the cached tokens close to the training distribution prevents the deviation from accumulating, so the quality holds over the rollout rather than degrading. The only exception is the aesthetic σ on Self-Forcing at 240 seconds, slightly higher than Deep Forcing, where Trace Forcing still shows a lower Δ and higher aesthetic quality in Table 1.

Qualitative Results. Figure 4 shows frames sampled at uniform intervals for the baselines and Trace Forcing. The base Self-Forcing exhibits increasingly severe color shift and texture repetition toward the later frames and eventually collapses into unrecognizable content, and the other baselines alleviate but do not eliminate these artifacts. In the 240-second result on Self-Forcing, the baselines produce repeated high-contrast lines across the background, whereas Trace Forcing avoids this artifact while maintaining dynamic motion. In the 480-second result, the baselines further corrupt the semantic structure of the girl, whereas Trace Forcing keeps the semantics intact. On LongLive, the baselines suffer from structural collapse, whereas Trace Forcing preserves the semantics of the man, suggesting that the anchor tokens updated along valid correspondences faithfully retain the content.

6.3 Ablation Analysis

Table 3 adds each component in turn, comparing the anchor token alone at $\alpha = 0.05$, which matches the effective update rate of the full model at $\alpha_0 = 0.2$, so that the comparison isolates the effect of adaptive updating from that of the update rate, which trades image fidelity against dynamics. The anchor token improves image fidelity over the base model and reduces the fluctuation, but its uniform rate lowers the dynamic degree below the base, as it binds dynamic content to the past as strongly as static content. Adaptive updating resolves this by letting dynamic content absorb recent information while binding static content to the past, so that Trace Forcing further improves the video quality while raising the dynamic degree above the base model. Figure 5 plots imaging quality, aesthetic quality, and dynamic degree, along with their fluctuation (σ), as a function of α without adaptive updating. As α decreases, imaging and aesthetic quality rise and their fluctuation falls, indicating that the anchor token suppresses the deviation as the update is reduced, whereas the dynamic degree declines as the anchors become bound to the past, which necessitates adaptive updating.

7 Conclusion

We showed that, beyond the training horizon, the cached tokens deviate from those observed during training, and that the deviation degrades the quality of generated frames and propagates through the KV cache to subsequently generated tokens. We introduced Trace Forcing, a training-free method that conditions the model on anchor tokens, which aggregate earlier tokens of corresponding content along traces across frames and are less deviated than raw tokens, with the update rate of each anchor adapted to its content. Trace Forcing achieves the highest VBench-Long average score and the lowest quality drift among training-free methods on both base models, demonstrating that keeping the cached tokens close to those observed during training mitigates degradation in long-horizon generation.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing of the manuscript and to evaluate the proposed methodology and experimental design. Separately, the experimental protocol in Section 6 uses an LLM for prompt refinement and a VLM for visual stability evaluation.

REFERENCES

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

Federico Barbero, Alvaro Arroyo, Xiangming Gu, Christos Perivolaropoulos, Michael Bronstein, Petar Veličković, and Razvan Pascanu. Why do llms attend to the first token? *arXiv preprint arXiv:2504.02732*, 2025.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Leo Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 1(8):1, 2024.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

Jiayu Chen, Junbei Tang, Wenbiao Zhao, Maoliang Li, Jiayi Luo, Zihao Zheng, Jiawei Yang, Guojie Luo, and Xiang Chen. Pyramid forcing: Head-aware pyramid kv cache policy for high-quality long video generation. *arXiv preprint arXiv:2605.13111*, 2026a.

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*, 2026b.

Jiaxing 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. In *International Conference on Learning Representations*, volume 2026, pp. 85802–85822, 2026a.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Lol: Longer than longer, scaling video generation to hour. *arXiv preprint arXiv:2601.16914*, 2026b.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International conference on learning representations*, volume 2024, pp. 2632–2652, 2024.

Ruili Feng, Han Zhang, Zhilei Shu, Zhantao Yang, Longxiang Tang, Zhicai Wang, Andy Zheng, Jie Xiao, Zhiheng Liu, Ruihang Chu, et al. The matrix: Infinite-horizon world generation with real-time moving control. *Advances in Neural Information Processing Systems*, 38:87318–87344, 2026.

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.

Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, et al. Seedance 1.0: Exploring the boundaries of video generation models. *arXiv preprint arXiv:2506.09113*, 2025.

Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, et al. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 2026.

Google DeepMind. Gemini 3.1 pro model card. `https://deepmind.google/models/model-cards/gemini-3-1-pro/`, February 2026. Accessed 2026-09-25.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *International Conference on Learning Representations*, volume 2025, pp. 97114–97144, 2025.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. Lm-infinite: Simple on-the-fly length generalization for large language models. 2023.

Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, et al. Relic: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818. IEEE, 2024.

J Stuart Hunter. The exponentially weighted moving average. *Journal of quality technology*, 18(4): 203–210, 1986.

Xuan Ju, Yiming Gao, Zhaoyang Zhang, Ziyang Yuan, Xintao Wang, Ailing Zeng, Yu Xiong, Qiang Xu, and Ying Shan. Miradata: A large-scale video dataset with long durations and structured captions. *Advances in Neural Information Processing Systems*, 37:48955–48970, 2024.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. Musiq: Multi-scale image quality transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5128–5137. IEEE, 2021.

Youngrae Kim, Qixin Hu, C-C Jay Kuo, and Peter Beerel. Memrope: Training-free infinite video generation via evolving memory tokens. In *European Conference on Computer Vision*, pp. 244–261. Springer, 2026.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Haodong Li, Shaoteng Liu, Zhe Lin, and Manmohan Chandraker. Rolling sink: Bridging limited-horizon training and open-ended testing in autoregressive video diffusion. *arXiv preprint arXiv:2602.07775*, 2026a.

Jia Li, Xiaomeng Fu, Xurui Peng, Weifeng Chen, Youwei Zheng, Tianyu Zhao, Jiexi Wang, Fangmin Chen, Xing Wang, and Hayden Kwok-Hay So. Train short, inference long: Training-free horizon extension for autoregressive video generation. *arXiv preprint arXiv:2602.14027*, 2026b.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. In *International Conference on Learning Representations*, volume 2026, pp. 91177–91196, 2026.

Yunhong Lu, Yanhong Zeng, Haobo Li, Hao Ouyang, Qiuyu Wang, Ka Leong Cheng, Jiapeng Zhu, Hengyuan Cao, Zhipeng Zhang, Xing Zhu, et al. Reward forcing: Efficient streaming video generation with rewarded distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34385–34397, 2026.

Guoqing Ma, Haoyang Huang, Kun Yan, Liangyu Chen, Nan Duan, Shengming Yin, Changyi Wan, Ranchen Ming, Xiaoniu Song, Xing Chen, et al. Step-video-t2v technical report: The practice, challenges, and future of video foundation model. *arXiv preprint arXiv:2502.10248*, 2025.

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.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. *arXiv preprint arXiv:2402.17762*, 2024.

Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

Wan Team, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2(3):4, 2025.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

Jiahao Tian, Yiwei Wang, Gang Yu, and Chi Zhang. Head forcing: Long autoregressive video generation via head heterogeneity. *arXiv preprint arXiv:2605.14487*, 2026.

Ruiqi Wu, Xuanhua He, Meng Cheng, Tianyu Yang, Yong Zhang, Zhuoliang Kang, Xunliang Cai, Xiaoming Wei, Chunle Guo, Chongyi Li, et al. Infinite-world: Scaling interactive world models to 1000-frame horizons via pose-free hierarchical memory. *arXiv preprint arXiv:2602.02393*, 2026.

Weijia Wu, Zeyu Zhu, and Mike Zheng Shou. Automated movie generation via multi-agent cot planning. *arXiv preprint arXiv:2503.07314*, 2025.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

Zhexiao Xiong, Yizhi Song, Hao Kang, Qing Yan, Liming Jiang, Jenson Yang, Zhoujie Fu, Stathi Fotiadis, Angtian Wang, Zichuan Liu, et al. Actworld: From explorable to interactive world model via action-aware memory. *arXiv preprint arXiv:2606.17730*, 2026.

Zhenran Xu, Longyue Wang, Jifang Wang, Zhouyi Li, Senbao Shi, Xue Yang, Yiyu Wang, Baotian Hu, Jun Yu, and Min Zhang. Filmagent: A multi-agent framework for end-to-end film automation in virtual 3d spaces. *arXiv preprint arXiv:2501.12909*, 2025.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, et al. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025a.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025b.

Hidir Yesiltepe, Tuna Meral, Adil Kaan Akan, Kaan Oktay, and Pinar Yanardag. Infinity-rope: Action-controllable infinite video generation emerges from autoregressive self-rollout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40256–40265, 2026.

Jung Yi, Wooseok Jang, Paul Hyunbin Cho, Jisu Nam, Heeji Yoon, and Seungryong Kim. Deep forcing: Training-free long video generation with deep sink and participative compression. *arXiv preprint arXiv:2512.05081*, 2025.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6613–6623. IEEE, 2024.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974. IEEE, 2025.

Zangwei Zheng, Xiangyu Peng, Yuxuan Lou, Chenhui Shen, Tom Young, Xinying Guo, Binluo Wang, Hang Xu, Hongxin Liu, Mingyan Jiang, et al. Open-sora 2.0: Training a commercial-level video generation model in $200k$. *arXiv preprint arXiv:2503.09642*, 2025.

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.

A Details of the Deviation Analysis

This section provides the details of Section 4. The first paragraph describes the protocol under which the deviation is measured for Figure 1 (a), and the following two paragraphs define how the training-horizon statistics are estimated and the deviation measure. The remaining two paragraphs define the attention-weighted deviation visualized in Figure 2 and the two quantities compared in Figure 1 (b).

Measurement Protocol. The deviation, defined below, is measured on the tokens generated up to 480 latent frames for 36 prompts of MiraBench (Ju et al., 2024), disjoint from those used for estimating the training-horizon statistics. Throughout the measurement, the block-relative index assignment of MemRoPE (Kim et al., 2026) is applied so that every temporal index stays within the trained range, isolating the deviation of the token from that of the positions. In Figure 1 (a), the deviation is averaged over all heads and all tokens in each latent frame. Since the scale of the deviation differs across prompts, the per-frame deviation of each prompt is normalized by its mean over the first cache window of W frames. The normalized deviations are then averaged across prompts, with the shaded region indicating one standard deviation across prompts.

Statistics within the Training Horizon. Let $\mathbf{k}_i^{(l)}$ denote the i-th key from $\mathbf{K}^f$ at head $l \in \{1, \dots, L\}$ across all layers, with values $\mathbf{v}_i^{(l)}$ from $\mathbf{V}^f$ defined analogously and treated identically below. For each head l, the mean $\boldsymbol{\mu}^{(l)}$ and covariance $\boldsymbol{\Sigma}^{(l)}$ are computed over the keys that the base model produces within its training horizon of N_{train} latent frames ($N_{\text{train}} = 21$ for Self-Forcing (Huang et al., 2026)):

$$\boldsymbol{\mu}^{(l)} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{k}_i^{(l)}, \qquad \boldsymbol{\Sigma}^{(l)} = \frac{1}{n}\sum_{i=1}^{n} (\mathbf{k}_i^{(l)} - \boldsymbol{\mu}^{(l)})(\mathbf{k}_i^{(l)} - \boldsymbol{\mu}^{(l)})^\top, \tag{7}$$

where n is the number of collected keys. The keys are collected from the clean forward pass that writes each frame into the KV cache, before RoPE is applied, so that the statistics characterize exactly the representations that the model conditions on within the training horizon while remaining free of positional information. These statistics are estimated from 64 randomly sampled prompts of MiraBench, each generated up to the training horizon of the base model.

Deviation Measure. With the statistics fixed, the deviation of a key $\mathbf{k}_i^{(l)}$ is measured by the Mahalanobis distance to the training-horizon statistics of head l:

$$d_i^{(l)} = \sqrt{(\mathbf{k}_i^{(l)} - \boldsymbol{\mu}^{(l)})^\top (\boldsymbol{\Sigma}^{(l)})^{-1} (\mathbf{k}_i^{(l)} - \boldsymbol{\mu}^{(l)})}. \tag{8}$$

Attention-Weighted Deviation. Figure 2 visualizes, for each query at position c of the current frame, the deviation of the cached tokens weighted by the attention weights of the query:

$$D_{\text{cache}}(c) = \frac{1}{L}\sum_{l=1}^{L}\sum_{i} a_{c,i}^{(l)}\, d_i^{(l)}, \tag{9}$$

where i ranges over all tokens in the KV cache, $a_{c,i}^{(l)}$ is the attention weight from the query at c to the cached token i at head l, and $d_i^{(l)}$ is the deviation of the cached token i from Equation 8. The attention weights are read at the final denoising step of the current block, at which the output of the block is determined, and are averaged over the heads of all layers. The resulting map is reshaped to the spatial layout of the frame, so that a darker region indicates a position whose query concentrates its attention on more deviated cached tokens. The maps are computed separately with $d_i^{(l)}$ measured on the keys and on the values, shown in the middle and bottom rows of Figure 2, respectively.

Attended and Newly Generated Deviation. The two quantities compared in Figure 1 (b) are defined as

$$d_{\text{cache}}(c) = \frac{D_{\text{cache}}(c)}{\frac{1}{L}\sum_{l=1}^{L}\sum_{i} a_{c,i}^{(l)}}, \qquad d_{\text{new}}(c) = \frac{1}{L}\sum_{l=1}^{L} d_c^{(l)}, \tag{10}$$

where $d_{\text{cache}}(c)$ normalizes the attention-weighted deviation $D_{\text{cache}}(c)$ by the attention mass that the query at c places on the cache, and $d_c^{(l)}$ is the deviation of the key or value newly generated at position c for the cache. The normalization removes the effect of simply attending more to the cache, so that $d_{\text{cache}}(c)$ lies on the same scale as a deviation and reflects only how deviated the attended tokens are. The two quantities are measured at different points of the rollout. Since $d_{\text{new}}(c)$ concerns the token that enters the cache, it is measured on the additional forward pass at the clean noise level, which produces exactly the keys and values that the subsequent blocks read. In contrast, $d_{\text{cache}}(c)$ uses the attention weights at the final denoising step, as in Equation 9. Note that $d_i^{(l)}$ and $d_c^{(l)}$ are the same measure of Equation 8, applied to a cached token and to a newly generated token, respectively, and the deviation of a token is fixed once it is produced, as the raw tokens are cached unchanged. Each point in Figure 1 (b) corresponds to one position c, and the reported correlation is the Pearson coefficient between d_{cache} and d_{new} over all positions:

$$r = \frac{\sum_c \left(d_{\text{cache}}(c) - \bar{d}_{\text{cache}}\right)\left(d_{\text{new}}(c) - \bar{d}_{\text{new}}\right)}{\sqrt{\sum_c \left(d_{\text{cache}}(c) - \bar{d}_{\text{cache}}\right)^2}\sqrt{\sum_c \left(d_{\text{new}}(c) - \bar{d}_{\text{new}}\right)^2}}, \tag{11}$$

where $\bar{d}_{\text{cache}}$ and $\bar{d}_{\text{new}}$ are the means over all positions.

B Analysis of Token Deviation on LongLive

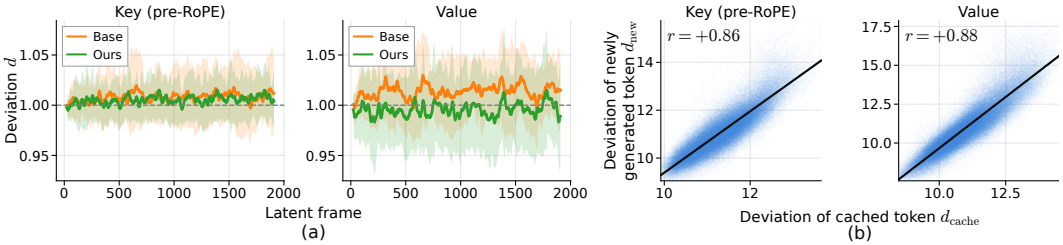

Figure 6: (a) Per-frame deviation of the cached value tokens from the tokens within the training horizon on LongLive, averaged over the heads from the 18th layer onward and normalized by the first cache window, for LongLive (orange) and Trace Forcing (green). Shaded regions indicate one standard deviation across prompts. (b) Deviation d_{new} of the token generated at each query position against the deviation d_{cache} of the cached tokens attended by the query.

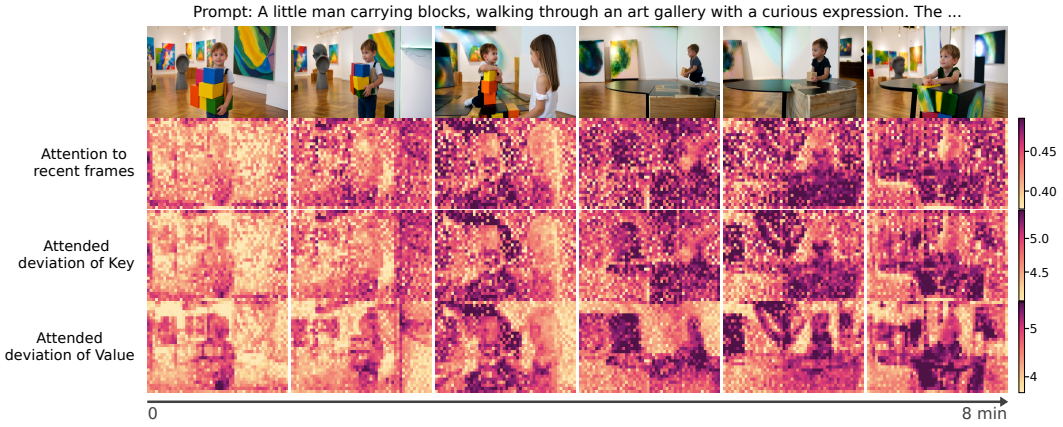

Figure 7: For each frame generated by LongLive (top), the maps visualize the attention mass that each query places on the recent slot of the cache (second row) and the deviation $D_{\text{cache}}(c)$ of the cached tokens weighted by the attention weights of each query, measured on the keys (third row) and values (bottom), where darker regions indicate queries that attend more to deviated tokens.

Section 4 conducts the analysis on Self-Forcing, whose training horizon spans 5 seconds. This section repeats the analysis on LongLive (Yang et al., 2025a), which is trained with a substantially

longer horizon of 60 seconds, under the same protocol as Appendix A with the training horizon set to $N_{\text{train}} = 241$ latent frames.

On LongLive, the deviation is observed to arise from the 18th layer onward, so the deviation is averaged over the heads of these layers only. Figure 6 (a) shows that the value tokens of the base LongLive deviate markedly (orange line), and that Trace Forcing (green line) mitigates this deviation. Unlike Self-Forcing, whose deviation grows as generation proceeds, the deviation of LongLive is low only in the very first frames. The remaining observations are consistent with those on Self-Forcing. In Figure 7, the artifacts emerge in the regions whose queries attend strongly to the deviated tokens (third and bottom rows), whereas the attention mass placed on the recent slot of the cache alone (second row) does not distinguish the artifact regions, indicating that the deviation of the attended tokens matters rather than the amount of attention to the cache. In Figure 6 (b), the deviation of the attended cached tokens is correlated with that of the newly generated token, following the same trend as Figure 1 (b).

C Design Choices of Adaptive Updating

This section provides the details of the two terms in the adaptive update rate of Section 5.2.

Content-Adaptive Gate. In practice, the gate is computed with $\bar{\delta}_c$ clamped from below by ϵ, as in z_c^f:

$$\gamma_{j,c} = \frac{\max(\bar{\delta}_c, \epsilon)^m}{\max(\bar{\delta}_c, \epsilon)^m + (\delta_{0,j})^m}, \tag{12}$$

so that the gate remains well defined for the most recent anchor, whose threshold $\delta_{0,j} = 0$ keeps the gate fully open regardless of $\bar{\delta}_c$. The threshold $\delta_{0,j}$, at which $\gamma_{j,c}$ reaches one half, decreases from the oldest to the most recent anchor. The oldest anchor thus opens its gate only for strongly evolving content and serves as a long-horizon reference, whereas the most recent anchor, whose context most directly conditions the subsequent frames, remains responsive to most of the content.

Deviation Rejection. The normalized dissimilarity z_c^f indicates whether the entering token lies within the range that the anchor tolerates ($z_c^f \lesssim 1$) or far beyond it, where ϵ prevents division by a vanishing $\bar{\delta}_c$. The exponential term, whose width is set by z_0, decays with z_c^f, so that tokens within the tolerated range are admitted at nearly the full rate while tokens far beyond it, which signal a deviated token, are suppressed.

D Implementation Details

Update Rule. The global scale α_0 is set to 0.2 for Self-Forcing and 0.02 for LongLive. The per-anchor thresholds $\delta_{0,j}$, indexed by j from the oldest to the most recent anchor, are set to $[0.20, 0.15, 0.10, 0.05, 0]$ for Self-Forcing and $[0.08, 0.06, 0.04, 0.02, 0]$ for LongLive, so that the most recent anchor keeps its gate fully open. The remaining parameters are shared across the base models: the sharpness $m = 8$, the rejection width $z_0 = 4$, and $\epsilon = 10^{-3}$. During the warm-up, the anchors are updated with $\alpha_{\text{warm}} = 0.1$ over the training horizon of $N_{\text{train}} = 21$ and 241 latent frames for Self-Forcing and LongLive, respectively, and the frames updated into each anchor within this horizon constitute $\mathcal{F}_{\text{train}}$.

Matching and Updating. Correspondence matching selects, for each token, the top-1 nearest neighbor in the immediately preceding frame by the cosine similarity of the rotary-embedded keys, independently per layer and head. The cosine distance δ_c^f and the resulting update rate are computed from the keys, and the same rate is applied to both the key and the value anchors.

E Algorithm of Trace Forcing

The overall inference procedure of Trace Forcing is summarized in Algorithm 1, written in block-level notation for brevity, where a block refers to a chunk of frames. The denoising of each block follows the standard rollout with the three-component cache, where block-relative RoPE is applied online to

the unrotated keys at every step. After each block is denoised, the anchor update proceeds frame by frame: the entering frame is matched to its predecessor, and the frame evicted from the recent slot traverses the anchors from the most recent to the oldest, being updated into each with the adaptive rate before it is discarded. Once g exceeds N_{train}, $\bar{\delta}$ of each anchor is computed from the recorded δ_c^g and used thereafter.

Algorithm 1 Trace Forcing Inference

Require: Diffusion Transformer G_θ, warm-up rate α_{warm}, and the hyperparameters of Eq. 6
Notation: $\mathcal{Q}$: queue of raw frames evicted from the recent slot; g: global temporal index of a frame in $\mathcal{Q}$; $\mathbf{k}_c^g \in \mathbf{K}^g$, $\mathbf{v}_c^g \in \mathbf{V}^g$: tokens at position c; $\tilde{\mathbf{K}}$: rotary-embedded keys, where the cache stores unrotated keys and applies RoPE online (Kim et al., 2026)

Generate the first block and cache it as the sink
for each block b **do**
  Denoise block b over $t = T, \dots, 1$ with $\mathbf{K} = [\mathbf{K}^{\text{sink}} \parallel \mathbf{K}^{\text{anc}} \parallel \mathbf{K}^{\text{rct}}]$ ▷ denoising pass
  $\mathbf{K}^b, \mathbf{V}^b \leftarrow G_\theta(\mathbf{x}^b, \sigma_0, \mathbf{K}, \mathbf{V})$ ▷ clean pass
  for each frame f of block b **do**
    $\pi^f \leftarrow$ nearest neighbor of $\tilde{\mathbf{K}}^f$ in $\tilde{\mathbf{K}}^{f-1}$ ▷ Eq. 3
    Append $(\mathbf{K}^f, \mathbf{V}^f, \pi^f)$ to the recent slot
    Move the oldest frame of the recent slot to $\mathcal{Q}$ if the slot exceeds R
    for $j = A, \dots, 1$ **do**
      $(\mathbf{K}^g, \mathbf{V}^g, \pi^g) \leftarrow$ frame of $\mathcal{Q}$ reaching anchor slot j
      Relocate $\mathbf{k}_j^{\text{anc}}, \mathbf{v}_j^{\text{anc}}, \bar{\delta}_j$ along π^g
      if $g \le N_{\text{train}}$ **then**
        $\alpha_c^g \leftarrow \alpha_{\text{warm}}$; record δ_c^g for $\bar{\delta}_{j,c}$ ▷ warm-up
      else
        $\alpha_c^g \leftarrow$ Eq. 6 with $\bar{\delta}_{j,c}$ ▷ for each position c
      end if
      $\mathbf{k}_{j,c}^{\text{anc}} \leftarrow (1-\alpha_c^g)\,\mathbf{k}_{j,c}^{\text{anc}} + \alpha_c^g\,\mathbf{k}_c^g,\ \mathbf{v}_{j,c}^{\text{anc}} \leftarrow (1-\alpha_c^g)\,\mathbf{v}_{j,c}^{\text{anc}} + \alpha_c^g\,\mathbf{v}_c^g$ ▷ Eq. 2
    end for
    Discard the frame of $\mathcal{Q}$ past the oldest anchor slot
  end for
end for

F Evaluation Details

Quality Drift. Since VBench-Long scores are averaged over the entire video, a long rollout may retain a high average score while its quality degrades over time. The quality drift reported in Section 6 therefore complements the averages with two measures of the temporal change in quality. Both measures are computed on the imaging quality and the aesthetic quality of VBench-Long, reported on the same 0–100 scale, with lower values indicating higher stability. In the following, $\{V_1, \dots, V_P\}$ denotes the P generated videos of a setting, and $s(\cdot)$ denotes the score of a video segment.

The first measure, following Rolling Forcing (Liu et al., 2026) and Deep Forcing (Yi et al., 2025), quantifies the cumulative degradation over the horizon by the difference in quality between the beginning and the end of a video:

$$\Delta = \frac{1}{P}\sum_{i=1}^{P} \big|\, s(V_i^{\text{first}}) - s(V_i^{\text{last}}) \,\big|, \tag{13}$$

where V_i^{first} and V_i^{last} denote the first and the last 80 frames (5 seconds at 16 FPS) of V_i. As VBench-Long scores two-second clips whose boundaries do not align with a five-second window, the two segments are extracted directly from the generated video and re-encoded under identical settings for every method (`libx264`, CRF 10, 16 FPS). The segments are then scored with the scoring functions of VBench, namely MUSIQ (Ke et al., 2021) for imaging quality and the LAION aesthetic predictor (Schuhmann et al., 2022) on CLIP ViT-L/14 features for aesthetic quality, where the 0–10 output of the aesthetic predictor is rescaled to 0–100 following VBench.

The second measure, following Reward Forcing (Lu et al., 2026), quantifies the fluctuation of quality within the horizon. VBench-Long partitions each video into two-second clips and scores each clip

independently, yielding M clip scores $s_{i1}, \ldots, s_{iM}$ for V_i, and the sample standard deviation within each video is averaged over videos:

$$\sigma = \frac{1}{P}\sum_{i=1}^{P}\sqrt{\frac{1}{M-1}\sum_{m=1}^{M}\left(s_{im}-\bar{s}_i\right)^2}, \qquad \bar{s}_i = \frac{1}{M}\sum_{m=1}^{M} s_{im}. \tag{14}$$

This measure responds to both a monotone decline and clip-to-clip variation, and thus reflects the overall instability of quality across the rollout. The original formulation applies this measure to imaging quality on 60-second videos, and the same formulation is extended here to aesthetic quality.

The two measures are complementary. Equation 14 accounts for the entire trajectory but does not distinguish a steady decline from oscillation around a constant level, whereas Equation 13 isolates the change between the two ends but is estimated from only two segments per video and thus exhibits higher variance. A method that is stable over the horizon is expected to attain low values on both. All methods are evaluated on the identical set of prompts with the same seeds, so that every comparison is paired per prompt.

G Additional Qualitative Results

Additional qualitative results are presented in Figure 8, Figure 9, and Figure 10, covering Self-Forcing at 240 and 480 seconds and LongLive at 480 seconds. Across all base models and horizons, Trace Forcing preserves the subject identity and visual fidelity throughout the generation, whereas the baselines progressively degrade as generation proceeds far beyond the training horizon.

H VLM Evaluation

Table 4: VLM-based visual stability evaluation on 240-second video generation. Each method is applied on top of Self-Forcing and compared within the same base model. The best results are highlighted in **bold**, and the second-best results are underlined.

Method	Stability ↑
Results on Self-Forcing (240 seconds)	
Self-Forcing	2.44
Deep Forcing	3.20
∞-RoPE	<u>3.68</u>
MemRoPE	3.62
Ours	**3.70**

We further assess long-horizon visual stability using the vision-language model Gemini 3.1-Pro (Google DeepMind, 2026). Following the protocol of Self-Forcing++ (Cui et al., 2026a) and Deep Forcing (Yi et al., 2025), we prompt the VLM to rate each 240-second generated video on a 0–5 exposure stability scale, using the same 128 prompts as in Table 1, and all methods are evaluated under an identical prompt and decoding setting. As shown in Table 4, Trace Forcing achieves the highest stability score. The evaluation prompt is as follows:

VLM Evaluation Prompt

You are tasked with rating the exposure stability of a video. Assign a score according to the following scale:
0: Catastrophic Exposure. Nearly the entire frame is either blown out (pure white) or crushed (pure black), rendering the scene unreadable.
1: Severe Exposure Failure. Large portions are dominated by over- or under-exposure, substantially impairing visibility.
2: Noticeable Exposure Problems. Persistent clipping in highlights or shadows; significant areas lose detail.
3: Moderate Exposure Issues. Over-exposed highlights or under-exposed shadows occur but are limited in extent or duration.
4: Minor Exposure Flaws. Small regions are occasionally too bright or too dark, but do not meaningfully disrupt visibility.
5: Well-Exposed. Balanced lighting across the frame with no distracting over-exposure or darkening.

I Analysis of the Adaptive Update Rate

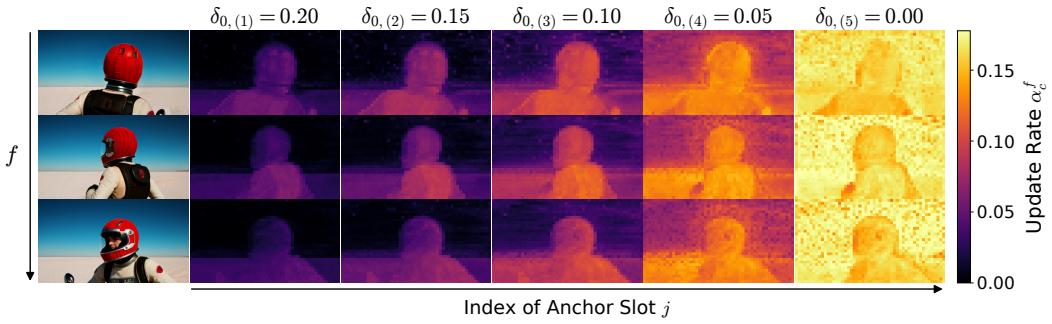

Figure 11: Visualization of the adaptive update rate across the anchors. Update rate α_c^f of each anchor when generating the frame on the left, in increasing order of the anchor index j. Since $\delta_{0,j}$ increases with the distance from the recent frames, the anchors farther from the recent are updated less and preserve the past.

Figure 12 presents the distribution of the update rate α_c^f for each anchor slot with its threshold $\delta_{0,j}$, where a larger $\delta_{0,j}$ shifts the distribution toward lower rates. Accordingly, the anchors with smaller indices are assigned larger $\delta_{0,j}$ to suppress their updates and preserve the past representations, whereas those with larger indices are assigned smaller $\delta_{0,j}$ to admit more updates and reflect the recent context. Figure 11 visualizes the resulting rates on the generated frames, in which the update rate decreases monotonically as j decreases. Within each anchor slot, the relatively static background receives lower rates, whereas the dynamically evolving foreground receives higher rates. These observations confirm that adaptive updating operates as designed, allocating the update rate according to both the anchor index and the evolution of the content.

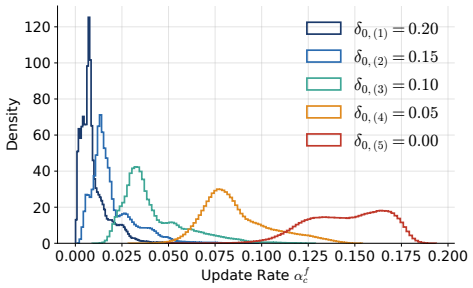

Figure 12: Distribution of the update rate α_c^f for each anchor slot j with its threshold $\delta_{0,j}$.

J LIMITATIONS

Trace Forcing is designed to constrain the cached tokens to the training distribution, and adaptive updating varies the update rate across positions according to how the content at each position evolves within the training horizon. The degree of evolution is estimated where the model generates reliably, and a comparable estimate is not available for content that emerges after the warm-up, since the generation beyond the training horizon no longer provides a reliable reference. Such content is currently updated at the rate assigned to the anchor it enters, and estimating the evolution of newly emerging or disappearing content is left for future work. In addition, since each anchor is updated with the generated tokens, a small portion of the deviation may still accumulate in the anchors in some cases. The degree of suppression can be controlled by the global scale α_0 and the thresholds $\delta_{0,j}$, which allows Trace Forcing to be adjusted to each case, although finding the appropriate values requires manual effort. Determining them automatically is left for future work.

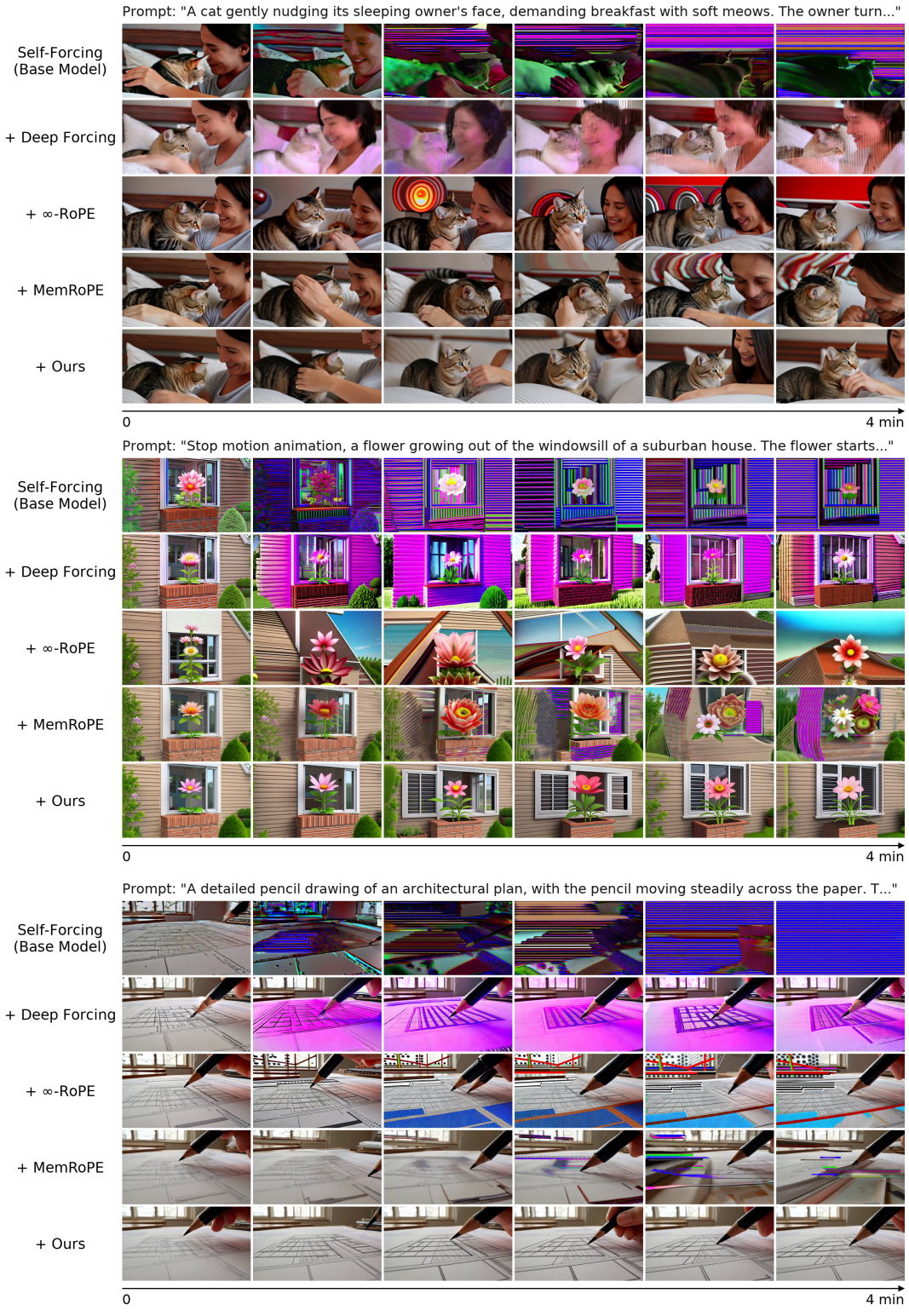

Figure 8: Additional qualitative results on Self-Forcing at 240 seconds.

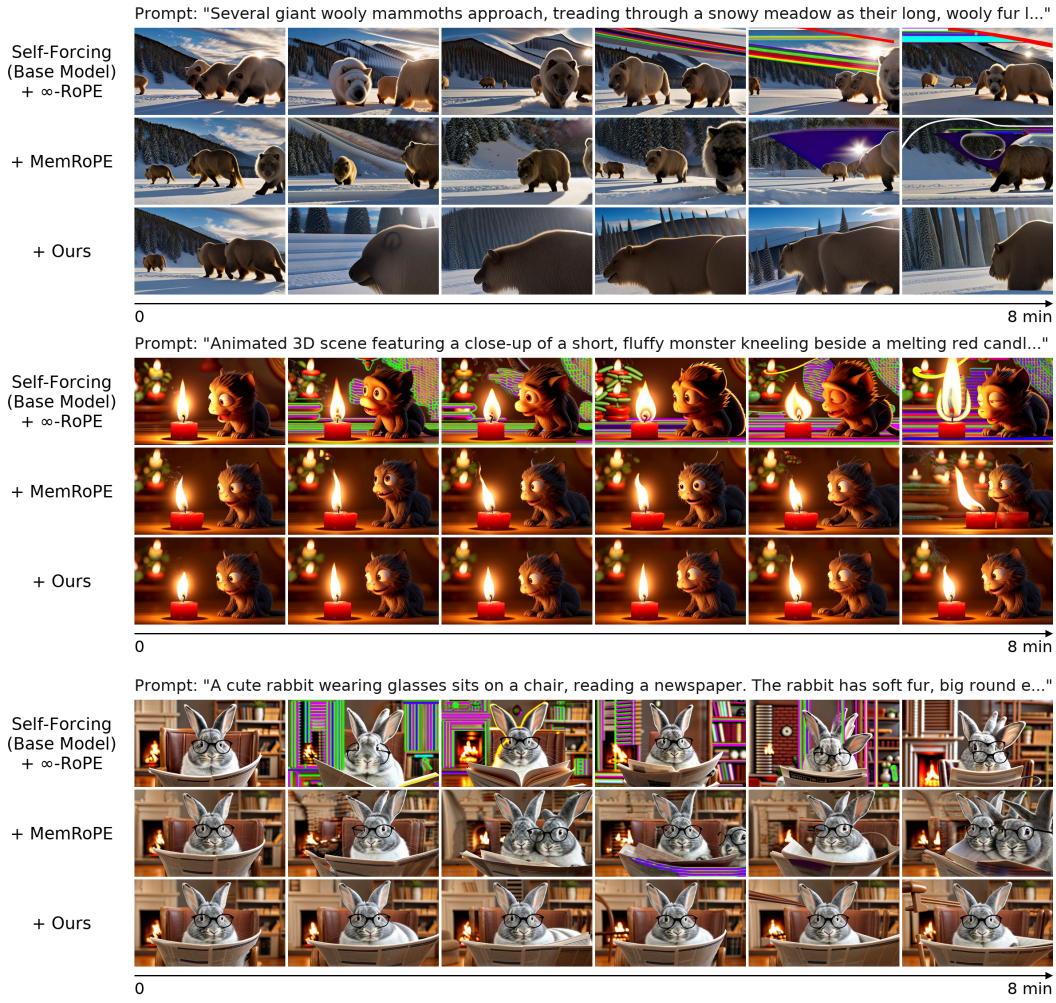

Figure 9: Additional qualitative results on Self-Forcing at 480 seconds.

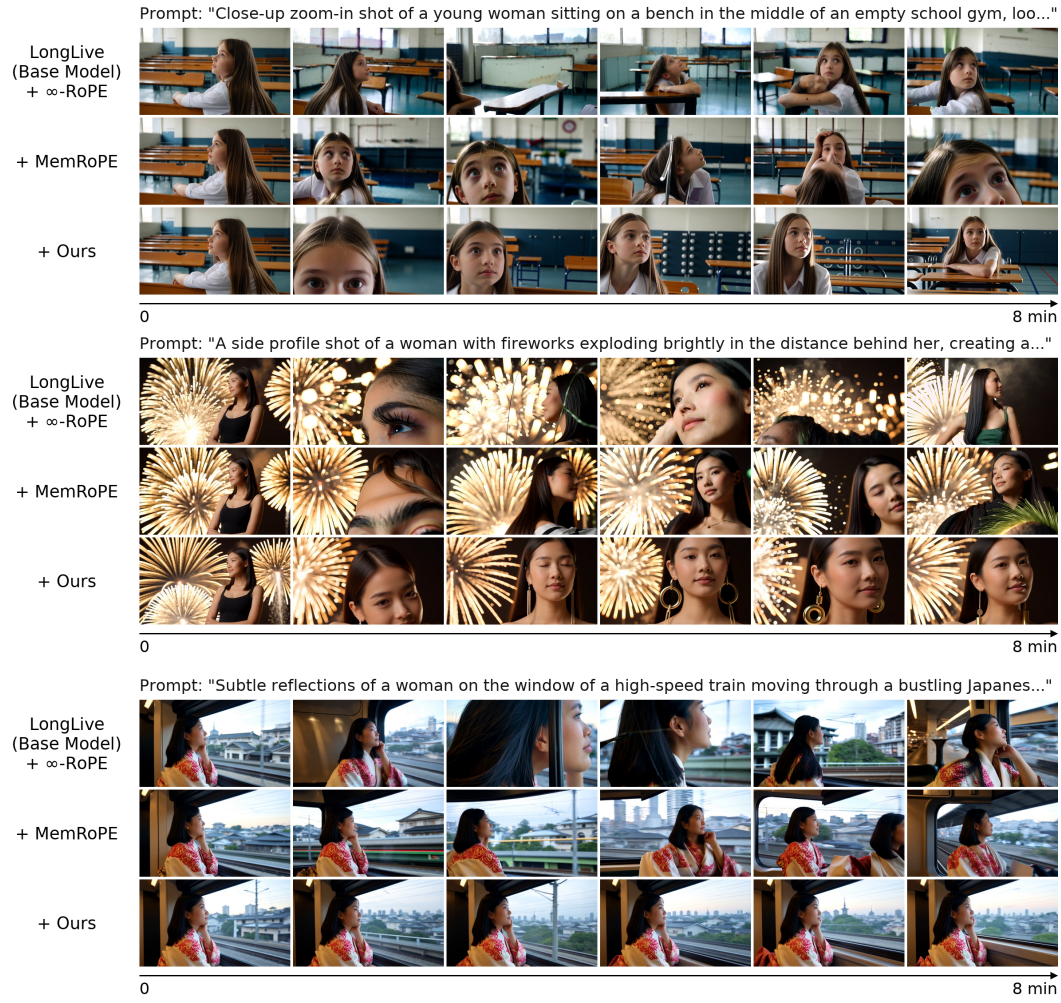

Figure 10: Additional qualitative results on LongLive at 480 seconds.