FUTURE FORCING: FUTURE-AWARE TRAINING-FREE KV CACHE POLICY FOR AUTOREGRESSIVE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) video generation has emerged as a promising paradigm for long-horizon video synthesis, where each frame is generated conditioned on previously generated tokens. To accelerate inference, the KV cache is used to avoid redundant recomputation across generation steps. Nevertheless, its growth increases memory use, while compression errors can accumulate over generation, limiting the scalability of AR models to even longer sequences. Existing KV cache compression methods mitigate this issue by selectively retaining only video tokens deemed important. However, most existing methods assess token importance using short-horizon signals derived from the current or historical generation context, making these methods prone to overlooking tokens that appear unimportant at early steps but later become critical for future frames. In this work, we identify an important property of trained AR video models: ***although RoPE-modulated queries evolve across autoregressive steps, pre-RoPE query distributions exhibit relative temporal stability under fixed prompts***. This observation motivates an online future-query proxy, whose utility we validate through future-importance ranking and proxy-construction ablations. Building on this insight, we propose **Future Forcing**[1], a training-free future-aware KV cache policy for AR video generation. Specifically, Future Forcing first constructs a future query proxy from historical statistics, then scores KV cache tokens by their importance under this proxy, and finally merges redundant token pairs using their future-query profiles. Extensive experiments show that Future Forcing improves long-horizon consistency under limited KV caches, achieving up to 1.49 improvement in subject consistency on VBench-Long for 60s generation over existing AR video KV cache policies.

1 INTRODUCTION

Bidirectional Diffusion Transformer (DiT)-based video generation models Wan et al. (2025); Kong et al. (2024); Wu et al. (2025); Chen et al. (2025); Hong et al. (2023) employ bidirectional attention to jointly denoise all frames, enabling coherent full-sequence modeling and high-quality synthesis. Yet this paradigm is inherently offline, ill-suited for streaming or continual video extension Huang et al. (2025); Yin et al. (2025). By contrast, autoregressive (AR) video models Zhu et al. (2026); Liu et al. (2025a) causally condition each frame on the prior tokens, enabling incremental generation.

To make incremental decoding efficient, AR video models reuse historical key and value representations through KV caching, avoiding repeated computation over the full history at each step. Since the cache grows with generation length, practical memory constraints require operating under a bounded KV cache budget. Under this constraint, cache selection becomes critical: evicting tokens needed by future frames distorts attention over historical context, and these errors propagate through subsequent autoregressive steps, accumulating into visual degradation, temporal inconsistency, and motion drift.

However, most existing KV cache compression methods for AR video generation estimate token importance from current or recent context-derived signals Yi et al. (2025); Zhu et al. (2026); Liu et al. (2025a); Lu et al. (2025); Huang et al. (2025). These signals reflect present relevance but cannot

[1] https://anonymous.4open.science/w/FutureForcing

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.

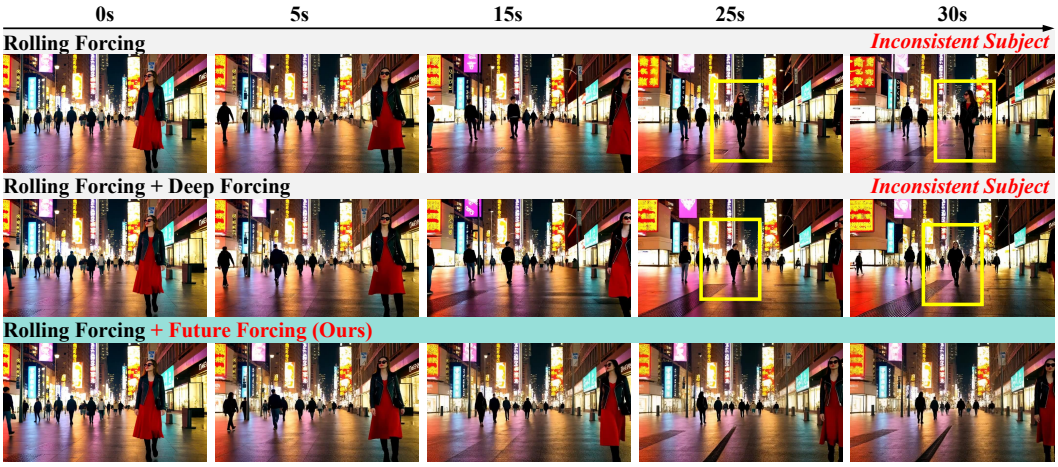


Figure 1: Qualitative comparison of 30s long video generation across different KV cache strategies. Competing methods exhibit temporal and subject inconsistency as generation progresses, while our Future Forcing maintains coherent subject consistency throughout the full AR generation process.

anticipate future attention demands, as future query representations are unavailable at KV cache eviction time. ***This induces a short horizon bias under a limited cache budget***: *tokens with low current importance may be discarded before becoming critical for scene structure, object identity, or motion continuity, causing attention errors that accumulate into severe long horizon degradation.*

In this work, we identify an intriguing property of trained AR video models by examining query representations before and after RoPE modulation. In transformer attention, rotary positional embedding (RoPE) Su et al. (2021) is applied to queries before attention weights are computed, producing position encoded queries that determine how each frame attends to cached history. Our key observation is that **these RoPE-modulated attention queries vary substantially across autoregressive steps, whereas their underlying pre-RoPE query distributions remain stable under a fixed prompt**. This reveals a separation between stable semantic query structure and step dependent positional modulation, providing a basis for anticipating future attention behavior without additional training.

Building on this insight, we propose Future Forcing, a future-aware training-free KV cache policy for AR video generation that preserves generation quality and consistency under a fixed cache budget to alleviate short-horizon limitations. Specifically, Future Forcing first constructs query proxies based on the stable pre-RoPE query distribution and uses them to score cached tokens by their predicted future importance, rather than relying only on their current or historical importance. Then, our Future Forcing further merges evicted tokens into retained entries with similar future-query-induced attention profiles to reduce compression-induced loss. In summary, our core contributions are as follows:

- We investigate query representation dynamics in well-trained AR video generation models and uncover an intriguing phenomenon: RoPE-modulated queries vary substantially across autoregressive steps, whereas the underlying pre-RoPE query distribution remains stable under a fixed prompt.
- We investigate the short-horizon limitation of existing KV cache compression methods, which rely on local context-derived signals and may discard tokens that are critical for future decoding. To address this, we propose Future Forcing, a future-aware training-free KV cache policy that uses future query proxies to guide token eviction and merging under the fixed KV cache budget.
- Extensive experiments on representative autoregressive video generation models show that Future Forcing outperforms existing autoregressive video KV cache policies, achieving up to 1.49 subject-consistency improvement while maintaining generation quality under limited KV cache budgets.

2 Related Work

Autoregressive video generation. Bidirectional diffusion transformers such as Wan and Hunyuan-Video jointly model video frames Wan et al. (2025); Kong et al. (2024). Autoregressive models

instead support incremental synthesis. CausVid Yin et al. (2025), Self-Forcing Huang et al. (2025), Causal-Forcing Zhu et al. (2026), Rolling-Forcing Liu et al. (2025a), and Reward-Forcing Lu et al. (2025) improve causal video diffusion through distillation and training strategies. LongLive Yang et al. (2025) extends this setting to interactive generation. Our policy requires no backbone retraining.

KV cache policies for video. Rolling caches and frame sinks bound the retained history Huang et al. (2025); Yang et al. (2025), while Deep Forcing Yi et al. (2025) selects tokens using historical attention. PackCache Li et al. (2026), Dummy Forcing Guo et al. (2026), FlowCache Ma et al. (2026), and temporal cache compression Samuel et al. (2026) explore compact historical representations; Quant VideoGen Xi et al. (2026) uses low-bit caches. Relax Forcing, Anchor Forcing, and PackForcing organize long-term memory Zhao et al. (2026); Yang et al. (2026); Mao et al. (2026b). PaFu-KV Chen et al. (2026) learns salience from past- and future-informed supervision. In contrast, our future proxies guide eviction and merging without a trained salience predictor.

Query-based KV compression. Unlike Gaussian query modeling Devoto et al. (2025) or offline-calibrated query/key centers Mao et al. (2026a) for LLMs, Future Forcing estimates prompt-conditioned pre-RoPE queries online from recent video frames. It applies future positional rotations and uses the resulting query profiles for both cache eviction and merging. This design exploits temporal stability within a video while adapting its statistics to the current prompt. Appendix I.4 details these distinctions and transferred-policy comparisons.

3 Motivation and Analysis

3.1 Preliminaries

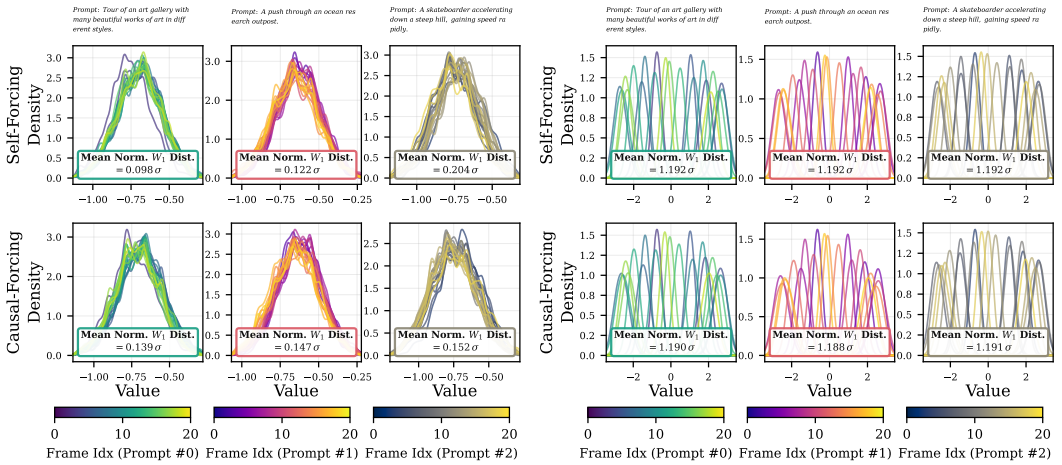
Autoregressive Video Generation and KV Caching. We consider autoregressive (AR) video generation in the latent space Huang et al. (2025). Given a prompt c, the model generates a video token sequence $\mathbf{x}_{1:T} = [\mathbf{x}_1, \ldots, \mathbf{x}_T]$ causally, where $\mathbf{x}_t \in \mathbb{R}^{N\times d}$ denotes the N latent tokens of frame t. The generation process is factorized as $p(\mathbf{x}_{1:T} \mid c) = \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{<t}, c)$, so each frame is predicted conditioned only on previously generated context. For a token i with hidden representation $\mathbf{h}_i \in \mathbb{R}^d$, its query, key, and value are defined as $\mathbf{q}_i = \mathbf{W}_Q\mathbf{h}_i$, $\mathbf{k}_i = \mathbf{W}_K\mathbf{h}_i$, and $\mathbf{v}_i = \mathbf{W}_V\mathbf{h}_i$. For a newly decoded token, causal self-attention is computed over previously generated tokens, with attention weights $\alpha_{ij} = \frac{\exp(\tilde{\mathbf{q}}_i^\top \tilde{\mathbf{k}}_j/\sqrt{d})}{\sum_{\ell\le i}\exp(\tilde{\mathbf{q}}_i^\top \tilde{\mathbf{k}}_\ell/\sqrt{d})}$ and output $\mathrm{Attn}(\mathbf{q}_i, \mathbf{K}_{\le i}, \mathbf{V}_{\le i}) = \sum_{j\le i}\alpha_{ij}\mathbf{v}_j$, where $\tilde{\mathbf{q}}_i$ and $\tilde{\mathbf{k}}_j$ denote the position-modulated query and key. During autoregressive decoding, past keys and values are typically cached and reused to avoid recomputing the entire history during generation.

RoPE in AR Video Generation. To encode spatiotemporal positions, Rotary Position Embedding (RoPE) is commonly adopted in transformer attention layers for video generation Su et al. (2021); Wei et al. (2025); Liu et al. (2025b). For a token at 3D position $\mathbf{p}_i = (t_i, y_i, x_i)$, RoPE applies a position-dependent transformation to the query and key, in its standard matrix notation, as $\tilde{\mathbf{q}}_i = R(\mathbf{p}_i)\mathbf{q}_i$ and $\tilde{\mathbf{k}}_i = R(\mathbf{p}_i)\mathbf{k}_i$, where $R(\mathbf{p}_i)$ denotes the corresponding RoPE transformation determined by the token's spatiotemporal coordinates. Under this notation, the attention score can be explicitly written as $\tilde{\mathbf{q}}_i^\top\tilde{\mathbf{k}}_j = \mathbf{q}_i^\top R(\mathbf{p}_i)^\top R(\mathbf{p}_j)\mathbf{k}_j$, directly showing that RoPE injects positional information through the interaction between the position-dependent transformations applied to queries and keys. In this work, we refer to $\mathbf{q}_i$ as the *pre-RoPE query* and to $\tilde{\mathbf{q}}_i$ as the *RoPE-modulated query*. The former precedes the current layer's positional rotation; the latter enters attention.

3.2 Pre-RoPE Query Stability Across AR Video Generation

We compare pre- and post-RoPE query distributions under fixed prompts. Figure 2 illustrates three examples; Table 1 extends the analysis to 128 randomly sampled MovieGen prompts across four backbones. Pre-RoPE queries exhibit substantially lower mean-normalized Wasserstein distance, supporting relative temporal stability beyond the visualized examples. Additional dimensions and dynamic scenes appear in Appendices G and H.

This consistent empirical phenomenon motivates estimating future attention from historical pre-RoPE query statistics. We further evaluate the resulting proxy through future-importance ranking and proxy-construction comparisons in Section 5.3.



(a) Pre-RoPE queries distribution (**Stable**). (b) Post-RoPE queries distribution (**Varying**).

Figure 2: Query distribution across latent frames for different autoregressive video generation models. (Quantitative results based on mean normalized Wasserstein distance are detailed in Appendix G). While the distributions of *RoPE-modulated queries vary noticeably* across autoregressive generation stages, the distributions of the corresponding *pre-RoPE queries remain considerably more stable*.

Table 1: Query stability over 128 MovieGen prompts. Lower normalized Wasserstein distance indicates less cross-frame distributional drift.

Query	Self-Forcing	Causal-Forcing	Rolling-Forcing	Reward-Forcing
Pre-RoPE	**0.1202**	**0.1313**	**0.1219**	**0.1268**
Post-RoPE	1.1929	1.1922	1.1934	1.1506

4 METHOD

In this section, we introduce our **Future Forcing**, a future-aware training-free KV cache policy for autoregressive video generation that reduces KV-cache memory overhead while mitigating the error accumulation during long-horizon decoding. As illustrated in Figure 3, Future Forcing comprises two complementary modules: (1) a *Future-aware KV Cache Eviction Module* that estimates future query proxies from historical pre-RoPE statistics to retain only the most relevant KV-cache entries; and (2) a *Future-aware KV Cache Merging Module* that consolidates evicted entries into the retained cache in the future query profiling space, reducing information loss under a fixed KV budget.

4.1 FUTURE-AWARE KV CACHE EVICTION MODULE

As shown in Figure 2, pre-RoPE query distributions remain relatively stable under fixed prompts, while RoPE-modulated queries vary across autoregressive steps. Intuitively, a fixed prompt maintains a common semantic focus, while RoPE maps queries to different spatiotemporal positions at each generation step. Video frames under a fixed prompt often share scene content and object identity Xi et al. (2026); Samuel et al. (2026); Ma et al. (2026). We therefore use recent query statistics to represent this shared content and explicitly apply the positional rotation for each future frame. The following analysis characterizes the contribution of positional rotation.

At a fixed layer and head, let $\mathbf{Q}_t = \mathbf{X}_t \mathbf{W}_Q^\top$ and $\tilde{\mathbf{Q}}_t$ denote pre- and post-RoPE queries at matching spatial indices. Define $\epsilon = \|\mathbf{X}_s - \mathbf{X}_t\|_F$ and $\gamma(\Delta p) = \max_{k \in \mathcal{T}} 2|\sin(\Delta p \theta_k / 2)|$ over temporal RoPE blocks $\mathcal{T}$. The relative query drifts are

$$\rho_{\text{pre}}^{t \to s} = \frac{\|\mathbf{Q}_s - \mathbf{Q}_t\|_F}{\|\mathbf{Q}_t\|_F}, \qquad \rho_{\text{post}}^{t \to s} = \frac{\|\tilde{\mathbf{Q}}_s - \tilde{\mathbf{Q}}_t\|_F}{\|\tilde{\mathbf{Q}}_t\|_F}.$$

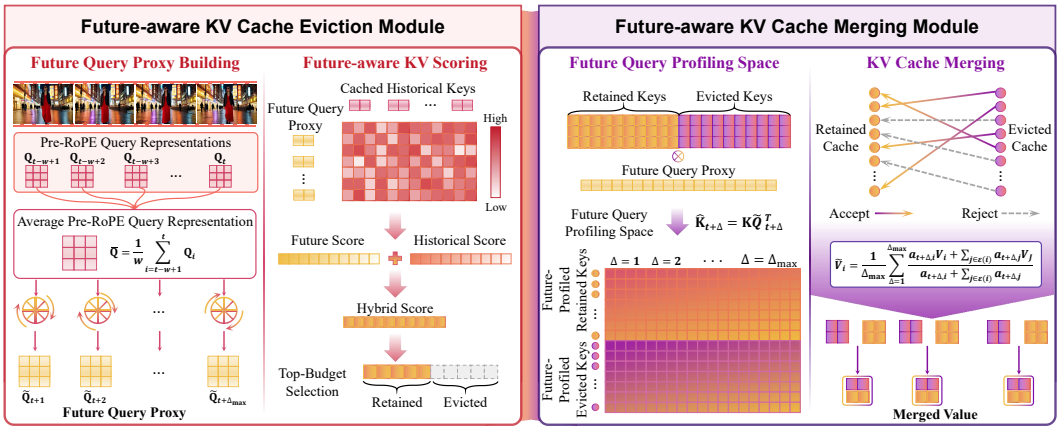


Figure 3: Overview of Future Forcing, which constructs future query proxies from stable pre-RoPE queries, combines future- and history-aware scores to retain KV pairs, and merges evicted entries into retained ones in future-query profiling space, reducing information loss under a limited cache budget.

Theoretical Analysis 1 (Query drift bound). *For $\|\mathbf{Q}_t\|_F > 0$, the relative query drifts satisfy*

$$\rho_{\text{pre}}^{t\to s} \le \frac{\epsilon\|\mathbf{W}_Q\|_{\text{op}}}{\|\mathbf{Q}_t\|_F}, \qquad \rho_{\text{post}}^{t\to s} \le \gamma(\Delta p) + \frac{\epsilon\|\mathbf{W}_Q\|_{\text{op}}}{\|\mathbf{Q}_t\|_F}, \tag{1}$$

The bounds separate hidden-state variation from the explicit temporal rotation term, with the proof in Appendix A. This decomposition motivates constructing the proxy before positional rotation and then applying the desired future positions.

Building on the empirical analysis, we estimate future attention without additional backbone forward passes. Let $\{\mathbf{Q}_{t-w+1}, \dots, \mathbf{Q}_t\}$ denote the pre-RoPE query matrices from the most recent w latent frames, with rows indexed by spatial tokens. Their mean $\bar{\mathbf{Q}}_t = \frac{1}{w}\sum_{s=t-w+1}^{t} \mathbf{Q}_s$ provides a representative query matrix over the recent temporal window. For a future frame $t+\Delta$, where $\Delta \in \{1, \dots, \Delta_{\max}\}$, we construct $\tilde{\mathbf{Q}}_{t+\Delta} = \mathcal{R}_{t+\Delta}(\bar{\mathbf{Q}}_t)$, where $\mathcal{R}_{t+\Delta}$ applies RoPE at the future position. This proxy estimates the positionally encoded queries of the future frame.

We then estimate the attention logit of frame $t+\Delta$ against each cached key $\hat{\mathbf{K}}_j$ as $\hat{s}_{t+\Delta,j} = \langle \tilde{\mathbf{Q}}_{t+\Delta}, \hat{\mathbf{K}}_j \rangle / \sqrt{d_h}$, and obtain its softmax-normalized weight $\hat{a}_{t+\Delta,j}$ as a per-token future importance estimate. The final cache entry score combines predicted future relevance with average historical attention across frames as follows:

$$\text{score}_j = \lambda \underbrace{\frac{1}{\Delta_{\max}} \sum_{\Delta=1}^{\Delta_{\max}} \hat{a}_{t+\Delta,j}}_{\text{future}} + (1-\lambda) \underbrace{\frac{1}{t}\sum_{s=1}^{t} a_{s,j}}_{\text{history}}, \tag{2}$$

where λ balances predicted future relevance and historical attention. The two terms combine anticipated demand with attention observed during generation. Cache entries are ranked by this score, and those below the budget threshold are evicted.

Custom Attention Kernel Design. Since existing attention kernels such as FlashAttention (Dao et al., 2022; Dao, 2024) do not expose per-token attention scores, we implement a lightweight two-pass Triton kernel to compute Eq. equation 2 without materializing the full $\mathbf{QK}^\top$ attention score matrix, reducing future-aware scoring computational overhead. Implementation details are in Appendix B.

4.2 Future-aware KV Cache Merging Module

Under a fixed KV cache budget, naively discarding evicted entries introduces irreversible information loss. A straightforward remedy is to merge entries with high key similarity (Zhang et al., 2024), so that evicted information is absorbed into retained ones. However, as the attention output $\mathbf{O} = \sum_j a_j \mathbf{V}_j$ depends on the attention weights $a_j = \text{softmax}(\mathbf{QK}_j^\top / \sqrt{d_h})$, *the contribution of each KV entry is*

inherently query-dependent: two close keys may receive drastically different attention weights under different queries, making key-similarity alone an unreliable merging criterion. Instead, we propose to perform merging in the future-query profiling space: rather than measuring proximity by Euclidean distance, we group entries by the similarity of their predicted attention logit profiles under future queries, aiming to reduce compression-induced distortion.

Specifically, let $\mathcal{S}$ and $\mathcal{E}$ denote the retained and evicted index sets. For each evicted entry $j \in \mathcal{E}$, we determine its merge target by measuring similarity in the future query profiling space. Let $\mathbf{Q}_{\text{proxy}}$ stack the future proxy queries and define key profiles $\hat{\mathbf{K}} = \mathbf{K}\mathbf{Q}_{\text{proxy}}^{\top}$. High cosine similarity between $\hat{\mathbf{K}}_i$ and $\hat{\mathbf{K}}_j$ identifies entries $i \in \mathcal{S}$ and $j \in \mathcal{E}$ with similar directional responses to the future queries. We route each evicted entry to its nearest retained neighbor $i^\star(j) = \arg\max_{i \in \mathcal{S}} \cos(\hat{\mathbf{K}}_i, \hat{\mathbf{K}}_j)$, and admit the merge only when $\cos(\hat{\mathbf{K}}_{i^\star(j)}, \hat{\mathbf{K}}_j) \geq \tau$ for fidelity threshold $\tau \in [0, 1]$, discarding j otherwise. The merged key is set to $\widetilde{\mathbf{K}}_i = \mathbf{K}_i$ to preserve RoPE consistency, while the merged value is obtained by weighting each entry by its future attention mass across the look-ahead horizon:

$$\widetilde{\mathbf{V}}_i \;=\; \frac{1}{\Delta_{\max}} \sum_{\Delta=1}^{\Delta_{\max}} \frac{\hat{a}_{t+\Delta,i}\, \mathbf{V}_i + \sum_{j \in \mathcal{E}(i)} \hat{a}_{t+\Delta,j}\, \mathbf{V}_j}{\tilde{a}_{t+\Delta,i}}, \tag{3}$$

where $\tilde{a}_{t+\Delta,i} = \hat{a}_{t+\Delta,i} + \sum_{j \in \mathcal{E}(i)} \hat{a}_{t+\Delta,j}$ is the re-normalized attention mass at slot i, and $\mathcal{E}(i) = \{j \in \mathcal{E} : i^\star(j) = i,\ \cos(\hat{\mathbf{K}}_i, \hat{\mathbf{K}}_j) \geq \tau\}$ is the set of admitted evicted entries routed to i.

Dynamic RoPE in Long Video Generation. Following Yi et al. (2025); Zhu et al. (2026), as absolute frame indices grow beyond the model's trained temporal range during long video generation, we apply a RoPE correction after each eviction step that remaps surviving cached frame positions to be contiguous immediately before the current generation position. This correction is applied in-place to the time-axis of cached key tensors, leaving spatial RoPE channels untouched, limiting temporal distances during extrapolation. Future offsets are defined in this remapped coordinate system. Each step scores the aligned cache, evicts and merges entries, remaps surviving keys, and then appends the generated block. We isolate correction from our policy in Section 5.4.

5 Experiment

5.1 Experimental Settings

Implementation Details. We evaluate our Future Forcing across a diverse set of representative autoregressive video diffusion generation models, including Self Forcing Huang et al. (2025), Causal Forcing Zhu et al. (2026), Rolling Forcing Liu et al. (2025a), Reward Forcing Lu et al. (2025), LongLive Yang et al. (2025), and Deep Forcing Yi et al. (2025). For the main hyperparameters, we set the future-history balance $\lambda = 0.5$, look-ahead horizon $\Delta_{\max} = 6$, merge threshold $\tau = 0.95$, and query averaging window $w = 12$ latent frames. For the used KV cache budget, we retain 18 latent frames for Self Forcing and Causal Forcing, 15 latent frames for Rolling Forcing, and 9 latent frames for Reward Forcing in our implementation.

Evaluation Metrics. We use MovieGen prompts Polyak et al. (2024) for two settings: (1) 30s/60s generation beyond the training horizon, evaluated with VBench-Long; and (2) 5s generation within the training horizon, evaluated with PSNR/SSIM Wang et al. (2004), LPIPS Zhang et al. (2018), and VBench Huang et al. (2024). For short videos, we vary only the cache strategy within each backbone and use full-cache outputs as reconstruction references. This measures fidelity to uncompressed generation, not agreement with ground-truth video. Table 2 uses the configured backbone-specific budgets; Table 3 compares equal retained-cache ratios. Additional diagnostic studies use 20 sampled prompts and are reported separately from the main benchmark.

5.2 Generation Quality Study

Long Video Generation (Beyond Training Horizon). We evaluate Future Forcing on 30s and 60s long video generation with VBench-Long metrics Huang et al. (2024) using MovieGen prompts Polyak et al. (2024), with results in Table 2. Future Forcing improves subject consistency on Self-, Rolling-, and Causal-Forcing and yields competitive results on Reward-Forcing.

Table 2: Quantitative comparison of Future Forcing and baselines on **30-second and 60-second long video generation** with VBenchLong Huang et al. (2024) on the full MovieGen prompt set Polyak et al. (2024). ↑: higher is better. Best results within each group are bold.

Config	Backbone + KV Cache Policy	Subject Consist.↑	Background Consist.↑	Motion Smooth.↑	Dynamic Degree↑	Aesthetic Quality↑	Imaging Quality↑	Overall Consist.↑
30s	Self Forcing+Native	96.01	95.20	98.16	45.27	55.43	68.92	23.09
	Self Forcing+Deep Forcing	96.39	95.75	98.36	59.34	57.67	69.05	**23.91**
	Self Forcing+Future Forcing	**97.22**	**96.18**	**98.48**	**63.08**	**57.93**	**70.22**	23.60
	Reward Forcing+Native	97.02	**96.06**	98.34	**70.28**	57.04	70.29	**23.94**
	Reward Forcing+Deep Forcing	96.81	96.03	98.35	57.62	56.80	68.11	23.75
	Reward Forcing+Future Forcing	**97.11**	95.96	**98.57**	69.77	**57.68**	**70.35**	23.61
	Rolling Forcing+Native	97.91	96.56	98.59	34.20	57.15	70.89	23.25
	Rolling Forcing+Deep Forcing	97.87	96.54	98.78	**35.97**	57.68	70.29	23.84
	Rolling Forcing+Future Forcing	**98.15**	**96.72**	**98.81**	32.42	**58.04**	**71.48**	**24.01**
	Causal Forcing+Native	94.74	94.70	96.80	97.14	58.71	**69.07**	23.85
	Causal Forcing+Deep Forcing	94.64	94.65	96.75	**97.23**	58.49	68.85	23.87
	Causal Forcing+Future Forcing	**96.11**	**95.45**	**97.47**	88.16	**59.31**	**69.07**	**24.51**
60s	Self Forcing+Native	96.45	95.12	98.46	36.15	54.23	67.28	20.30
	Self Forcing+Deep Forcing	96.42	**96.09**	98.45	51.01	**57.47**	68.86	23.80
	Self Forcing+Future Forcing	**97.01**	96.06	**98.54**	**62.63**	57.21	**69.94**	**23.87**
	Reward Forcing+Native	96.50	95.47	98.30	**72.35**	57.74	69.84	**23.94**
	Reward Forcing+Deep Forcing	**97.08**	95.14	98.22	60.45	56.68	66.62	23.37
	Reward Forcing+Future Forcing	96.55	**95.78**	**98.59**	69.12	**58.71**	**69.99**	23.73
	Rolling Forcing+Native	97.73	96.48	98.76	34.20	56.65	71.09	23.25
	Rolling Forcing+Deep Forcing	97.69	96.41	98.75	**34.97**	56.87	71.03	23.19
	Rolling Forcing+Future Forcing	**98.05**	**96.60**	**98.78**	31.43	**57.71**	**71.38**	**23.50**
	Causal Forcing+Native	94.62	94.61	96.71	**97.40**	58.30	68.94	23.50
	Causal Forcing+Deep Forcing	94.50	94.57	96.67	97.30	58.08	68.62	23.57
	Causal Forcing+Future Forcing	**96.11**	**95.44**	**97.48**	86.79	**59.09**	**69.31**	**24.20**

Table 3: 5s reconstruction fidelity relative to full-cache generation on the full MovieGen prompt set. Ratios denote retained KV cache relative to full cache. Full VBench results appear in Table 8.

Backbone	Policy	28.57% retained			42.86% retained		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Self-Forcing	Reward Forcing	18.438	0.665	0.174	21.344	0.758	0.116
	Deep Forcing	18.562	0.660	0.176	20.943	0.744	0.122
	Dummy Forcing	17.939	0.633	0.200	19.333	0.683	0.161
	Future Forcing	**19.258**	**0.690**	**0.158**	**22.164**	**0.780**	**0.106**
Causal-Forcing	Reward Forcing	16.290	0.664	0.199	18.958	0.756	0.129
	Deep Forcing	16.047	0.651	0.206	18.417	0.729	0.139
	Dummy Forcing	15.191	0.612	0.249	16.571	0.659	0.199
	Future Forcing	**16.704**	**0.681**	**0.189**	**19.127**	**0.758**	**0.126**
LongLive	Reward Forcing	20.001	0.734	0.126	23.313	0.824	0.075
	Deep Forcing	15.932	0.595	0.270	21.480	0.775	0.101
	Dummy Forcing	18.572	0.689	0.167	20.298	0.739	0.127
	Future Forcing	**20.830**	**0.761**	**0.109**	**24.035**	**0.842**	**0.065**

Comparison on a common backbone. Table 4 compares video cache policies with Expected Attention and TriAttention, adapted from LLM KV-cache compression to AR video generation on Self-Forcing. Future Forcing obtains the highest SC, BC, and MS on this 20-prompt subset. Adaptation details and Causal-Forcing results appear in Appendix I.4.

Table 4: KV cache policies on Self-Forcing (30s, 20 prompts). Best scores are bold.

KV Cache Policy	Subject Consistency↑	Background Consistency↑	Motion Smoothness↑
Native	96.47	95.66	98.42
Deep Forcing	96.94	96.03	98.41
Reward Forcing policy	97.11	95.98	98.38
Causal Forcing policy	96.75	96.48	98.47
Rolling Forcing policy	96.92	96.17	98.35
Dummy Forcing	96.89	95.75	98.16
Expected Attention (adapted from LLM)	97.09	96.48	98.27
TriAttention (adapted from LLM)	96.74	96.12	98.33
Future Forcing	**97.67**	**96.51**	**98.49**

Table 5: Future-importance ranking on 20 prompts (Spearman ↑).

Score	SF	CF
History only	0.3304	0.4315
Future only	0.6545	0.6592
Future + History	**0.6732**	**0.6657**

Table 6: Proxy construction (30s, 20 prompts). SC / BC are reported; Ours uses averaging.

Proxy	Self-Forcing		Causal-Forcing	
	SC	BC	SC	BC
Recent	97.08	96.00	95.01	94.96
EMA	97.46	**96.64**	96.05	95.20
Sampling	96.85	96.39	95.71	95.02
Ours	**97.67**	96.51	**96.11**	**95.50**

Discussion on Dynamic Degree Decrease. For the Causal-Forcing backbone, Future Forcing exhibits a lower dynamic degree in Table 2. This does **not necessarily indicate weaker motion**: in the visualized baseline videos, higher scores arise from **abrupt, undesired scene changes** rather than meaningful object or camera motion. By avoiding these unstable transitions, Future Forcing preserves **reasonable dynamics with better temporal consistency**. Visual comparisons are provided in Appendix F and on our anonymous website.

Short Video Generation (Within Training Horizon). Table 3 evaluates Future Forcing on 5s videos within the training horizon. With a reduced KV cache budget, Future Forcing most closely matches the full-cache reference among compared methods, indicating lower reconstruction loss among the evaluated policies. These results suggest a practical, scalable approach to reducing KV cache memory for future video models trained on longer horizons.

5.3 Future-Query Proxy Validation

Future-importance ranking. Table 5 compares cache-importance rankings with those induced by realized future queries on 20 prompts. Future-only scoring improves correlation from 0.3304 to 0.6545 on Self-Forcing and from 0.4315 to 0.6592 on Causal-Forcing. Adding history further improves ranking agreement on both backbones.

Query approximation. On 20 prompts, our proxy achieves relative L_2 errors of 0.4512/0.4264 and cosine similarities of 0.8796/0.8863 against realized future queries on Self-/Causal-Forcing. Post-RoPE averaging yields errors of 0.8375/0.8259 and cosine similarities of 0.4147/0.4229. A separate 10-prompt study compares six layers spanning layers 0–29. Pre-RoPE averaging gives higher cosine similarity than post-RoPE averaging at every probed layer on both backbones; the complete layer-wise measurements for both backbones appear in Appendix I.

Proxy construction. Table 6 compares the latest query, EMA (0.9), Gaussian sampling, and window averaging. Averaging gives the highest subject consistency on both backbones, while EMA gives higher background consistency on Self-Forcing. Appendix I reports query errors, layer-wise similarity, historical-attention controls, and window sensitivity.

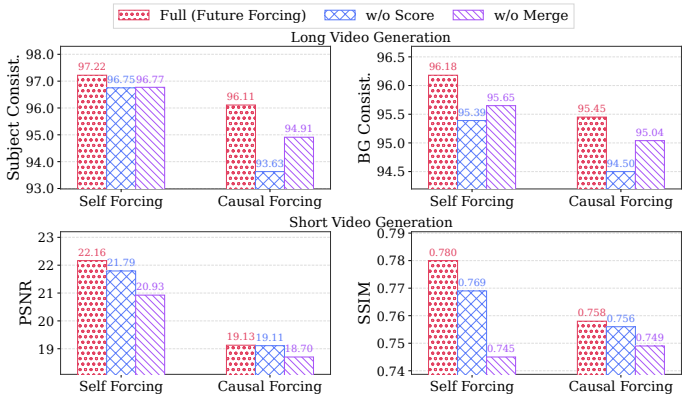


Figure 4: Ablation study of Future Forcing on long- and short-video generation, isolating future scoring and merging modules.

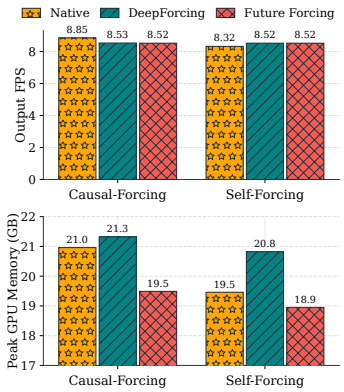


Figure 5: Efficiency and memory study of our Future Forcing.

Table 7: Dynamic RoPE ablation on 20 prompts (30s). SC and BC measure subject and background consistency; IQ measures imaging quality.

Variant	Self-Forcing			Causal-Forcing		
	SC	BC	IQ	SC	BC	IQ
Native	96.47	95.66	71.65	95.13	94.46	71.22
+ RoPE correction	96.81	96.04	72.38	95.34	94.82	71.30
+ Future Forcing	**97.67**	**96.51**	**72.80**	**96.11**	**95.50**	**71.43**

5.4 Component Ablations

To assess each component, we conduct ablations with two variants: (1) *w/o Score*, replacing Future-aware KV Cache Eviction with Sink + FIFO, and (2) *w/o Merge*, removing Future-aware KV Cache Merging and discarding evicted entries. We evaluate them on 30s long-video and 5s short-video generation, respectively. As shown in Figure 4, both modules improve performance: future-aware eviction contributes more to long-video generation, while merging is critical for short-video fidelity.

Dynamic RoPE correction. Table 7 reports native caching, native caching with correction, and Future Forcing with correction. Correction alone improves Self-Forcing SC from 96.47 to 96.81; adding our policy increases it to 97.67. Causal-Forcing shows the same trend. The comparison shows the additional consistency gain from our cache policy after applying RoPE correction.

Appendix I reports merge acceptance rates, threshold sensitivity, and complex-scene ablations.

5.5 Efficiency and Peak GPU Memory Usage Study

Figure 5 reports output FPS and peak GPU memory on NVIDIA A100 80GB GPUs. Across Self-Forcing and Causal-Forcing, Future Forcing maintains comparable inference efficiency to the baselines and reduces peak GPU memory by up to 1.8GB on Causal-Forcing.

6 Conclusion

In this work, we presented Future Forcing, a future-aware training-free KV cache policy for AR video generation. We identify a key property of trained AR video models: although RoPE-modulated queries vary substantially across AR steps, their pre-RoPE query distributions remain relatively stable under a fixed prompt. Based on this insight, Future Forcing constructs future query proxies to guide cache eviction and merges evicted entries in a future-query profiling space to reduce information loss. Experiments across representative AR video models show improved long-horizon consistency and short-video reconstruction fidelity under limited KV cache budgets.

AI USE STATEMENT

AI tools were used only to polish the writing of this paper. All content, methods, code, and experiments were done by the authors, who take full responsibility for the paper.

ETHICS STATEMENT

Future Forcing is a training-free KV cache policy that reduces the memory cost of existing AR video models and does not add new generation capabilities. Like other video generation methods, the underlying models may be misused to create misleading content. We encourage responsible use.

REPRODUCIBILITY STATEMENT

The method, hyperparameters, KV cache budgets, and evaluation settings are described in the main paper, with further details in the appendix. Since Future Forcing is training-free, results can be reproduced with public model checkpoints. We will release our code.

REFERENCES

Hanmo Chen, Chenghao Xu, Xu Yang, Xuan Chen, and Cheng Deng. Past-and future-informed kv cache policy with salience estimation in autoregressive video diffusion. *arXiv preprint arXiv:2601.21896*, 2026.

Yi Chen, Sen Liang, Zixiang Zhou, Ziyao Huang, Yifeng Ma, Junshu Tang, Qin Lin, Yuan Zhou, and Qinglin Lu. Hunyuanvideo-avatar: High-fidelity audio-driven human animation for multiple characters. *arXiv preprint arXiv:2505.20156*, 2025.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *ICLR*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022.

Alessio Devoto, Maximilian Jeblick, and Simon Jégou. Expected attention: Kv cache compression by estimating attention from future queries distribution. *arXiv preprint arXiv:2510.00636*, 2025.

Hang Guo, Zhaoyang Jia, Jiahao Li, Bin Li, Yuanhao Cai, Jiangshan Wang, Yawei Li, and Yan Lu. Efficient autoregressive video diffusion with dummy head. *arXiv preprint arXiv:2601.20499*, 2026.

Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *ICLR*, 2023.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *NeurIPS*, 2025.

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. *CVPR*, 2024.

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.

Kunyang Li, Mubarak Shah, and Yuzhang Shang. Packcache: A training-free acceleration method for unified autoregressive video generation via compact kv-cache. *arXiv preprint arXiv:2601.04359*, 2026.

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*, 2025a.

Zikang Liu, Longteng Guo, Yepeng Tang, Junxian Cai, Kai Ma, Xi Chen, and Jing Liu. Vrope: Rotary position embedding for video large language models. *arXiv preprint arXiv:2502.11664*, 2025b.

Yunhong Lu et al. Reward forcing: Efficient streaming video generation with rewarded distribution matching distillation. *arXiv preprint arXiv:2512.04678*, 2025.

Yuexiao Ma, Xuzhe Zheng, Jing Xu, Xiwei Xu, Feng Ling, Xiawu Zheng, Huafeng Kuang, Huixia Li, Xing Wang, Xuefeng Xiao, et al. Flow caching for autoregressive video generation. *arXiv preprint arXiv:2602.10825*, 2026.

Weian Mao, Xi Lin, Wei Huang, Yuxin Xie, Tianfu Fu, Bohan Zhuang, Song Han, and Yukang Chen. Triattention: Efficient long reasoning with trigonometric kv compression. *arXiv preprint arXiv:2604.04921*, 2026a.

Xiaofeng Mao, Shaohao Rui, Kaining Ying, Bo Zheng, Chuanhao Li, Mingmin Chi, and Kaipeng Zhang. Packforcing: Short video training suffices for long video sampling and long context inference. *arXiv preprint arXiv:2603.25730*, 2026b.

Adam Polyak et al. Movie Gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Dvir Samuel, Issar Tzachor, Matan Levy, Micahel Green, Gal Chechik, and Rami Ben-Ari. Fast autoregressive video diffusion and world models with temporal cache compression and sparse attention. *arXiv preprint arXiv:2602.01801*, 2026.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2021.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Pan Zhang, Yuhang Cao, Jian Tong, Haodong Duan, Qipeng Guo, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. Videorope: What makes for good video rotary position embedding? *ICML*, 2025.

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025.

Haocheng Xi, Shuo Yang, Yilong Zhao, Muyang Li, Han Cai, Xingyang Li, Yujun Lin, Zhuoyang Zhang, Jintao Zhang, Xiuyu Li, Zhiying Xu, Jun Wu, Chenfeng Xu, Ion Stoica, Song Han, and Kurt Keutzer. Quant videogen: Auto-regressive long video generation via 2-bit kv-cache quantization. *arXiv preprint arXiv:2602.02958*, 2026.

Shuai Yang et al. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025.

Yang Yang, Tianyi Zhang, Wei Huang, Jinwei Chen, Boxi Wu, Xiaofei He, Deng Cai, Bo Li, and Peng-Tao Jiang. Anchor forcing: Anchor memory and tri-region rope for interactive streaming video diffusion. *arXiv preprint arXiv:2603.13405*, 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 et al. From slow bidirectional to fast autoregressive video diffusion models. In *CVPR*, 2025.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Yuxin Zhang, Yuxuan Du, Gen Luo, Yunshan Zhong, Zhenyu Zhang, Shiwei Liu, and Rongrong Ji. Cam: Cache merging for memory-efficient llms inference. In *ICML*, 2024.

Zengqun Zhao, Yanzuo Lu, Ziquan Liu, Jifei Song, Jiankang Deng, and Ioannis Patras. Relax forcing: Relaxed kv-memory for consistent long video generation. *arXiv preprint arXiv:2603.21366*, 2026.

Hongzhou Zhu et al. Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. *arXiv preprint arXiv:2602.02214*, 2026.

A Proof of the Query Drift Bound

Theoretical Analysis A.1 (Query drift bound)**.** *At a fixed layer and attention head, let $\mathbf{X}_t$ denote the hidden representations entering the query projection, $\mathbf{Q}_t = \mathbf{X}_t\mathbf{W}_Q^\top$, and $\tilde{\mathbf{Q}}_t$ their RoPE-rotated counterparts. Compare matching spatial indices and assume $\|\mathbf{Q}_t\|_F > 0$. Define $\epsilon = \|\mathbf{X}_s - \mathbf{X}_t\|_F$ and $\gamma(\Delta p) = \max_{k\in\mathcal{T}} 2|\sin(\Delta p\theta_k/2)|$, where $\mathcal{T}$ indexes temporal rotation blocks. Then*

$$\frac{\|\mathbf{Q}_s - \mathbf{Q}_t\|_F}{\|\mathbf{Q}_t\|_F} \le \frac{\epsilon\|\mathbf{W}_Q\|_{\mathrm{op}}}{\|\mathbf{Q}_t\|_F}, \qquad \frac{\|\tilde{\mathbf{Q}}_s - \tilde{\mathbf{Q}}_t\|_F}{\|\tilde{\mathbf{Q}}_t\|_F} \le \gamma(\Delta p) + \frac{\epsilon\|\mathbf{W}_Q\|_{\mathrm{op}}}{\|\mathbf{Q}_t\|_F}. \tag{A.1}$$

Proof. The first inequality follows from $\mathbf{Q}_s - \mathbf{Q}_t = (\mathbf{X}_s - \mathbf{X}_t)\mathbf{W}_Q^\top$ and the mixed Frobenius–operator norm inequality. For each spatial token, add and subtract $R_s\mathbf{q}_t$ to write

$$R_s\mathbf{q}_s - R_t\mathbf{q}_t = R_s(\mathbf{q}_s - \mathbf{q}_t) + (R_s - R_t)\mathbf{q}_t. \tag{A.2}$$

Orthogonality preserves norms. Matching spatial coordinates cancel the spatial rotation differences, and each temporal block satisfies $\|R_2(p_s\theta_k) - R_2(p_t\theta_k)\|_{\mathrm{op}} = 2|\sin(\Delta p\theta_k/2)|$. The operator norm of the block-diagonal difference is the maximum over temporal blocks. Applying the triangle inequality across tokens and dividing by $\|\tilde{\mathbf{Q}}_t\|_F = \|\mathbf{Q}_t\|_F$ gives the result. □

The two bounds share a hidden-state variation term; the post-RoPE bound additionally includes the explicit temporal rotation term $\gamma(\Delta p)$. Section 3.2 measures distributional stability, and Section 5.3 evaluates the resulting query proxy.

B Custom Kernel Design

Computing the per-token importance scores in Eq. equation 2 requires reducing the softmax attention weights $\mathrm{softmax}(\mathbf{Q}\mathbf{K}^\top/\sqrt{d_h})$ over the query dimension. Standard attention kernels, such as FlashAttention (Dao et al., 2022; Dao, 2024), do not expose these weights, since they are designed to return only the value aggregated attention output while avoiding materialization of the intermediate softmax matrix. A naive dense PyTorch implementation must therefore instantiate the full $\mathbf{Q}\mathbf{K}^\top$ probability matrix before reduction. To address this, we implement a custom two-pass Triton kernel that evaluates Eq. equation 2 without instantiating the full attention weight matrix, adapting the online softmax primitive from value aggregation to score reduction.

Algorithm. The kernel specializes the online softmax recurrence for the reduction required by Eq. equation 2. Instead of accumulating a weighted sum of values, it accumulates the softmax mass assigned to each key slot. This requires two passes over $\mathbf{K}$.

Pass 1 streams $\mathbf{K}$ in tiles of size B_K, while maintaining the per query running maximum m_q and partial normalization sum ℓ_q in SRAM. After all key tiles have been processed, the kernel writes the finalized log sum exp value

$$\mathrm{LSE}_{b,h,q} = m_q + \log \ell_q \tag{B.1}$$

to an $O(BHQ)$ buffer. This pass is necessary because normalized attention weights cannot be computed until the softmax denominator has been finalized.

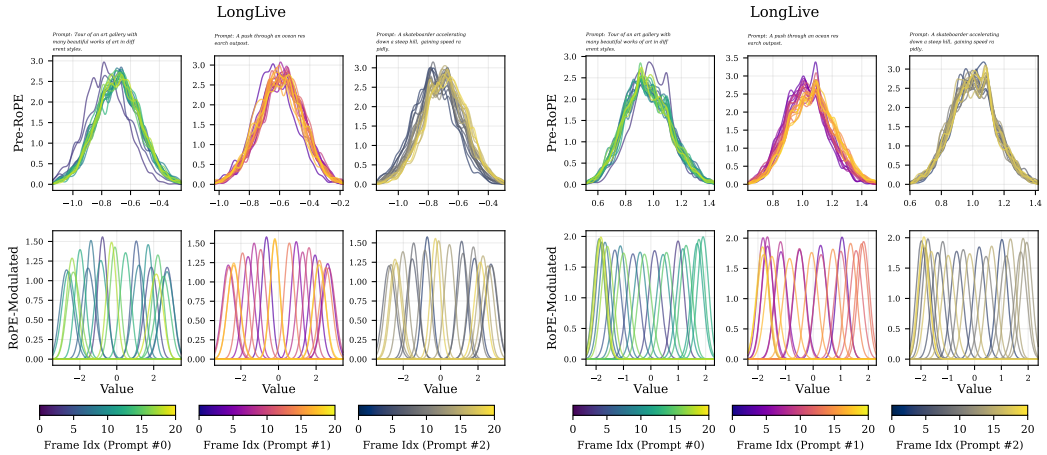
Pass 2 streams $\mathbf{K}$ again using the same $(BH, \lceil Q/B_Q\rceil)$ launch grid. For each tile, the kernel recomputes the raw dot product

$$s_{q,t} = \mathbf{Q}_q\mathbf{K}_t^\top/\sqrt{d_h} \tag{B.2}$$

on chip, applies the finalized normalizer,

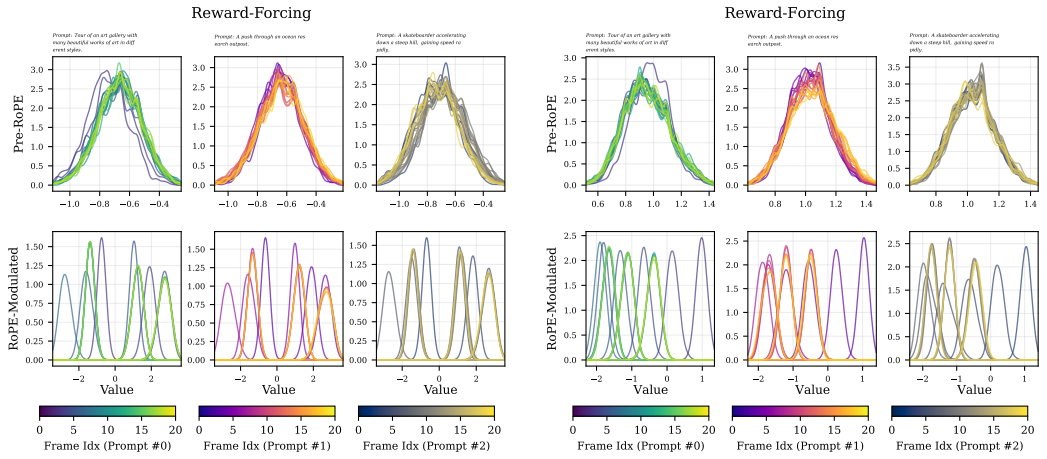
$$w_{q,t} = \exp(s_{q,t} - \mathrm{LSE}_{b,h,q}), \tag{B.3}$$

and immediately reduces over the query tile to accumulate $\sum_{q\in\mathrm{tile}} w_{q,t}$ for each key slot. Summing these partial contributions across query tiles yields $\mathrm{score}_{b,h,t}$ as defined in Eq. equation 2.



(a) Dimension = 0 (b) Dimension = 5

Figure 6: LongLive query distributions across two dimensions.



(a) Dimension = 0 (b) Dimension = 5

Figure 7: Reward-Forcing query distributions across two dimensions.

Implementation Details. Both passes use tile sizes $B_Q{=}B_K{=}128$ and share the same launch configuration. All tile level intermediates, including **Q** tiles, **K** tiles, running statistics m_q and ℓ_q, and normalized weights $w_{q,t}$, are kept in SRAM during each tile iteration. The head dimension d_h is padded to the next power of two to support valid `tl.dot` operations for arbitrary head widths. The resulting working set is

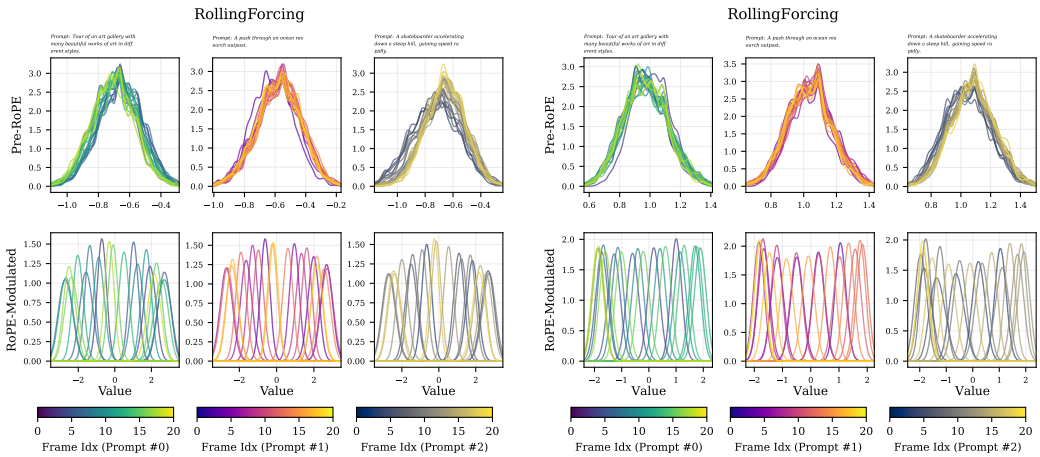
$$O\left(BHQ + BH\left\lceil \frac{Q}{B_Q} \right\rceil T\right), \tag{B.4}$$

which avoids the dense $O(BHQT)$ matrix and enables efficient per-token scoring.

C Additional Experiment Results

C.1 Additional Query Distribution Analysis Results.

We provide additional query distribution visualizations to further validate the pre-RoPE query stability discussed in Section 3.2. In particular, we extend the analysis beyond the main-text examples to three additional representative autoregressive video generation models: LongLive (Yang et al., 2025), Reward-Forcing (Lu et al., 2025), and Rolling-Forcing (Liu et al., 2025a). For each model, we visualize query distributions at two representative query dimensions, as shown in Figure 6, Figure 7,



(a) Dimension = 0 (b) Dimension = 5

Figure 8: Rolling-Forcing query distributions across two dimensions.

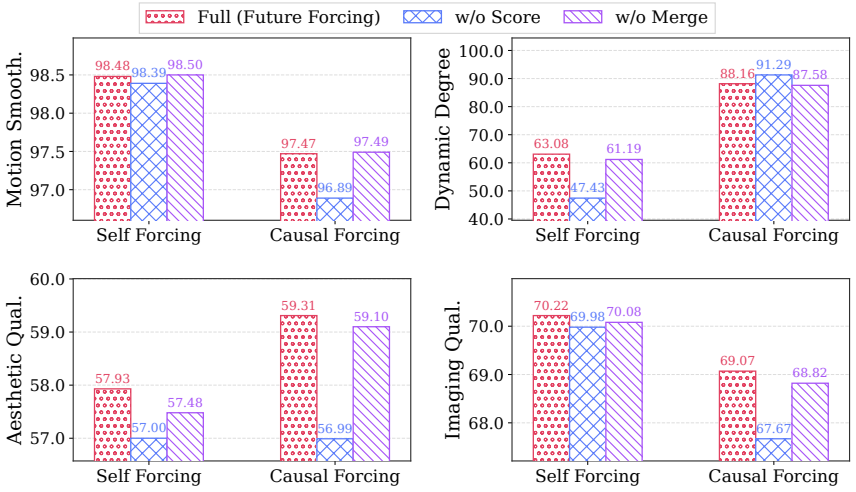


Figure 9: Additional ablation results

and Figure 8. Across all models and dimensions, we observe a consistent pattern: the canonical pre-RoPE query distributions remain concentrated and stable across autoregressive generation steps under a fixed prompt, whereas the corresponding RoPE-modulated query distributions exhibit substantially larger temporal variation. This result is consistent with the observation in Figure 2, and motivates using historical pre-RoPE statistics.

C.2 Additional Ablation Results.

We further report additional ablation results on VBench dimensions beyond the main text, including Motion Smoothness, Dynamic Degree, Aesthetic Quality, and Imaging Quality. As shown in Figure 9, the full Future Forcing maintains competitive or superior overall performance across both Self Forcing and Causal Forcing settings. In particular, removing the future-aware eviction score causes noticeable degradation on Aesthetic Quality and Imaging Quality under Causal Forcing, indicating that standard cache eviction fails to reliably preserve future-relevant tokens when the cache budget is limited. Removing the merging module also degrades several metrics, showing that directly discarding evicted entries introduces non-negligible information loss. Although individual metrics may fluctuate slightly, the full model provides the best overall trade-off across generation smoothness, dynamics, and visual quality. Both components contribute: future-aware eviction improves token selection, while merging reduces compression-induced distortion.

Table 8: Quantitative comparison of Future Forcing against baselines for **5-second short video generation** under reduced KV cache budgets. Ratios denote retained KV cache relative to full cache. ↑ indicates higher is better, while ↓ indicates lower is better. Best results are bold. The results show that Future Forcing incurs less information loss under the reduced KV cache budgets.

Backbone	Config	KV Cache Policy	PSNR↑	SSIM↑	LPIPS↓	Subject Consist.↑	Background Consist.↑	Aesthetic Quality↑	Imaging Quality↑
Self Forcing	Full Cache		-	-	-	94.18	94.72	63.71	69.97
	Retained 28.57%	Reward Forcing	18.438	0.665	0.174	94.29	94.76	63.69	70.01
		Deep Forcing	18.562	0.660	0.176	94.09	94.64	63.65	69.92
		Dummy Forcing	17.939	0.633	0.200	94.17	**94.81**	63.68	70.02
		Future Forcing	**19.258**	**0.690**	**0.158**	**94.38**	94.69	**63.75**	**70.14**
	Retained 42.86%	Reward Forcing	21.344	0.758	0.116	94.21	94.74	63.72	70.03
		Deep Forcing	20.943	0.744	0.122	94.14	94.69	63.67	69.90
		Dummy Forcing	19.333	0.683	0.161	**94.33**	94.75	63.69	70.03
		Future Forcing	**22.164**	**0.780**	**0.106**	**94.33**	**94.77**	**63.77**	**70.09**
Causal Forcing	Full Cache		-	-	-	94.33	94.82	64.50	70.85
	Retained 28.57%	Reward Forcing	16.290	0.664	0.199	94.24	**94.82**	64.39	**70.87**
		Deep Forcing	16.047	0.651	0.206	94.17	94.65	64.23	70.63
		Dummy Forcing	15.191	0.612	0.249	94.18	94.66	64.15	70.64
		Future Forcing	**16.704**	**0.681**	**0.189**	**94.36**	94.75	**64.48**	70.85
	Retained 42.86%	Reward Forcing	18.958	0.756	0.129	94.30	94.71	64.38	**70.88**
		Deep Forcing	18.417	0.729	0.139	94.29	94.75	64.37	70.77
		Dummy Forcing	16.571	0.659	0.199	94.31	94.76	64.39	70.74
		Future Forcing	**19.127**	**0.758**	**0.126**	**94.32**	**94.81**	**64.49**	70.85
LongLive	Full Cache		-	-	-	95.85	95.92	65.12	70.09
	Retained 28.57%	Reward Forcing	20.001	0.734	0.126	95.97	95.97	65.10	70.08
		Deep Forcing	15.932	0.595	0.270	**96.56**	**96.18**	64.98	69.87
		Dummy Forcing	18.572	0.689	0.167	95.98	95.95	65.01	70.01
		Future Forcing	**20.830**	**0.761**	**0.109**	95.90	95.97	**65.12**	**70.15**
	Retained 42.86%	Reward Forcing	23.313	0.824	0.075	**95.88**	95.94	**65.16**	70.11
		Deep Forcing	21.480	0.775	0.101	95.84	95.89	65.02	70.03
		Dummy Forcing	20.298	0.739	0.127	95.85	95.92	65.08	70.11
		Future Forcing	**24.035**	**0.842**	**0.065**	**95.88**	**95.95**	65.11	**70.14**

C.3 Additional Generation Quality Results.

We provide additional short-video generation results on VBench motion-related dimensions in Table 9. These experiments complement the reconstruction-based metrics reported in Table 3 by evaluating whether KV cache compression preserves temporal quality and motion dynamics within the training horizon. All methods are evaluated under the same reduced KV cache budgets, and the full-cache setting is included as a reference. Overall, Future Forcing maintains competitive or superior performance on both Motion Smoothness and Dynamic Degree across different backbones and retained ratios. On Self-Forcing, Future Forcing achieves the best Motion Smoothness at the 28.57% retained ratio and remains comparable to the strongest baselines at the 42.86% ratio, while preserving similar Dynamic Degree to the full-cache reference. On Causal-Forcing, Future Forcing consistently achieves the best or near-best Dynamic Degree under both retained ratios. On LongLive, Future Forcing recovers the full-cache Motion Smoothness and improves Dynamic Degree over most compressed baselines, especially under the 28.57% retained ratio. These results further show that Future Forcing preserves motion quality under constrained KV cache budgets.

C.4 Additional Efficiency and Memory Consumption Analysis Results.

We further evaluate the inference efficiency and peak GPU memory consumption of Future Forcing under different forcing strategies, with results shown in Figure 10. Compared with Naive and DeepForcing, Future Forcing achieves comparable output FPS with competitive peak GPU memory

Table 9: Quantitative comparison across models, retained KV cache ratios, and forcing strategies on additional VBench Motion Smoothness and Dynamic Degree. ↑ denotes higher is better.

Backbone	Config	Method	Motion Smoothness↑	Dynamic Degree↑
Self Forcing	Full Cache		98.32	61.91
	Retained 28.57%	Reward Forcing	98.29	62.41
		Deep Forcing	98.30	62.17
		Dummy Forcing	98.37	59.52
		Future Forcing	98.43	61.42
	Retained 42.86%	Reward Forcing	98.32	61.71
		Deep Forcing	98.29	62.31
		Dummy Forcing	98.35	60.52
		Future Forcing	98.35	61.78
Causal Forcing	Full Cache		97.67	76.87
	Retained 28.57%	Reward Forcing	97.59	78.46
		Deep Forcing	97.55	78.46
		Dummy Forcing	97.59	79.86
		Future Forcing	97.58	79.92
	Retained 42.86%	Reward Forcing	97.62	78.27
		Deep Forcing	97.64	77.77
		Dummy Forcing	97.60	78.07
		Future Forcing	97.64	78.17
LongLive	Full Cache		98.94	37.19
	Retained 28.57%	Reward Forcing	98.82	35.19
		Deep Forcing	98.80	39.48
		Dummy Forcing	98.75	35.59
		Future Forcing	98.95	37.69
	Retained 42.86%	Reward Forcing	98.94	36.39
		Deep Forcing	98.90	37.39
		Dummy Forcing	98.95	35.39
		Future Forcing	98.94	37.49

usage. Under Rolling-Forcing, Future Forcing attains an output FPS of 7.03, close to Naive and DeepForcing, while reducing peak memory from 21.62GB and 22.05GB to 20.53GB. Under Reward-Forcing, Future Forcing maintains a similar output FPS of 10.11, compared with 10.32 for Naive and 10.21 for DeepForcing, with peak memory of 16.71GB, compared with 16.64GB and 17.57GB, indicating modest inference overhead under constrained KV budgets.

Reason for lower memory consumption. In our experiments, under the Rolling-Forcing setting, Future Forcing uses a smaller KV cache budget than the baselines while achieving better or comparable performance. This further demonstrates the effectiveness of our future-aware KV cache policy: Future Forcing can retain more future-relevant historical tokens under a tighter memory budget, thereby reducing peak GPU memory consumption without sacrificing generation quality.

C.5 Custom Triton Contribution Analysis Results.

To evaluate the contribution of our custom Triton kernel, we compare Future Forcing with and without Triton across different forcing strategies, as shown in Figure 11. The Triton kernel consistently reduces inference time by 6.0%, 10.4%, 6.6%, and 8.9% under Self-Forcing, Causal-Forcing, Rolling-Forcing, and Reward-Forcing, respectively. This improvement mainly comes from accelerating the token-level importance scoring step, showing that our Triton implementation effectively reduces the computational overhead of future-aware scoring.

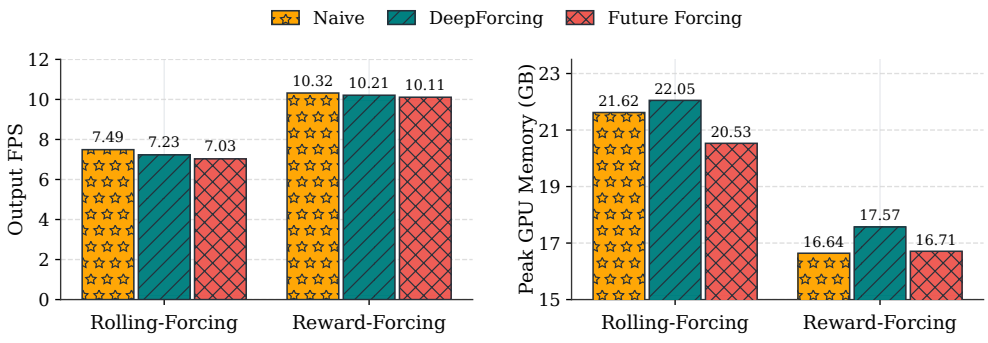


Figure 10: Efficiency and memory consumption across AR video backbones.

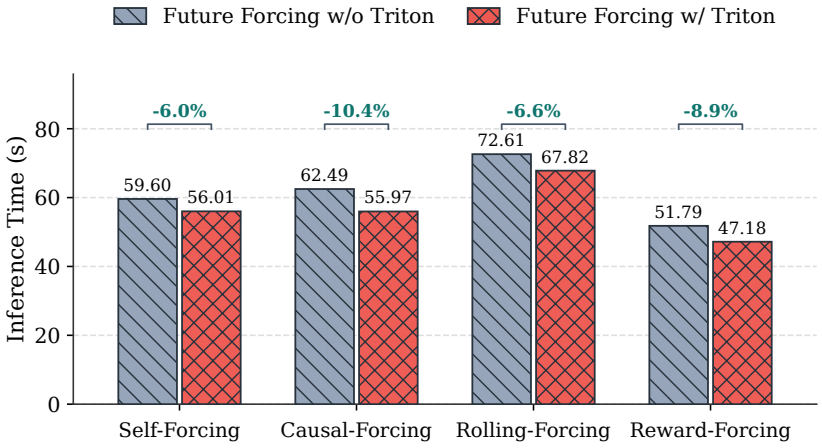


Figure 11: Contribution of the custom Triton kernel to inference efficiency.

C.6 Additional Hyperparameter Analysis Results.

To further examine the sensitivity of Future Forcing to its key hyperparameters, we conduct additional analyses on two representative autoregressive video generation backbones, Self-Forcing Huang et al. (2025) and Causal-Forcing Zhu et al. (2026). We consider two key factors in our future-aware KV cache policy design: the future query proxy horizon $\Delta_{\max}$, which controls the range of future positions used to construct future query proxies, and the future-history balance coefficient λ, which balances predicted future attention importance and historical attention evidence in the cache eviction score. We evaluate $\Delta_{\max} \in \{1, 3, 6, 9, 12\}$ and $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, where larger λ assigns greater weight to predicted future attention importance. For all experiments, we randomly select 20 prompts from the MovieGen Polyak et al. (2024) prompt set, keep all other hyperparameters fixed, and use the same initial noise for each prompt across different hyperparameter settings to ensure fair comparison. We save the generated videos for every configuration and evaluate them using VBench-Long Huang et al. (2024) metrics across all evaluation dimensions.

As shown in Table 10, Table 11, Table 12 and Table 13, Future Forcing remains robust across a wide range of hyperparameter settings on both backbones. Varying $\Delta_{\max}$ leads to only moderate changes, indicating that the future query proxy construction is not overly sensitive to the exact horizon. For λ, assigning more weight to future attention importance generally improves motion-related performance, especially on Self-Forcing, where a larger λ will increase dynamic degree while only slightly reducing subject and background consistency. On Causal-Forcing, performance remains stable across different λ values, with $\lambda = 0.7$ providing a strong balance among motion, consistency, and visual quality. Overall, these results show that Future Forcing consistently maintains strong video quality and temporal consistency under fixed KV cache budgets across most hyperparameter configurations.

Table 10: Hyperparameter analysis of the future query proxy horizon $\Delta_{\max}$ on the Self-Forcing backbone, evaluated on 20 randomly selected prompts from MovieGen Polyak et al. (2024).

Value	Dynamic Degree↑	Motion Smoothness↑	Overall Consistency↑	Imaging Quality↑	Aesthetic Quality↑	Subject Consistency↑	Background Consistency↑
1	56.00	98.55	25.20	72.05	55.93	97.69	96.52
3	61.00	98.46	25.58	72.29	55.56	97.61	96.50
6	56.67	98.49	25.59	72.80	56.27	97.67	96.51
9	56.67	98.49	25.59	71.80	56.25	97.66	96.49
12	54.00	98.50	25.83	72.42	55.82	97.70	96.48

Table 11: Hyperparameter analysis of the future-history balance coefficient λ on the Self-Forcing backbone, evaluated on 20 randomly selected prompts from MovieGen Polyak et al. (2024).

Value	Dynamic Degree↑	Motion Smoothness↑	Overall Consistency↑	Imaging Quality↑	Aesthetic Quality↑	Subject Consistency↑	Background Consistency↑
0.1	53.00	98.53	25.71	72.10	55.81	97.60	96.57
0.3	51.67	98.53	25.42	72.32	55.76	97.70	96.57
0.5	56.67	98.49	25.59	72.80	56.27	97.67	96.51
0.7	62.00	98.35	25.51	72.15	55.58	97.57	96.37
0.9	65.67	98.25	25.23	72.25	55.99	97.28	96.20

C.7 Difference to PaFu-KV

PaFu-KV Chen et al. (2026) and our Future Forcing both aim to improve KV cache management for autoregressive video generation by incorporating future-aware token importance. However, the two methods differ fundamentally in how future relevance is obtained and used. **PaFu-KV is a training-based** method that relies on a bidirectional teacher model to provide past- and future-informed salience supervision, and further trains a lightweight salience estimation head to predict token-level importance during autoregressive inference. In contrast, **Future Forcing is fully training-free**: it does not require an additional teacher model, auxiliary salience head, or any extra fine-tuning. Instead, Future Forcing exploits the empirical stability of canonical pre-RoPE query distributions across autoregressive steps to construct future query proxies, which are then used to estimate future attention importance directly within the pretrained AR video generator. Moreover, while PaFu-KV primarily performs salience-based cache eviction, Future Forcing further introduces future-aware KV cache merging, which preserves information from evicted entries by merging them according to their predicted future attention profiles. This design reduces the attention-output distortion caused by token removal and enables future-aware KV cache compression without retraining the model. **Since the source code of PaFu-KV is not publicly available, we do not include it in direct comparisons.**

D Experiments Compute Resources.

All experiments are conducted on NVIDIA A100 GPUs with 80GB memory. Since our KV cache policy Future Forcing is training-free, the computational cost mainly comes from autoregressive inference and KV-cache scoring and merging during the autoregressive video generation process.

E Prompt Changes

Our query statistics are estimated within a fixed-prompt generation session. Changing the prompt can shift the pre-RoPE query distribution: on LongLive, normalized Wasserstein distance increases from 0.2529 under a fixed prompt to 0.6868 after a prompt change. For interactive generation, the query statistics should therefore be refreshed together with the prompt-dependent KV cache. Evaluating this integration across successive prompt changes is left for future work.

Table 12: Hyperparameter analysis of the future query proxy horizon $\Delta_{\max}$ on the Causal-Forcing backbone, evaluated on 20 randomly selected prompts from MovieGen Polyak et al. (2024).

Value	Dynamic Degree↑	Motion Smoothness↑	Overall Consistency↑	Imaging Quality↑	Aesthetic Quality↑	Subject Consistency↑	Background Consistency↑
1	94.33	97.13	23.84	71.35	57.53	95.93	95.39
3	90.00	97.07	24.01	71.22	57.55	96.01	95.41
6	91.33	97.08	24.08	71.43	57.79	96.11	95.50
9	94.33	97.23	24.20	71.18	57.70	96.05	95.56
12	90.67	97.27	24.31	71.12	57.59	95.96	95.52

Table 13: Hyperparameter analysis of the future-history balance coefficient λ on the Causal-Forcing backbone, evaluated on 20 randomly selected prompts from MovieGen Polyak et al. (2024).

Value	Dynamic Degree↑	Motion Smoothness↑	Overall Consistency↑	Imaging Quality↑	Aesthetic Quality↑	Subject Consistency↑	Background Consistency↑
0.1	89.00	97.14	24.03	71.32	58.12	96.10	95.47
0.3	92.00	97.11	23.66	71.28	57.60	96.12	95.43
0.5	91.33	97.08	24.08	71.43	57.79	96.11	95.50
0.7	92.33	97.25	23.98	71.71	57.70	96.17	95.52
0.9	89.33	97.23	23.92	71.44	57.90	96.11	95.54

F Additional Visualization Results.

To further facilitate visual comparison, we provide additional qualitative results comparing Future Forcing with representative baselines in Figure 13, Figure 14, Figure 15, Figure 16, and Figure 17.

Discussion about the dynamic degree. For the Causal-Forcing backbone, Future Forcing obtains a lower dynamic degree than some baselines, as shown in Table 2 and Figure 13. A lower score alone does not determine motion quality. In the visualized examples, higher dynamic degree of the native KV-cache strategy and Causal-Forcing+Deep-Forcing often comes from **abrupt, undesired scene changes** during unstable long-horizon generation, rather than meaningful object or camera motion. In contrast, Future Forcing avoids such transitions and achieves **reasonable and stable dynamics with better long-horizon consistency**, as shown in Figure 14, Figure 15, Figure 16, Figure 17.

G Quantitative Analysis of Pre-RoPE Query Stability

To further quantify the cross-frame stability of pre-RoPE queries, we compute the average pairwise Wasserstein distance between frame-wise query distributions. To avoid scale-dependent comparisons, we normalize the distance by the pooled standard deviation and report normalized W_1. Lower values indicate stronger cross-frame distributional stability. As shown in Table 14, pre-RoPE queries consistently exhibit much smaller normalized W_1 than RoPE-modulated queries across different autoregressive video generation backbones, query dimensions, and prompts. Across all evaluated settings, pre-RoPE queries achieve an average normalized W_1 of only 0.123σ, whereas RoPE-modulated queries reach 1.176σ, showing nearly one order of magnitude larger distributional drift.

H Pre-RoPE Stability under Highly Dynamic Scenarios

To further examine whether pre-RoPE query stability holds under highly dynamic visual conditions, we construct three challenging prompts involving rapid camera motion, complex object interactions, and substantial scene dynamics. Figure 18 visualizes pre-RoPE and RoPE-modulated query distributions under these scenarios. Even under strong motion, fast viewpoint changes, and dense dynamic interactions, pre-RoPE query distributions remain substantially more stable than their RoPE-modulated counterparts. The three prompts are listed below.

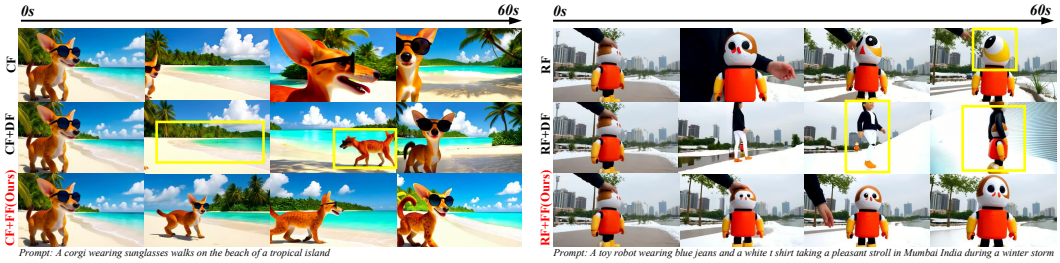


Figure 12: Visual comparison of Future Forcing and baselines for 60-second long-video generation.

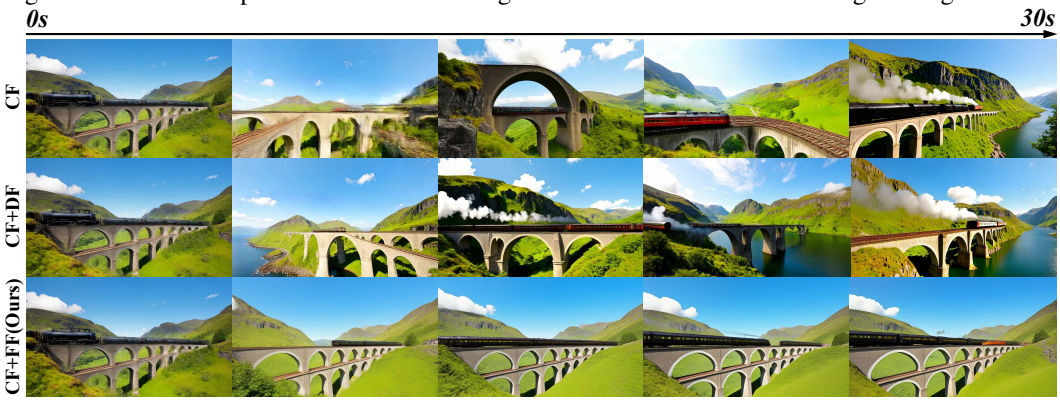


Figure 13: Additional visualization results for Causal-Forcing in 30-second video generation.

Table 14: Normalized pairwise Wasserstein distance W_1 of pre-RoPE and RoPE-modulated queries.

Backbone	Dim.	Pre-RoPE Query			RoPE-Modulated Query		
		Prompt 0	Prompt 1	Prompt 2	Prompt 0	Prompt 1	Prompt 2
Self Forcing	0	0.098	0.122	0.204	1.192	1.192	1.192
	5	0.092	0.092	0.094	1.187	1.187	1.188
Causal Forcing	0	0.139	0.147	0.152	1.190	1.188	1.191
	5	0.103	0.103	0.150	1.185	1.185	1.185
LongLive	0	0.126	0.113	0.207	1.192	1.193	1.194
	5	0.097	0.167	0.059	1.187	1.188	1.189
Reward Forcing	0	0.126	0.088	0.195	1.149	1.148	1.150
	5	0.066	0.101	0.068	1.110	1.086	1.102
Rolling Forcing	0	0.168	0.083	0.206	1.192	1.194	1.192
	5	0.093	0.093	0.130	1.187	1.189	1.186

Highly Dynamic Prompts

1. A drone-level chase through a neon night market, weaving between sprinting cyclists, dancing crowds, flying paper lanterns, and sudden rain as the camera whips around tight corners.
2. A storm rescue on a rocky coast, with a helicopter spotlight sweeping across crashing waves, a lifeboat pitching violently, lightning flashes, spray hitting the lens, and people moving in several directions.
3. A rooftop parkour race at sunset, jumping over vents and glass skylights while trains pass below, birds scatter, shadows stretch fast, and the camera swings from wide aerial views to close handheld pursuit.



Figure 14: Additional visualization results for Reward-Forcing in 30-second video generation.

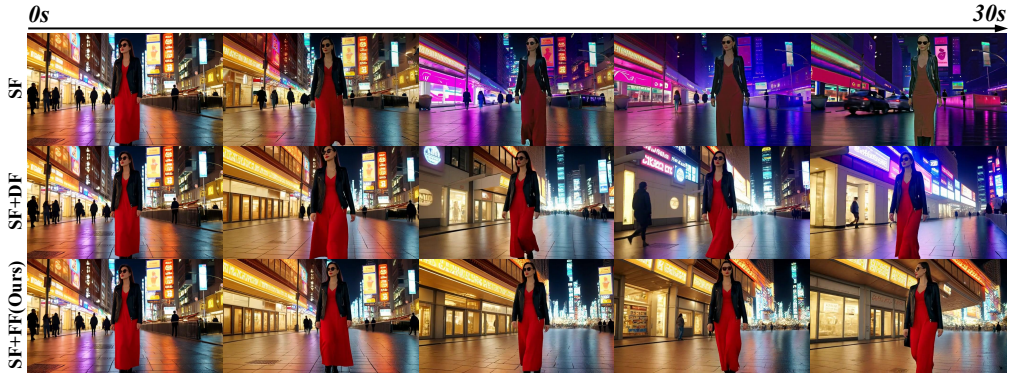


Figure 15: Additional visualization results for Self-Forcing in 30-second video generation.

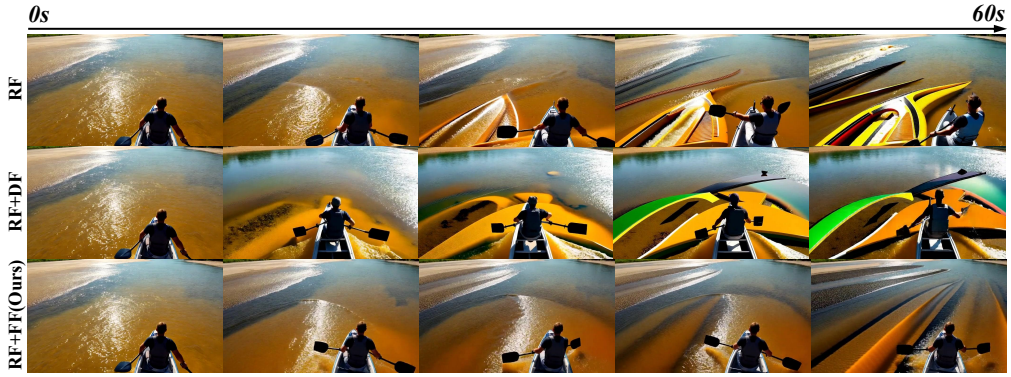


Figure 16: Additional visualization results for Reward-Forcing in 60-second video generation.

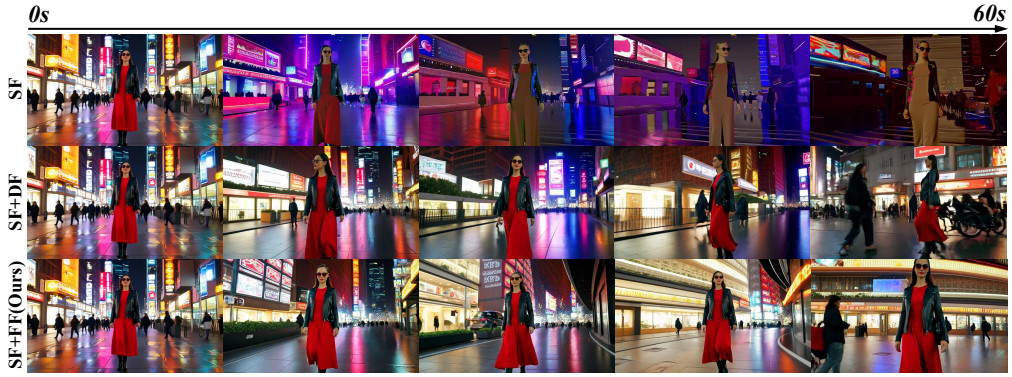
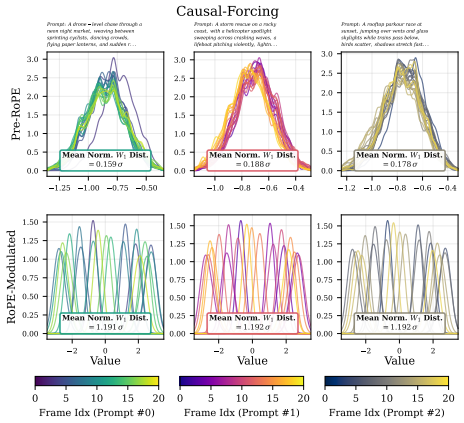
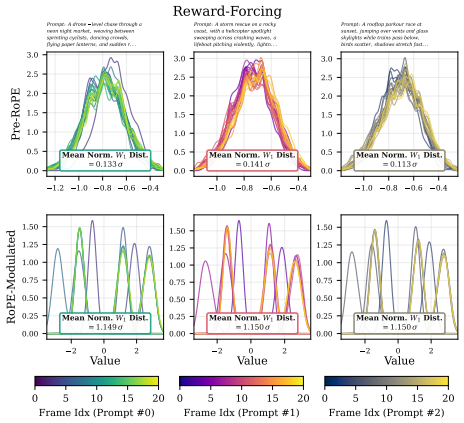


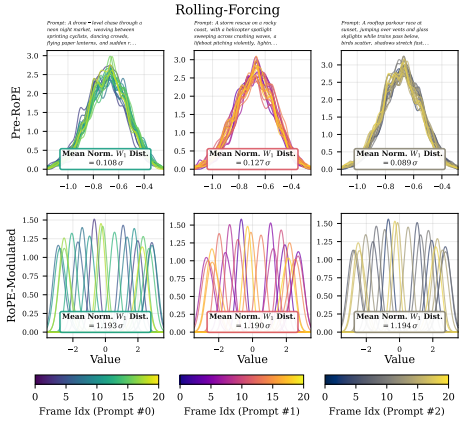
Figure 17: Additional visualization results for Self-Forcing in 60-second video generation.



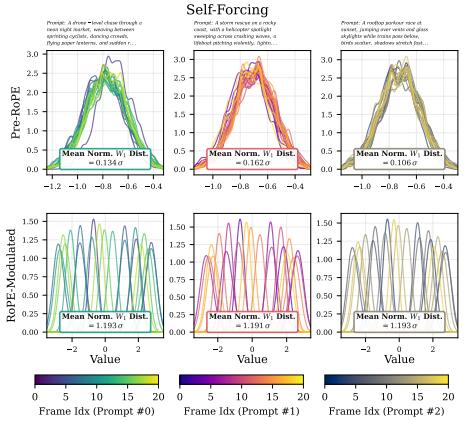
(a) Causal Forcing



(b) Reward Forcing



(c) Rolling Forcing



(d) Self Forcing

Figure 18: Pre-RoPE and RoPE-modulated query distributions under **highly dynamic scenarios** across representative autoregressive video generation models.

I Additional Proxy and Compression Diagnostics

Unless otherwise stated, proxy construction, ranking, and generation comparisons use 20 randomly selected MovieGen prompts; layer-wise measurements use 10 prompts, and the broader stability study uses 128 prompts. The 128-prompt study measures distributional stability; proxy accuracy is evaluated separately. SF, CF, and RF denote Self-, Causal-, and Reward-Forcing. Metric abbreviations are SC (subject consistency), BC (background consistency), MS (motion smoothness), IQ (imaging quality), AQ (aesthetic quality), and OC (overall consistency).

I.1 Proxy Accuracy and Historical-Attention Controls

For ranking evaluation, reference future queries are collected along compressed-cache rollouts. The resulting correlations measure ranking agreement on those trajectories. We compare the proxy with realized future queries using relative L_2 error and cosine similarity. These measurements complement the ranking comparison in Table 5. The full mixed score has ranking correlations 0.6732 (SF) and 0.6657 (CF), while future-only scoring achieves 0.6545 and 0.6592.

To distinguish proxy scoring from smoothing past attention, we replace it with EMA or averaging of historical attention. Future Forcing improves subject and background consistency in both backbones, but not every motion score. EMA of historical attention yields ranking correlations 0.3516 (SF) and 0.3562 (CF), compared with 0.3304 and 0.4315 for historical averaging.

Table 15: Query approximation (20 prompts). Lower L_2 / higher cosine is better.

Proxy	$L_2 \downarrow$	Cos.↑
SF / Post EMA	0.8595	0.4393
SF / Post Avg	0.8375	0.4147
SF / Ours	**0.4512**	**0.8796**
CF / Post EMA	0.8470	0.4531
CF / Post Avg	0.8259	0.4229
CF / Ours	**0.4264**	**0.8863**

Table 16: Historical-attention controls (30s, 20 prompts).

Variant	SC	BC	MS
SF / Attn EMA	96.70	96.16	98.10
SF / Attn Avg	96.79	96.26	98.24
SF / Ours	**97.67**	**96.51**	**98.49**
CF / Attn EMA	94.92	93.97	**97.23**
CF / Attn Avg	94.97	95.10	97.15
CF / Ours	**96.11**	**95.50**	97.08

I.2 Layer-wise and Alternative-Proxy Analysis

We measure cosine similarity between window-averaged queries and realized future queries at six layers. Pre-RoPE averages have higher similarity at each probed layer.

Table 17: Layer-wise cosine similarity (10 prompts). Pre-/post-RoPE are compared.

Layer	Self-Forcing		Causal-Forcing	
	Pre	Post	Pre	Post
0	.987	.482	.936	.483
5	.918	.469	.916	.460
10	.790	.349	.748	.415
15	.818	.408	.850	.458
25	.794	.328	.788	.365
29	.973	.272	.970	.321

Table 18: Proxy construction (30s, 20 prompts). EMA: 0.9; sampling: Gaussian.

Proxy	SC	BC	MS
SF / recent	97.08	96.00	98.35
SF / EMA	97.46	**96.64**	**98.60**
SF / sampling	96.85	96.39	98.29
SF / Ours	**97.67**	96.51	98.49
CF / recent	95.01	94.96	96.74
CF / EMA	96.05	95.20	96.95
CF / sampling	95.71	95.02	96.73
CF / Ours	**96.11**	**95.50**	**97.08**

Averaging and EMA perform comparably; we adopt averaging for its simplicity.

I.3 Window and History-Weight Sensitivity

Window size w is measured in latent frames. We vary the query window and future-history balance while retaining the remaining configuration. Small windows can be more sensitive to local variation, but performance is not monotonic in window size. Including a history term is competitive with or better than future-only scoring across the reported quality dimensions.

Table 19: Window-size sweep (20 prompts). The default is $w = 12$ latent frames.

Model	w	SC	BC	MS
SF	3	96.98	96.35	98.31
SF	6	97.16	**96.54**	98.37
SF	9	97.06	96.42	98.25
SF	12	**97.67**	96.51	**98.49**
SF	15	97.32	96.51	98.47
CF	3	95.85	94.79	96.82
CF	6	95.89	94.82	96.83
CF	9	95.87	95.12	97.02
CF	12	96.11	**95.50**	97.08
CF	15	**96.19**	94.99	**97.17**

Table 20: Future-history weight sweep. The default is $\lambda = 0.5$.

Model	λ	OC	IQ	AQ
SF	.3	25.42	72.32	55.76
SF	.5	**25.59**	**72.80**	**56.27**
SF	.7	25.51	72.15	55.58
SF	.9	25.23	72.25	55.99
SF	1.0	25.30	72.13	55.87
CF	.3	23.66	71.28	57.60
CF	.5	**24.08**	71.43	57.79
CF	.7	23.98	**71.71**	57.70
CF	.9	23.92	71.44	**57.90**
CF	1.0	23.72	71.38	57.70

I.4 Distinctions from LLM Query-Statistics Methods

Expected Attention. Expected Attention Devoto et al. (2025) estimates future attention through Gaussian modeling of RoPE-encoded queries in LLMs. Future Forcing instead averages recent pre-RoPE queries within the current video, rotates the proxy to future positions, and uses its profiles for eviction and merging. Table 18 compares averaging with Gaussian sampling.

TriAttention. TriAttention Mao et al. (2026a) uses offline-calibrated pre-RoPE query/key centers to estimate distance-dependent importance in LLMs. Future Forcing estimates prompt-conditioned statistics online. Cross-prompt Wasserstein distances exceed within-prompt distances on both SF and RF (Table 22), supporting video-specific query estimates over a shared center.

Transferred-policy comparison. We adapt Expected Attention and TriAttention from LLM KV-cache compression to autoregressive video generation. Expected Attention uses historical queries; TriAttention uses 30 offline calibration prompts. Future Forcing achieves higher SC, BC, and MS on both SF (Table 4) and CF (Table 21) under these adaptations.

Table 21: KV-cache policies adapted from LLMs to Causal-Forcing (20 prompts).

Policy	SC	BC	MS
Expected Attn.	95.58	95.24	96.99
TriAttention	94.95	94.79	96.73
Ours	**96.11**	**95.50**	**97.08**

Table 22: Normalized Wasserstein distance within and across prompts.

Backbone	Within	Across
Self-Forcing	0.1202	0.4337
Reward-Forcing	0.1268	0.4302

I.5 RoPE Correction and Merging

The correction comparison in Table 7 also yields IQ scores of 71.65, 72.38, and 72.80 for SF, and 71.22, 71.30, and 71.43 for CF (native, native with correction, and Future Forcing with correction).

Average merge acceptance rates among evicted entries are 6.88% (SF), 5.80% (CF), and 11.71% (RF). Thus, only a minority of evicted entries are merged.

Table 23: Merge-threshold sensitivity. Default $\tau = 0.95$. Best results are bold.

τ	Self-Forcing			Causal-Forcing		
	SC	BC	MS	SC	BC	MS
0.99	97.58	96.47	98.35	**96.45**	95.43	97.06
0.95	**97.67**	96.51	**98.49**	96.11	**95.50**	**97.08**
0.90	97.54	**96.68**	98.27	95.32	95.15	97.02
0.85	97.61	96.41	98.22	95.65	95.20	96.86
0.80	97.40	96.49	98.19	95.46	95.07	96.91

I.6 Complex Scenes and Distribution Changes

Five additional challenging prompts describe a crowded night market, a busy kitchen, rainforest wildlife, a futuristic robot battle, and an underwater coral ecosystem. These examples involve dense objects, interactions, and fine detail. Their normalized pre-/post-RoPE distances are 0.1421/1.1941 (SF), 0.1905/1.1928 (CF), 0.1263/1.1942 (Rolling-Forcing), and 0.1347/1.1516 (RF). These results complement the 128-prompt evaluation.

Merging improves several metrics, but CF subject consistency is essentially unchanged.

Prompt-change results and the corresponding cache-refresh setting are discussed in Appendix E.

Table 24: Merging on five complex scenes.

Variant	SC	BC	MS
SF / merging	**95.42**	**95.26**	**97.53**
SF / no merging	94.96	95.09	97.11
CF / merging	93.50	**93.15**	**95.86**
CF / no merging	**93.51**	92.86	95.80

Table 25: Merging acceptance rates.

Backbone	Acceptance
Self-Forcing	6.88%
Causal-Forcing	5.80%
Reward-Forcing	11.71%

I.7 Quality over Long-Horizon Generation

Table 26 reports SC and IQ in three 20s intervals of 60s videos for 20 prompts. Future Forcing achieves the highest SC throughout and the highest IQ in the final two intervals.

Table 26: 60s temporal quality on Causal-Forcing (20 prompts). Each interval reports subject consistency (SC) and imaging quality (IQ).

Policy	0–20s		20–40s		40–60s	
	SC	IQ	SC	IQ	SC	IQ
Native	88.84	**71.82**	88.04	71.08	87.28	70.13
Deep Forcing	87.93	71.67	88.54	71.24	86.67	71.19
Future Forcing	**89.31**	71.53	**90.06**	**71.41**	**89.99**	**71.22**