TEMPOKV: PREDICTIVE KV CACHE QUANTIZATION FOR AUTOREGRESSIVE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive (AR) video diffusion models enable streaming generation of long videos but suffer from significant memory overhead due to the rapidly growing KV cache. Existing KV cache quantization methods are primarily designed for LLMs and transfer poorly to video models, where activations exhibit different statistics and strong spatiotemporal redundancy. We observe that the video KV cache is largely predictable over time, with a clear asymmetry. Keys are highly predictable from their temporal history, while values depend primarily on the current key and need a short history of preceding values. Based on these observations, we propose **TempoKV**, a training-free predictive KV cache quantization framework for AR video generation. TempoKV predicts keys and values according to their distinct dependencies and stores only the residuals at low precision. Both predictors are lightweight, calibrated once offline by linear regression on a small set of BF16 samples, and remain fixed across prompts and generation runs. Experiments on Self-Forcing, Causal Forcing, and LongCat-Video-13B show that TempoKV consistently achieves higher generation quality than existing KV cache quantization methods at comparable or higher compression ratios, reaching up to $6.71\times$ KV cache compression with the best overall quality–efficiency tradeoff. *Our code is in supplementary material and will be released on Github.*

1 INTRODUCTION

Recent video diffusion models (Wan et al., 2025; Wu et al., 2025) have achieved high generation quality, but extending generation to long-duration videos still requires significant computation and memory. Autoregressive approaches address this limitation by generating videos progressively while reusing previously generated context (Huang et al., 2025a; Zhu et al., 2026; Liu et al., 2026). However, the resulting *KV cache* becomes another bottleneck, as its size grows linearly with generation duration and quickly dominates GPU memory. For example, in Causal Forcing (Zhu et al., 2026), generating a 45-second 480p video requires 51.76 GB solely for the KV cache when stored in BF16, exceeding the memory of most consumer GPUs and limiting further extension of the generation horizon.

Existing KV cache quantization methods (Liu et al., 2024; Ashkboos et al., 2024; Kang et al., 2024; Hooper et al., 2024) are developed primarily for large language models (LLMs), focusing on the numerical characteristics of language KV caches. Autoregressive video diffusion models, however, exhibit different activation distributions and strong spatiotemporal redundancy across consecutive latent frames, making the direct transfer of these methods to video models suboptimal.

Recent work, QVG (Xi et al., 2026), takes an important step toward video-specific quantization by grouping similar video tokens and subtracting group centroid. Notably, QVG observes different compression behavior between keys (K) and values (V): the same smoothing operation reduces the quantization error of keys much more than that of values. Despite this discrepancy, it applies the same similarity-based formula to both. This difference suggests that video KV redundancy is not fully captured by token similarity alone. To further investigate, we analyze how the KV cache changes over latent frames and whether this evolution is predictable. As shown in the top row of Fig. 1, keys evolve smoothly and predictably along the latent frames, while values are markedly more irregular.

We find a clear asymmetry between keys and values. The current key can be accurately predicted from preceding keys at the same spatial position: a linear probe on the key history reaches a cosine similarity of 0.877. Values show a different pattern. Previous values alone are far less predictive of

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: **Magnitudes of KV caches and their prediction residuals on Self-Forcing.** Top: the original KV cache magnitudes at representative layers and heads. Bottom: the corresponding residual magnitudes after applying TempoKV. Prediction reduces the dynamic range of both keys and values, suppressing outliers across latent frames and channels.

the current value, reaching a cosine similarity of only 0.637, which rises to 0.881 once the aligned current key is included, i.e., the key at the same latent frame and spatial position.

Motivated by this observation, we introduce **TempoKV**, a predictive KV cache quantization framework for autoregressive video generation. TempoKV first predicts the key K_t from reconstructed historical keys, then quantizes only its residual. After reconstructing the current key, TempoKV uses the reconstructed key $\widehat{K}_t$ and reconstructed historical values to predict the current value V_t, and again quantizes the remaining residual. As shown in Fig. 1, prediction suppresses prominent outliers and narrows the dynamic range, producing low-magnitude residuals that are easier to quantize.

This asymmetry has two further implications for quantization. First, the aligned current K provides the primary predictive signal for V, while additional V history offers only limited benefit. In contrast, K relies mainly on its own temporal history and benefits from a longer context. This suggests using a shorter temporal context for V than for K. Second, the asymmetry also affects precision allocation. Quantization error in K not only degrades its own reconstruction, but is also carried into the subsequent V prediction through the cross predictor. This makes K more sensitive to error. Accordingly, we assign K a higher precision than V except at the lowest-precision INT2 setting.

Experiments on Self-Forcing (Huang et al., 2025a), Causal Forcing (Zhu et al., 2026), and LongCat-Video-13B (Meituan LongCat Team et al., 2025) validate the asymmetric predictive structure and demonstrate its effectiveness for low-bit KV compression. TempoKV improves generation quality over existing low-bit KV cache quantization methods at comparable or higher compression ratios, reaching up to $\mathbf{6.71}\times$. These gains remain robust across diverse autoregressive generation frameworks and resolutions, indicating that the KV asymmetry is not specific to a single backbone.

In summary, our main contributions are:

- **Asymmetric Predictive Dependencies.** We identify that K and V exhibit asymmetric predictive dependencies across latent frames. Keys are highly predictable from their history, whereas values benefit from the aligned current key in addition to previous values. Through layer-wise and temporal analysis, we show that this asymmetry is consistent across layers and throughout generation.
- **TempoKV Framework.** Inspired by this asymmetry, we propose TempoKV, a training-free predictive quantization framework for autoregressive video diffusion models. TempoKV predicts keys from their temporal history and values through a key-to-value cross predictor, assigns higher precision to keys, and quantizes only the remaining residuals. The predictors are fitted offline by linear regression on short BF16 traces with minimal overhead.
- **Strong Quality–Efficiency Tradeoff.** We evaluate TempoKV across diverse architectures and configurations. TempoKV outperforms prior low-bit KV cache quantization methods on all fidelity metrics, achieving competitive or higher compression with low runtime overhead.

2 Related Work

2.1 Autoregressive Video Generation

Recent autoregressive video models generate extended sequences causally while reusing previously generated context. To bridge bidirectional diffusion and causal generation, CausVid (Yin et al., 2025) adapts pretrained bidirectional diffusion models into autoregressive generators through distribution matching distillation, while Self-Forcing (Huang et al., 2025a) reduces exposure bias by training on autoregressive rollouts with self-generated outputs. Self-Forcing++ (Cui et al., 2026) further mitigates long-horizon error accumulation using teacher guidance on self-generated videos. Other works extend long-horizon generation through joint denoising and attention sinks (Liu et al., 2026), frame-level autoregression with short-window attention (Yang et al., 2026), chunk-wise prediction (Teng et al., 2025), frame-context packing (Zhang & Agrawala, 2025), or explicit memory mechanisms (Xiao et al., 2025; Henschel et al., 2024). LongCat-Video-13B (Meituan LongCat Team et al., 2025) adopts video continuation pretraining with coarse-to-fine generation. However, these AR approaches rely on historical context. Consequently, the linearly growing KV cache rapidly dominates GPU memory, becoming a severe bottleneck for high-resolution and long-video generation.

2.2 KV Cache Eviction and Compaction

To mitigate the growing KV-cache memory bottleneck, early efforts in LLMs selectively retain important context, such as attention sinks together with a recent window (Xiao et al., 2024), recent and heavy-hitter tokens (Zhang et al., 2023), or clustered important KV positions identified from an observation window (Li et al., 2024). Recent video-specific methods adapt cache management to autoregressive video. PaFu-KV (Chen et al., 2026) models time-heterogeneous token importance with a Salience Estimation Head distilled from a bidirectional teacher, while FutureForcing (Luo et al., 2026) uses future query proxies to guide token eviction and merging. Others use heterogeneous memory allocation (DummyForcing (Guo et al., 2026)) or hybrid static and dynamic pruning (Forcing-KV (Ji et al., 2026)), while PackCache (Li et al., 2026) dynamically compacts the cache through condition anchoring and cross-frame decay modeling.

2.3 KV Cache Quantization

LLM KV-cache quantization spans several directions. KIVI (Liu et al., 2024) and KVQuant (Hooper et al., 2024) adapt quantization to the distinct distributions of KV through asymmetric quantization granularity and non-uniform quantization. QuaRot (Ashkboos et al., 2024) and RotateKV (Su et al., 2025) use rotations to mitigate the effect of outliers, while SKVQ (Duanmu et al., 2024) and ZipCache (He et al., 2024) retain recent or salient tokens at higher precision. PQCache (Zhang et al., 2024) and CommVQ (Li et al., 2025) further explore product and vector quantization. Despite this diversity, these methods target language-model KV statistics. QVG (Xi et al., 2026) extends low-bit KV quantization to autoregressive video generation by leveraging spatiotemporal redundancy with Semantic-Aware Smoothing, followed by coarse-to-fine Progressive Residual Quantization. TempoKV takes a complementary predictive view: rather than grouping similar tokens, it models the temporal dependencies of K and V and uses their asymmetry to guide the quantization design.

3 Method

3.1 Preliminaries

An autoregressive video model generates a sequence of T latent frames, each containing S spatial tokens with consistent spatial indexing across time. For a given attention layer and head, let $K_{t,s}, V_{t,s} \in \mathbb{R}^D$ denote the key and value at spatial position s in latent frame t, where D is the head dimension. We use standard round-to-nearest quantization. For a tensor X, its b-bit affine quantization and reconstruction are defined as

$$Q_b(X) = \left\lfloor \frac{X}{s_X} \right\rceil + z_X, \qquad \hat{X} = \big(Q_b(X) - z_X\big)s_X, \tag{1}$$

where $s_X = (\max(X) - \min(X))/(2^b - 1)$ and $z_X = \lfloor -\min(X)/s_X \rceil$.

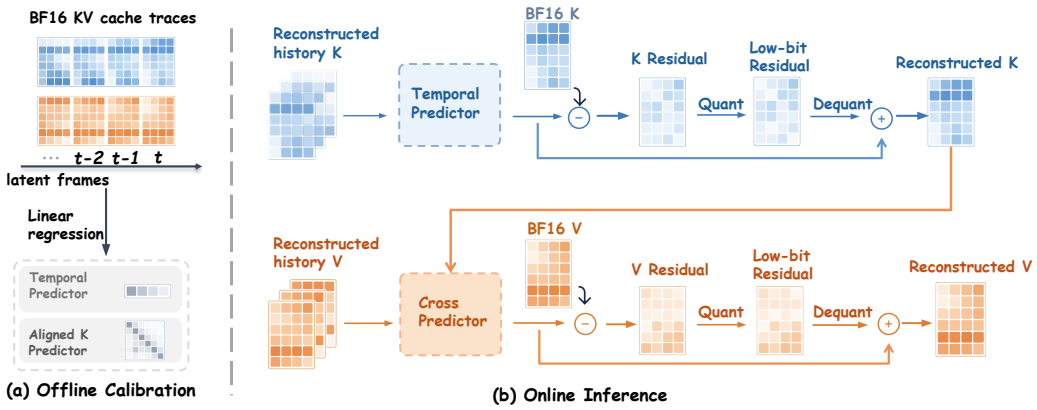


Figure 2: **TempoKV Framework.** (a) Offline Calibration: a clean BF16 KV trace is collected from 8 calibration prompts, and predictors are fitted on it by linear regression. (b) Online Inference: $\widetilde{K}_{t,s}$ is predicted from the reconstructed K history, and $\widetilde{V}_{t,s}$ from the aligned $\widehat{K}_{t,s}$ together with the reconstructed V history. Only the low-bit residual $R_{t,s}$ is stored.

3.2 Asymmetric Predictability of Video KV Caches

KV caches are highly correlated across the latent frames. To further examine this dependency and characterize how it differs between K and V, we fit four linear probes and measure the cosine similarity between the predicted and original caches. The first predicts $K_{t,s}$ from the key history at the same position, and the second predicts $V_{t,s}$ from the value history. The other two add one more input, the other cache of the current frame: $V_{t,s}$ for K prediction and $K_{t,s}$ for V prediction.

Fig. 3 reveals a clear asymmetry between K and V. The key history predicts $K_{t,s}$ with a cosine similarity of **0.877**. The value history predicts $V_{t,s}$ with only 0.637. Adding $V_{t,s}$ moves the K probe from 0.877 to 0.910, a gain of 0.033. Adding $K_{t,s}$ moves the V probe from 0.637 to **0.881**, a gain of **0.244**. The key history alone already provides an accurate estimate of the current key, while the value history is considerably less predictive of the current value. The same trend is observed in each layer and at every frame. In Fig. 3(b), $K_{t,s}$ improves V prediction in all 30 layers, by 0.148 to 0.307. The reverse gain, from $V_{t,s}$ to K, ranges from 0.009 to 0.056. Fig. 3(c) follows both probes along the latent frames. Both grow more accurate as generation proceeds, and the change is larger for V. The gain from $K_{t,s}$ to V remains higher than the reverse gain throughout the trajectory.

This asymmetry determines the direction of our cross predictor. We first predict $K_{t,s}$ from the key history, then predict $V_{t,s}$ from the reconstructed key $\hat{K}_{t,s}$ and the value history. The two cross directions cannot be used simultaneously at decoding, since each one needs the other cache to be available first. We keep the direction $K \to V$ that provides much larger gain.

Keys and values also require different precision. Key quantization perturbs the attention logits before softmax and can alter the resulting attention weights. Value quantization, in contrast, affects the attention output only after the weights are formed. This asymmetry is a property of attention. The cross predictor introduces a second and independent source of asymmetry. The quantization error in K not only affects the reconstructed key itself but also propagates to the value prediction via the cross predictor. Therefore, we assign higher precision to keys than values.

The observation further motivates different temporal history lengths. K is predicted from the key history alone and keeps improving as that history grows. V already receives a strong input from the current key, and extra value history adds little. A longer history also increases predictor parameters and keeps more data in the cache during decoding. Thus, we use $h_K = 2$ and $h_V = 1$ by default.

3.3 Predictive Residual Quantization

TempoKV framework. TempoKV predicts the KV cache from its own dependencies and quantizes only the residual (Fig. 2). Let $\widetilde{X}_{t,s}$ and $\widehat{X}_{t,s}$ denote the prediction and reconstruction, respectively.

$$R^X_{t,s} = X_{t,s} - \widetilde{X}_{t,s}, \qquad \widehat{X}_{t,s} = \widetilde{X}_{t,s} + D_b\big(Q_b(R^X_{t,s})\big)\,. \tag{2}$$

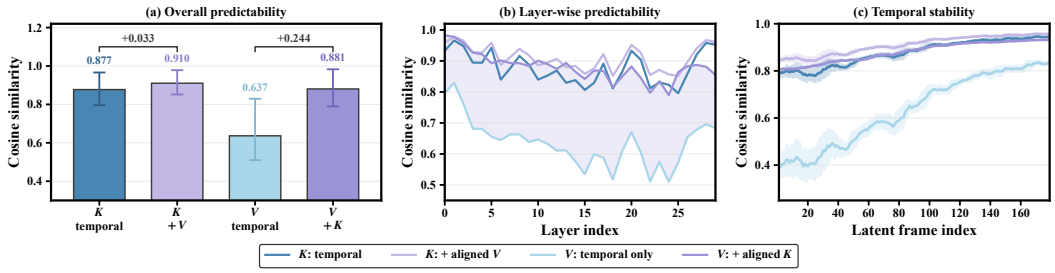


Figure 3: **Asymmetric predictability of video KV cache.** (a) Temporal history predicts keys well, while values benefit substantially from the aligned key. (b) This advantage holds across all layers. (c) The gain remains stable over long generation.

Following Sec. 3.2, TempoKV uses asymmetric predictive dependencies for K and V. Let $o_t^K \le h_K$ and $o_t^V \le h_V$ denote the available key and value history lengths, respectively. We use β^K and β^V to denote the bias terms of the K and V predictors. The key is predicted from its reconstructed history:

$$\widetilde{K}_{t,s} = \beta^K + \sum_{i=1}^{o_t^K} A_i^K \odot \widehat{K}_{t-i,s}, \tag{3}$$

where $\odot$ denotes element-wise multiplication. For the first frames after the anchor, fewer than h_K, we use a separate set of coefficients. After reconstructing the current key from its quantized prediction residual, the value is predicted from both $\widehat{K}_{t,s}$ and its reconstructed history. A dense projection $W_{K\to V} \in \mathbb{R}^{D\times D}$ maps the aligned key to the value:

$$\widetilde{V}_{t,s} = \beta^V + W_{K\to V}\widehat{K}_{t,s} + \sum_{j=1}^{o_t^V} A_j^V \odot \widehat{V}_{t-j,s}. \tag{4}$$

Anchor frames. TempoKV uses only the first latent frame as an anchor throughout the entire generation process. Unlike residual frames, the anchor is quantized without prediction. Its KV activations retain a larger dynamic range than the prediction residuals, making aggressive low-bit quantization more prone to large errors. Since anchor initializes subsequent reconstruction, preserving its accuracy is particularly important. We quantize anchor at a higher precision, $b_a = \max(4, b_K, b_V)$, where b_K and b_V denote the bit-widths of the K and V residuals.

Offline Calibration. Both predictors are fitted offline on BF16 keys and values recorded from a small set of calibration prompts. The current full-precision key or value serves as the regression target, while the predictor inputs are the corresponding caches after quantization and reconstruction:

$$\theta_X^\star = \arg\min_{\theta_X} \sum_{t,s} \left\| X_{t,s} - f_X\left(\mathcal{C}_{t,s}^X; \theta_X\right)\right\|_2^2 + \lambda_X \|\theta_X\|_2^2, \qquad X \in \{K, V\}, \tag{5}$$

where f_X is the predictor of Eq. 3 or 4, $\theta_K = \{A^K, \beta^K\}$ and $\theta_V = \{A^V, \beta^V, W_{K\to V}\}$ collect its coefficients, $\mathcal{C}_{t,s}^K$ is the reconstructed key history, and $\mathcal{C}_{t,s}^V$ the reconstructed aligned key together with the reconstructed value history. We first fit the key predictor and use the resulting reconstructed keys to fit the value predictor, following the asymmetric dependency used during inference. The recorded BF16 traces are shared across compression configurations. Different bit-width settings are obtained by applying the corresponding quantization and reconstruction to the same traces, without rerunning the model. Predictor fitting is performed entirely on these recorded traces, requiring neither backpropagation through the model nor retraining. Further details are provided in Appendix B.1.

4 Experiments

4.1 Settings

Models and baselines. We evaluate TempoKV on three autoregressive video generation models: Self-Forcing-Wan-1.3B (Huang et al., 2025a), Causal Forcing-Wan-1.3B (Zhu et al., 2026), and

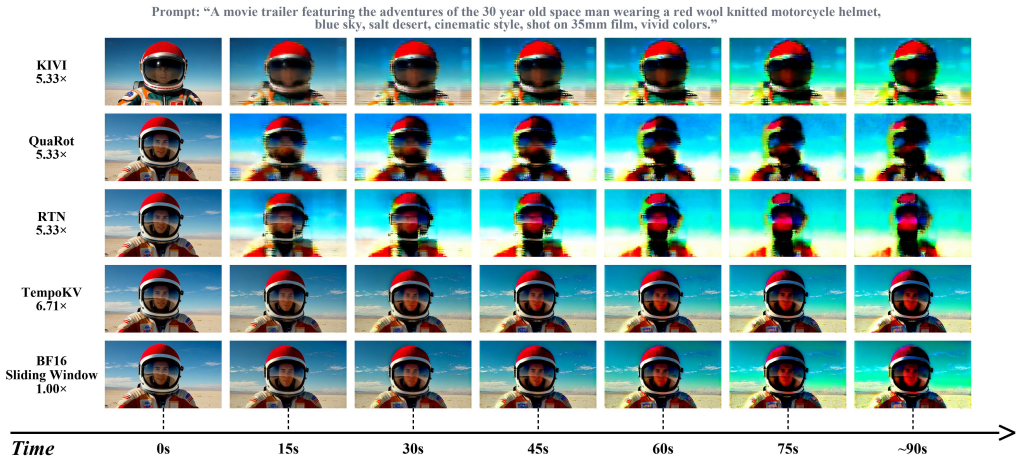


Figure 4: Qualitative comparison of long-horizon generation with Self-Forcing. While KIVI, QuaRot, and RTN exhibit progressively stronger visual degradation over time, TempoKV maintains greater temporal consistency, with results remaining close to the BF16 reference.

Table 1: Causal Forcing results on MovieGen at 832×480 and 717 frames. Best and second-best results within each precision setting are shown in **bold** and underline, respectively.

Method	Compression Ratio	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Background Consistency ↑	Subject Consistency ↑	Dynamic Degree ↑	Imaging Quality ↑	Aesthetic Quality ↑	Motion Smoothness ↑	Visual Quality ↑
BF16	1.00×	∞	1.00	0.00	0.00	95.91	95.81	58.25	61.18	53.46	97.08	76.95
INT2 KV Cache, 480p												
RTN	5.33×	10.32	0.42	0.58	143.05	95.76	95.05	15.05	42.05	41.22	96.73	64.31
KIVI	5.33×	10.57	0.33	0.66	168.80	95.29	95.09	12.43	42.75	43.03	97.43	64.34
QuaRot	5.33×	9.51	0.38	0.64	174.07	95.83	94.18	39.77	36.35	49.48	95.83	68.57
QVG	6.61×	16.21	0.70	0.22	14.24	95.99	95.78	54.62	58.58	52.42	**97.47**	75.81
QVG-Pro	4.34×	16.51	0.71	0.20	9.67	95.98	95.94	57.95	61.00	**53.34**	97.42	76.94
TempoKV (k2v2)	**6.71×**	17.23	0.76	0.17	9.15	96.01	95.80	**59.21**	60.84	53.28	97.11	77.04
TempoKV (k4v2)	4.73×	**22.85**	**0.79**	**0.13**	**6.34**	**96.03**	**95.96**	59.16	**61.12**	53.27	97.15	**77.11**
INT4 KV Cache, 480p												
RTN	3.20×	14.48	0.66	0.27	15.86	95.94	95.72	54.45	59.69	52.66	97.15	75.94
KIVI	3.20×	14.50	0.63	0.29	46.50	95.24	95.00	30.50	52.66	50.52	97.66	70.26
QuaRot	3.20×	14.61	0.67	0.26	14.36	95.90	95.85	56.52	60.08	52.55	97.29	76.37
QVG	3.08×	16.44	0.71	0.20	9.39	95.97	95.91	58.80	61.03	53.34	97.38	77.07
QVG-Pro	2.51×	16.38	0.71	0.20	9.20	95.98	95.96	58.19	61.16	53.30	97.47	77.01
TempoKV (k5v3)	**3.67×**	23.21	0.80	0.11	4.03	96.04	95.99	59.00	61.09	53.31	97.30	77.12
TempoKV (k6v4)	2.97×	**23.85**	**0.83**	**0.10**	**3.75**	**96.23**	**96.16**	**59.06**	**61.26**	**53.42**	**97.77**	**77.31**

LongCat-Video-13B (Meituan LongCat Team et al., 2025). We compare against three groups of baselines: (1)video-specific KV cache quantization, represented by QVG (Xi et al., 2026) and QVG-Pro, (2)general-purpose KV cache quantization, including RTN, KIVI (Liu et al., 2024), and QuaRot (Ashkboos et al., 2024) and (3) video-specific token pruning, represented by Forcing-KV (Ji et al., 2026). Detailed results compared with Forcing-KV are in the Appendix A.6.

Metrics and benchmark. We evaluate all video generation on the first 128 prompts from MovieGen (Polyak et al., 2024), using VBench-Long (Huang et al., 2024; 2025b) metrics to assess perceptual video quality. Visual Quality is the average of the six reported VBench dimensions. To quantify generation fidelity relative to uncompressed baselines, we report PSNR, SSIM, LPIPS, and FID. System efficiency is assessed across multiple dimensions, including the KV cache compression ratio, peak GPU memory footprint, and end-to-end latency.

4.2 Main Results

Quantitative results. Tables 1, 2 and 3 compare TempoKV with existing methods on three video generation models. Each precision setting includes two TempoKV configurations, where **`kNvM`** denotes N-bit keys and M-bit values. On Self-Forcing, TempoKV (**`k5v3`**) reaches $3.65\times$ compression at INT4, exceeding the $3.49\times$ of QVG, while improving PSNR from **18.86** to **24.17** and Visual Quality from **73.16** to **74.04**. TempoKV (**`k6v4`**) runs at $2.97\times$, close to the $3.01\times$ of QVG-Pro,

Table 2: Self-Forcing results on MovieGen at 832×480 and 717 frames. Best and second-best results within each precision setting are shown in **bold** and underline, respectively.

Method	Compression Ratio	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Background Consistency ↑	Subject Consistency ↑	Dynamic Degree ↑	Imaging Quality ↑	Aesthetic Quality ↑	Motion Smoothness ↑	Visual Quality ↑
BF16	1.00×	∞	1.00	0.00	0.00	96.36	97.88	42.39	70.39	39.77	98.61	74.23
INT2 KV Cache, 480p												
RTN	5.33×	12.03	0.48	0.57	227.40	95.83	96.71	12.32	43.36	38.15	98.04	64.06
KIVI	5.31×	14.11	0.53	0.48	150.31	95.82	97.50	12.79	51.63	38.64	98.32	65.78
QuaRot	5.33×	11.68	0.34	0.60	139.98	89.07	78.92	36.34	62.76	38.57	96.12	66.96
QVG	6.61×	18.34	0.70	0.18	15.17	95.25	97.05	39.40	66.48	39.69	98.49	72.73
QVG-Pro	4.65×	18.92	0.73	0.15	9.17	95.34	97.15	41.10	67.17	39.79	98.50	73.18
TempoKV (k2v2)	**6.71×**	19.85	0.78	0.13	10.04	95.35	97.04	**41.46**	67.81	39.79	98.48	73.32
TempoKV (k4v2)	4.73×	**23.17**	**0.85**	**0.07**	**5.23**	**95.83**	**97.89**	41.37	**70.50**	**39.90**	**98.73**	**74.04**
INT4 KV Cache, 480p												
RTN	3.20×	16.80	0.66	0.24	83.32	94.96	95.83	33.12	65.37	39.75	97.83	71.14
KIVI	3.19×	16.67	0.65	0.26	73.40	94.82	95.87	27.76	64.54	39.73	97.72	70.07
QuaRot	3.21×	16.81	0.66	0.24	47.11	94.80	96.73	36.07	66.68	39.74	97.73	71.96
QVG	3.49×	18.86	0.73	0.15	9.69	95.22	96.99	41.38	67.15	39.76	98.46	73.16
QVG-Pro	3.01×	19.21	0.74	0.14	9.30	95.33	97.12	41.34	66.96	39.80	98.50	73.18
TempoKV (k5v3)	**3.65×**	24.17	0.88	0.06	4.13	95.79	97.72	**41.43**	70.90	**39.89**	**98.53**	74.04
TempoKV (k6v4)	2.97×	**24.71**	**0.89**	**0.05**	**3.28**	**95.91**	**97.95**	41.36	**70.98**	**39.89**	**98.53**	**74.10**

Table 3: LongCat-Video-13B results on MovieGen at 832×480 and 717 frames. Best and second-best results within each precision setting are shown in **bold** and underline, respectively.

Method	Compression Ratio	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Background Consistency ↑	Subject Consistency ↑	Dynamic Degree ↑	Imaging Quality ↑	Aesthetic Quality ↑	Motion Smoothness ↑	Visual Quality ↑
BF16	1.00×	∞	1.00	0.00	0.00	95.70	97.77	49.33	72.45	40.09	98.53	75.65
INT2 KV Cache, 480p												
RTN	5.33×	14.21	0.57	0.36	96.48	95.73	97.55	30.20	56.86	38.46	97.60	69.40
KIVI	5.33×	15.33	0.63	0.34	62.02	94.96	94.39	30.67	64.07	38.84	96.63	69.93
QuaRot	5.33×	13.35	0.54	0.42	36.48	94.58	95.18	41.00	66.91	39.56	97.20	72.40
QVG	6.47×	15.75	0.66	0.25	24.29	95.61	97.66	47.33	69.54	39.95	98.52	74.77
QVG-Pro	4.21×	15.88	0.67	0.20	15.00	95.78	97.81	46.83	71.32	**40.10**	98.57	75.07
TempoKV (k2v2)	**6.66×**	16.57	0.71	0.15	12.40	**95.81**	97.66	48.33	71.62	40.09	**98.72**	75.37
TempoKV (k4v2)	4.72×	**17.39**	**0.75**	**0.12**	**7.13**	95.76	**97.85**	**49.50**	**72.32**	40.09	98.52	**75.67**
INT4 KV Cache, 480p												
RTN	3.20×	14.68	0.60	0.35	40.36	95.48	97.79	47.40	66.04	38.86	97.53	73.91
KIVI	3.20×	16.42	0.68	0.19	16.96	95.58	97.77	46.67	72.14	40.01	98.47	75.11
QuaRot	3.20×	16.64	0.71	0.16	15.37	**95.73**	97.73	46.83	72.17	40.11	98.48	75.18
QVG	3.59×	16.82	0.73	0.14	14.63	95.63	97.79	48.50	71.86	40.07	98.49	75.39
QVG-Pro	2.75×	17.34	0.74	0.12	12.51	95.60	97.78	47.56	72.35	**40.23**	**98.63**	75.36
TempoKV (k5v3)	**3.62×**	18.01	0.78	0.10	6.51	95.59	97.83	48.32	72.31	40.09	98.51	75.44
TempoKV (k6v4)	2.97×	**18.42**	**0.81**	**0.09**	**6.24**	95.67	**97.87**	**48.74**	72.38	40.11	98.53	**75.54**

and improves PSNR from 19.21 to **24.71** and FID from 9.30 to **3.28**. At INT2, TempoKV (**k2v2**) achieves a higher compression ratio (**6.71× vs.** 6.61×), while improving PSNR from **18.34** to **19.85**. TempoKV (**k4v2**) runs at 4.73×, close to the 4.65× of QVG-Pro, and improves PSNR from 18.92 to **23.17** and SSIM from 0.73 to **0.85**. Consistent gains on Causal Forcing and LongCat-Video-13B further show that these improvements are not specific to a single backbone or generation setting. As shown in Fig. 4, TempoKV maintains high visual fidelity and temporal consistency over long-horizon generation, while existing low-bit baselines exhibit severe degradation.

Quality–efficiency tradeoff. Fig. 5 reports the memory and latency cost of KV cache compression on Self-Forcing. At INT4, TempoKV reduces the KV cache from 48.18 GB to **13.2 GB** and peak memory usage falls from 68.27 GB to **35.59 GB**. Under the more aggressive INT2 setting, peak memory usage further decreases to **30.63 GB**, while LPIPS remains at **0.13**. Notably, this exceeds the INT4 quality of QVG while using 6.1 GB less peak memory. Decoding the compressed cache during attention adds latency. On our hardware, TempoKV adds only **11.9%** generation overhead to the BF16 latency of 337.9 s, against 15.5% for QVG, 23.5% to 25.2% for RTN and KIVI, and 103.9% for QuaRot. This overhead changes by less than 1 s from INT2 to INT4, whereas that of QVG grows by 9.9 s. Together, these results show that TempoKV preserves generation quality while reducing both KV storage and runtime memory, with minimal overhead.

4.3 Long-Horizon Generation

We further evaluate long-horizon generation on Self-Forcing by extending the rollout to 1401 frames, corresponding to approximately 90 seconds. With full-history attention, the BF16 baseline reaches only about 815 frames on our hardware before running out of memory, whereas TempoKV can complete the full 1401-frame rollout while retaining the entire KV cache. For the quality comparison

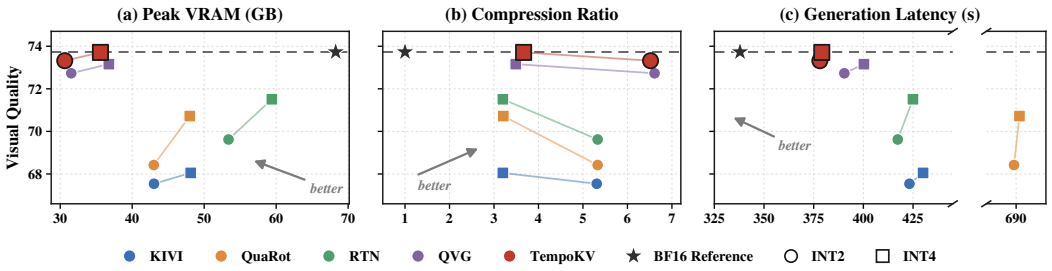


Figure 5: **Quality–efficiency tradeoffs.** Visual Quality is plotted against **(a)** peak VRAM, **(b)** KV-cache compression ratio, and **(c)** generation latency. TempoKV remains close to BF16 quality while providing the best balance among compression, runtime memory, and generation latency.

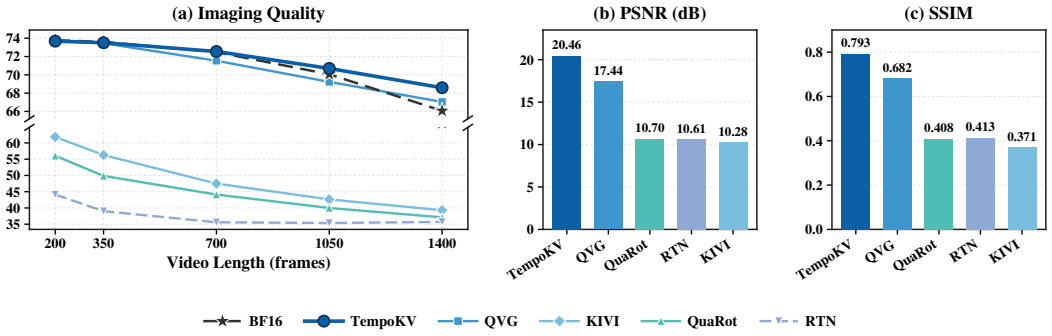


Figure 6: **Quality preservation over long video generation.** (a) Imaging Quality across increasing video lengths, where TempoKV remains close to the BF16 baseline and consistently outperforms existing quantization methods. (b–c) PSNR and SSIM comparisons further show that TempoKV better preserves quality under aggressive KV compression.

in Fig. 6, we use the same fixed attention window of 180 latent frames for all methods, including BF16, to avoid differences caused by attention context. As shown in Fig. 6, TempoKV remains closer to the BF16 reference as the generation horizon increases. Compared with QVG, it improves Imaging Quality from 71.54 to **72.55** at 700 frames, from 69.23 to **70.70** at 1050 frames, and from 67.04 to **68.59** at 1400 frames. Averaged across all lengths, TempoKV achieves the highest Imaging Quality among compressed methods, together with **20.46** PSNR and **0.793** SSIM.

4.4 Ablation Studies

Effect of the cross predictor. Fig. 7(a–b) compares the normalized reconstruction error at different effective bit-widths. For K, temporal prediction lowers the error at every bit-width, by a factor of $3.27\times$ at INT2. For V, temporal prediction alone reduces the error by $1.42\times$ while adding the cross predictor raises the reduction to $2.19\times$. The lower error propagates to video quality. Fig. 7(c) reports the VBench-Long score difference between the cross predictor and the temporal-only predictor. Visual Quality improves from 68.77 to 74.10. The gain is concentrated in Imaging Quality ($+13.44$), followed by Aesthetic Quality ($+1.49$). Background Consistency, Subject Consistency and Motion Smoothness change by less than 0.8. These three dimensions already exceed 95 without the cross predictor and are close to saturation. Consistency scores can also be raised by generating more static video. However, cross predictor's Dynamic Degree increases by 2.13, indicating that consistency scores were not maintained at the expense of reducing the magnitude of motion.

Mixed KV bit-widths. Sec. 3.2 argues that K should be given higher precision than V. Table 4 validates this design under different allocation schemes. At $4.73\times$, **`k4v2`** substantially outperforms the reversed **`k2v4`** allocation, improving PSNR from 19.83 to **23.17** and reducing FID from 10.28 to **5.23**. The same pattern appears at $5.55\times$, where **`k3v2`** clearly outperforms **`k2v3`**. The two directions of Table 4 are also asymmetric. Holding K at 2 bits and raising V from 2 to 4 leaves video quality virtually unchanged, while the compression ratio drops from $6.71\times$ to $4.73\times$. Holding V at 2 bits and

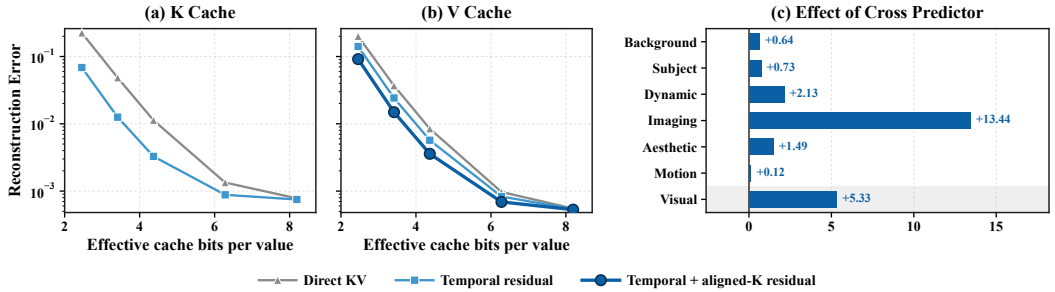


Figure 7: **Predictor ablation.** (a–b) Reconstruction error of the K and V caches on different effective bits. (c) Per-dimension difference between the cross predictor and the temporal-only predictor.

Table 4: Ablation on mixed-bit K/V precision allocation. Representative settings used in the main experiments are highlighted.

Setting	Compression Ratio	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Visual Quality ↑
k6v4	2.97×	24.71	0.89	0.05	3.28	74.10
k5v4	3.28×	24.43	0.88	0.06	4.03	74.06
k5v3	3.65×	24.17	0.88	0.06	4.13	74.04
k3v5	3.65×	22.60	0.84	0.07	6.36	73.71
k4v4	3.66×	23.55	0.86	0.06	4.99	73.96
k4v3	4.13×	23.21	0.85	0.07	5.20	73.95
k4v2	4.73×	23.17	0.85	0.07	5.23	74.04
k3v3	4.73×	22.47	0.84	0.07	6.43	73.92
k2v4	4.73×	19.83	0.78	0.13	10.28	73.58
k3v2	5.55×	22.22	0.83	0.08	6.94	73.75
k2v3	5.55×	19.77	0.78	0.13	10.36	73.64
k2v2	6.71×	19.85	0.78	0.13	10.04	73.32
k4v1	5.55×	12.31	0.45	0.61	139.63	66.09
k3v1	6.71×	12.33	0.45	0.62	141.82	65.71

Table 5: Ablation on the temporal history length for K and V prediction.

Setting	h_K/h_V	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	Visual Quality ↑
Key history (V: w/ aligned-K)						
	0 / 1	16.45	0.604	0.321	73.98	68.02
	1 / 1	19.73	0.776	0.134	10.42	73.06
default	**2 / 1**	**19.85**	**0.783**	**0.130**	**10.04**	**73.32**
	3 / 1	19.81	0.779	0.131	10.11	73.21
	4 / 1	19.76	0.777	0.133	10.28	73.24
	5 / 1	19.79	0.778	0.132	10.19	73.29
Value history (V: w/ aligned-K)						
	2 / 0	19.73	0.775	0.135	10.51	72.93
default	**2 / 1**	**19.85**	**0.783**	**0.130**	**10.04**	**73.32**
	2 / 2	19.80	0.778	0.132	10.16	73.29
	2 / 3	19.77	0.777	0.133	10.25	73.15
	2 / 4	19.74	0.776	0.134	10.37	73.18
	2 / 5	19.78	0.778	0.132	10.21	72.99

raising K from 2 to 4 improves PSNR by 3.3, from 19.85 to **23.17**. This further supports allocating additional precision to K rather than distributing the budget uniformly between K and V. However, values cannot be assigned excessively low precision. With 1-bit values, the quality collapses sharply.

Temporal history length. Table 5 studies the temporal history used by the K and V predictors and all settings in this ablation use **k2v2**. We first vary h_K while fixing h_V=1 and retaining aligned-K conditioning for V. Without temporal K prediction (h_K=0), PSNR drops from 19.85 to 16.45 and FID increases from 10.04 to 73.98. Using one previous K frame recovers most of this loss, while h_K=2 gives the best overall performance. Extending the history to three or more frames brings no consistent gain across the evaluation metrics. We therefore set h_K=2 by default. We then vary h_V while fixing h_K=2 and keeping aligned-K conditioning. The results are much less sensitive to the V history length. Using one previous V frame gives the best overall performance, while longer histories produce only minor fluctuations and no consistent improvement. This behavior is consistent with the aligned K providing most of the useful information for V prediction, with temporal V history serving as a short complementary context. We therefore use h_V=1 throughout the experiments.

5 Conclusion

In this work, we introduce **TempoKV**, a training-free predictive KV cache quantization framework for autoregressive video generation. TempoKV exploits asymmetric dependencies between keys and values. Keys are predictable from their own history, while values depend mainly on the aligned current key. It predicts both accordingly, quantizes only the residuals, and assigns keys higher precision. Extensive experiments demonstrate that TempoKV substantially improves low-bit KV reconstruction and achieves up to 6.71× cache compression, while delivering the best generation quality and compression–quality trade-off compared with existing KV cache quantization work.

AI USE STATEMENT

AI tools were used to assist with code review, LaTeX formatting, and manuscript polishing. Specifically, large language models were employed for document structure refinement and language editing to improve clarity and readability. They were not used to generate scientific claims, experimental results, or conclusions. All AI-assisted suggestions were carefully reviewed by the authors, who take full responsibility for the final manuscript and the correctness of all reported results.

REPRODUCIBILITY STATEMENT

To ensure full reproducibility of our results, we document the algorithmic formulation, implementation details, and experimental configurations of TempoKV throughout the main paper and appendix. Our code is provided as supplementary material and will be released publicly upon publication.

Appendix B describes the complete experimental protocol, including model and baseline configurations, calibration data collection, predictor fitting, and quantization settings. It further details the pre-RoPE key representation, the anchor frame and runtime implementation. The reported compression ratios account for the physical storage of quantized KV tensors together with all required metadata, including scales, zero points, anchors, and predictor parameters. Evaluation settings for memory, latency, reconstruction fidelity, and video generation quality are also specified to facilitate reproduction of all reported results.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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.

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, 2026.

Haojie Duanmu, Zhihang Yuan, Xiuhong Li, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. SKVQ: Sliding-window key and value cache quantization for large language models. *arXiv preprint arXiv:2405.06219*, 2024.

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.

Yefei He, Luoming Zhang, Weijia Wu, Jing Liu, Hong Zhou, and Bohan Zhuang. ZipCache: Accurate and efficient KV cache quantization with salient token identification. *arXiv preprint arXiv:2405.14256*, 2024.

Roberto Henschel, Levon Khachatryan, Daniil Hayrapetyan, Hayk Poghosyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. StreamingT2V: Consistent, dynamic, and extendable long video generation from text. *arXiv preprint arXiv:2403.14773*, 2024.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, volume 38, pp. 167283–167308. Curran Associates, Inc., 2025a. doi: 10.52202/085713-5576.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

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

Yicheng Ji, Zhizhou Zhong, Jun Zhang, Qin Yang, XiTai Jin, Ying Qin, Wenhan Luo, Shuiyang Mao, Wei Liu, and Huan Li. Forcing-KV: Hybrid KV cache compression for efficient autoregressive video diffusion models. *arXiv preprint arXiv:2605.09681*, 2026.

Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. GEAR: An efficient error reduction framework for KV cache compression in LLM inference. In *Proceedings of the 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pp. 305–321, 2024.

Junyan Li, Yang Zhang, Muhammad Yusuf Hassan, Talha Chafekar, Tianle Cai, Zhile Ren, Pengsheng Guo, Foroozan Karimzadeh, Colorado Reed, Chong Wang, and Chuang Gan. CommVQ: Commutative vector quantization for KV cache compression. *arXiv preprint arXiv:2506.18879*, 2025.

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.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970. Curran Associates, Inc., 2024. doi: 10.52202/079017-0722.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling Forcing: Autoregressive long video diffusion in real time. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32332–32344, 2024.

Jiayi Luo, Qiyan Liu, Tengyang Wang, JunHao Liu, Jiayu Chen, Cong Wang, Hanxin Zhu, Chen Gao, Xiaobin Hu, Qingyun Sun, and Zhibo Chen. Future Forcing: Future-aware training-free KV cache policy for autoregressive video generation. *arXiv preprint arXiv:2605.30083*, 2026.

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

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.

Zunhai Su, Zhe Chen, Wang Shen, Hanyu Wei, Linge Li, Huangqi Yu, and Kehong Yuan. RotateKV: Accurate and robust 2-bit KV cache quantization for LLMs via outlier-aware adaptive rotations. *arXiv preprint arXiv:2501.16383*, 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.

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

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, Linus, Patrol, Peizhen Zhang, Peng Chen, Penghao Zhao, Qi Tian, Songtao Liu, Weijie Kong, Weiyan Wang, Xiao He, Xin Li, Xinchi Deng, Xuefei Zhe, Yang Li, Yanxin Long, Yuanbo Peng, Yue Wu, Yuhong Liu, Zhenyu Wang, Zuozhuo Dai, Bo Peng, Coopers Li, Gu Gong, Guojian Xiao, Jiahe Tian, Jiaxin Lin, Jie Liu, Jihong Zhang, Jiesong Lian, Kaihang Pan, Lei Wang, Lin Niu, Mingtao Chen, Mingyang Chen, Mingzhe Zheng, Miles Yang, Qiangqiang Hu, Qi Yang, Qiuyong Xiao, Runzhou Wu, Ryan Xu, Rui Yuan, Shanshan Sang, Shisheng Huang, Siruis Gong, Shuo Huang, Weiting Guo, Xiang Yuan, Xiaojia Chen, Xiawei Hu, Wenzhi Sun, Xiele Wu, Xianshun Ren, Xiaoyan Yuan, Xiaoyue Mi, Yepeng Zhang, Yifu Sun, Yiting Lu, Yitong Li, You Huang, Yu Tang, Yixuan Li, Yuhang Deng, Yuan Zhou, Zhichao Hu, Zhiguang Liu, Zhihe Yang, Zilin Yang, Zhenzhi Lu, Zixiang Zhou, and Zhao Zhong. Hunyuanvideo 1.5 technical report, 2025. URL https://arxiv.org/abs/2511.18870.

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. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

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.

Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. WorldMem: Long-term consistent world simulation with memory. In *Advances in Neural Information Processing Systems*, 2025.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time interactive long video generation. In *International Conference on Learning Representations*, volume 2026, pp. 90037–90059, 2026.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974, June 2025.

Hailin Zhang, Xiaodong Ji, Yilin Chen, Fangcheng Fu, Xupeng Miao, Xiaonan Nie, Weipeng Chen, and Bin Cui. PQCache: Product quantization-based KVCache for long context LLM inference. *arXiv preprint arXiv:2407.12820*, 2024.

Lvmin Zhang and Maneesh Agrawala. Packing input frame context in next-frame prediction models for video generation. *arXiv preprint arXiv:2504.12626*, 2025.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34661–34710. Curran Associates, Inc., 2023. doi: 10.52202/075280-1506.

Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

A Additional Experiments

A.1 Predictability on LongCat-Video-13B

Fig. 8 repeats the predictability analysis on LongCat-Video-13B using cosine similarity between the predicted and original caches. The same K–V asymmetry remains clearly visible on this larger backbone: adding the aligned V improves K prediction from 0.876 to 0.918 (**+0.042**), whereas adding the aligned K improves V prediction from 0.455 to 0.609 (**+0.155**), yielding a much larger gain.

Predictability varies substantially across layers but remains stable over time. With temporal prediction alone, the layer-wise cosine similarity ranges from 0.827 to 0.990 for K and from 0.263 to nearly 1.0 for V. Adding the aligned cache preserves this large layer-wise variation. In contrast, the variation across latent frames is much smaller: *K + aligned V* remains between 0.909 and 0.924, while *V + aligned K* stays between 0.581 and 0.625. Across all four predictors, the layer-wise variation is approximately $3\times$–$13\times$ larger than the temporal variation. This layer-dependent but temporally stable behavior supports the design of TempoKV: we fit a separate predictor for each layer and head and reuse it across all latent frames. These results show that the asymmetric predictive dependencies identified in Sec. 3.2 persist across different autoregressive video models.

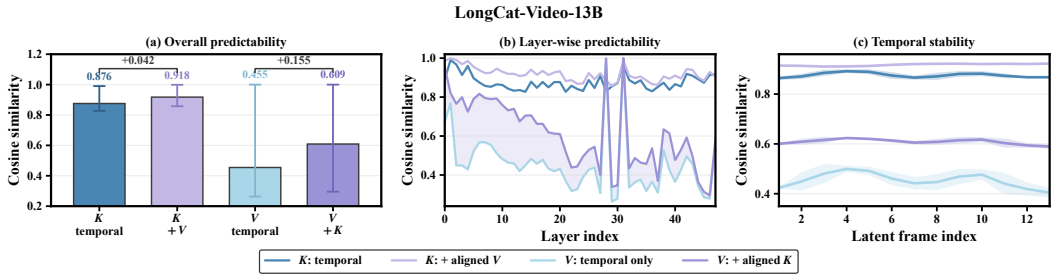


Figure 8: Predictability of the key and value caches on LongCat-Video-13B, measured by cosine similarity between predicted and original caches. (a) Mean over all layers, with and without the aligned cache as input. (b) Per layer. (c) Per latent frame.

A.2 Cross-prediction direction

We further test whether the cross prediction between K and V depends on its direction by reversing the predictor from K→V to V→K, with the rest of the compression pipeline unchanged. As shown in Table 6, K→V achieves better reconstruction quality across all bit allocations. The gap is particularly large at low precision. Under **`k4v2`**, reversing the direction reduces PSNR from **23.17** to **17.71** and increases FID from **5.23** to **73.80**. The difference remains at higher precision, e.g. **`k5v3`** and **`k6v4`**. These results indicate that the aligned key is a better predictor of values, which supports the K→V direction and the asymmetric bit allocation in TempoKV.

Table 6: **Cross predictor is directional.** Predicting V from the reconstructed K outperforms the reverse at every bit allocation and on every metric. The better direction is shown in **bold**.

Bits (K/V)	Direction	PSNR↑	SSIM↑	LPIPS↓	FID↓
2 / 2	K→V	**19.85**	**0.7822**	**0.1303**	**10.04**
	V→K	17.66	0.6394	0.2575	75.73
4 / 2	K→V	**23.17**	**0.8539**	**0.0729**	**5.23**
	V→K	17.71	0.6426	0.2562	73.80
5 / 3	K→V	**24.17**	**0.8815**	**0.0597**	**4.13**
	V→K	20.74	0.7886	0.1139	17.24
6 / 4	K→V	**24.71**	**0.8947**	**0.0526**	**3.28**
	V→K	22.60	0.8436	0.0763	7.62

A.3 ANCHOR PRECISION

Table 7 studies the anchor precision under **`k2v2`**. Anchors below 4 bits degrade reconstruction quality. FID rises from 10.04 with 4-bit anchors to 23.31 and 65.88 with 3-bit and 2-bit anchors, respectively. Above 4 bits, both reconstruction and VBench scores remain largely unchanged, while the compression ratio decreases as more bits are allocated to anchors. Therefore, we set $b_a = \max(4, b_K, b_V)$, which keeps anchors at least 4-bit while providing a sufficiently accurate reference for subsequent prediction without unnecessary storage overhead.

Table 7: **Ablation on anchor precision.** Anchor precisions below 4 bits degrade reconstruction quality, whereas increasing the precision beyond 4 bits provides little additional benefit.

Anchor bits	Comp.↑	PSNR↑	SSIM↑	LPIPS↓	FID↓	Subject Consistency↑	Motion Smoothness↑	Aesthetic Quality↑	Imaging Quality↑
2	6.739×	16.83	0.6095	0.2742	65.88	96.68	98.28	39.30	62.61
3	6.723×	19.53	0.7304	0.1471	23.31	96.96	98.43	39.69	67.35
4	**6.707×**	**19.85**	**0.7832**	**0.1303**	**10.04**	**97.04**	**98.48**	**39.79**	**67.81**
5	6.692×	19.79	0.7858	0.1243	10.09	97.04	98.49	39.83	68.02
6	6.676×	19.28	0.7818	0.1244	10.12	97.04	98.50	39.79	67.87
7	6.661×	19.81	0.7841	0.1242	10.01	97.06	98.49	39.78	67.73
8	6.646×	19.67	0.7808	0.1270	10.04	97.08	98.49	39.84	68.13
16 (BF16)	6.529×	19.57	0.7797	0.1274	10.07	97.04	98.50	39.83	67.84

A.4 HIGHER-RESOLUTION GENERATION

To evaluate the effectiveness of TempoKV at higher resolutions, we conduct experiments on LongCat-Video-13B at 720p under INT4 quantization. As shown in Table 8, TempoKV **`k5v3`** achieves the highest compression ratio of $3.65\times$ while also providing the best reconstruction fidelity, with a PSNR of 17.31, SSIM of 0.77, LPIPS of 0.12, and FID of 11.20. It also attains the highest Visual Quality score of 75.74 and the best Subject Consistency (97.76), Dynamic Degree (47.33), Imaging Quality (74.64), and Motion Smoothness (98.37) among the compression methods. These results show that the predictive structure exploited by TempoKV remains effective when the number of spatial tokens increases, preserving its compression advantage and generation quality at higher resolutions.

Table 8: Quality comparison on LongCat-Video at 720p under INT4 quantization. Best and second-best results are shown in **bold** and underline, respectively.

Method	Compression	PSNR↑	SSIM↑	LPIPS↓	FID↓	Background Consistency↑	Subject Consistency↑	Dynamic Degree↑	Imaging Quality↑	Aesthetic Quality↑	Motion Smoothness↑	Visual Quality↑
BF16	1.00×	∞	1.00	0.00	0.00	95.68	97.71	47.67	74.71	41.60	98.29	76.28
RTN	3.20×	15.65	0.72	0.17	15.23	95.28	97.61	43.12	74.33	40.59	98.32	74.87
KIVI	3.20×	13.13	0.61	0.32	59.83	95.60	94.81	26.67	65.02	39.14	96.39	69.61
QuaRot	3.20×	16.48	0.74	0.15	13.56	95.73	97.72	45.67	74.32	40.58	98.33	75.40
QVG	3.59×	15.43	0.66	0.23	18.94	95.58	97.51	47.00	74.32	40.60	98.18	75.53
QVG-Pro	2.75×	15.32	0.68	0.20	19.60	**95.82**	97.52	46.00	74.57	**40.63**	98.28	75.47
TempoKV (`k5v3`)	**3.65×**	**17.31**	**0.77**	**0.12**	**11.20**	95.73	**97.76**	**47.33**	**74.64**	40.62	**98.37**	**75.74**

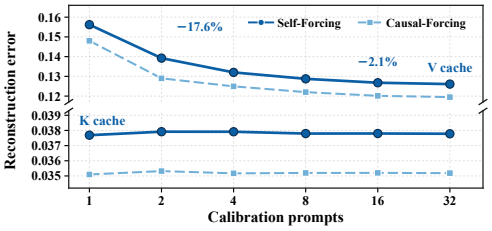
A.5 CALIBRATION ANALYSIS

Fig. 9 studies how many calibration prompts the predictors need and how sensitive they are to the calibration distribution. We build two disjoint calibration sets with different content. The *action* set focuses on motion and interactions, and the *scene* set focuses on appearance, composition, and environment. No calibration prompt overlaps with the MovieGen prompts used for evaluation.

A few calibration prompts are sufficient. We first vary the number of calibration prompts from 1 to 32. The K predictor is particularly data-efficient: a single prompt already achieves a reconstruction error within 0.3% of that obtained with 32 prompts. The cross predictor benefits more from additional calibration data, with its error decreasing by 17.6% from 1 to 8 prompts. Beyond eight prompts, however, the improvement quickly saturates, with only a further 2.1% reduction from 8 to 32 prompts. We use 8 calibration prompts by default to collect the BF16 traces for predictor fitting. These results show that a small calibration set is sufficient to obtain stable predictors.

Robust to calibration-set distribution. We next calibrate the predictors separately on the action and scene sets and evaluate each predictor on both distributions. Using a mismatched calibration

set changes the reconstruction error by at most 0.77% across K and V. Moreover, the change is not consistently negative: cross-distribution calibration slightly improves some settings while slightly degrading others. The predictors are therefore largely insensitive to the content distribution of the calibration set. Together with the small calibration requirement above, this makes TempoKV lightweight to calibrate and removes the need for a large or carefully matched calibration corpus.



(a) Few calibration prompts are sufficient

Calib. \ Eval	Action		Scene	
	K	V	K	V
Action	**0.03734**	**0.12644**	0.03828	0.12660
Scene	0.03730	0.12742	**0.03823**	**0.12744**
Δ vs. matched	-0.11%	$+0.77\%$	$+0.11\%$	-0.66%

(b) Robust to calibration-set distribution

Figure 9: **Calibration of TempoKV. Left:** A small calibration set is sufficient for TempoKV. Reconstruction error drops rapidly with the first few calibration prompts and nearly saturates thereafter on both Self-Forcing and Causal Forcing, with little additional gain beyond a handful of prompts. **Right:** TempoKV is robust to calibration-set distribution. Switching between action and scene prompts changes reconstruction error by at most 0.77% relative to the matched setting.

A.6 Comparison with token pruning

We compare TempoKV with Forcing-KV (Ji et al., 2026), which reduces KV cache memory by pruning cached tokens. Table 9 reports generation quality. On Causal Forcing, Forcing-KV reaches a PSNR of 9.32 and an FID of 40.07, while TempoKV in its lowest-bit configuration reaches 17.23 and 9.15. The gap is larger on Self-Forcing, where Forcing-KV reaches 9.44 and 83.86 and TempoKV reaches 19.85 and 10.04. Higher-bit configurations of TempoKV improve fidelity further, and all TempoKV configurations score higher than Forcing-KV in Visual Quality. Compared with token pruning, quantization preserves more information from each cached token and yields higher fidelity. Token pruning, in turn, reduces attention computation and can accelerate generation.

Table 9: **Comparison with token pruning.** Across both Causal Forcing and Self-Forcing, TempoKV improves fidelity in PSNR, SSIM, LPIPS, and FID, while also achieving higher overall Visual Quality.

Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	Visual Quality↑
Causal Forcing, 480p					
Forcing-KV	9.32	0.38	0.58	40.07	76.70
TempoKV (k2v2)	17.23	0.76	0.17	9.15	77.04
TempoKV (k4v2)	22.85	0.79	0.13	6.34	77.11
TempoKV (k5v3)	**23.21**	**0.80**	**0.11**	**4.03**	**77.12**
Self-Forcing, 480p					
Forcing-KV	9.44	0.25	0.71	83.86	72.79
TempoKV (k2v2)	19.85	0.78	0.13	10.04	73.32
TempoKV (k4v2)	23.17	0.85	0.07	5.23	74.04
TempoKV (k5v3)	**24.17**	**0.88**	**0.06**	**4.13**	**74.04**

B Implementation Details

Baseline configurations. RTN applies per-token asymmetric round-to-nearest quantization to both keys and values. KIVI (Liu et al., 2024), QuaRot (Ashkboos et al., 2024), and Forcing-KV (Ji et al., 2026) follow their released configurations. For QVG (Xi et al., 2026), we use the final configuration of the official implementation. QVG uses one stage of progressive residual quantization ($N_S = 1$) with group size $G = 64$, while QVG-Pro uses $N_S = 4$ and $G = 16$. Both use $N_C = 256$ k-means centroids. We implement TempoKV with Triton kernels and benchmark on NVIDIA A100 GPUs

(80GB) with CUDA 12.8. Since A100 GPUs do not provide native FP8 E4M3 support, we store the per-group scales in BF16 for both QVG and TempoKV. TempoKV uses a group size of $G = 64$ channels, with one BF16 scale and one UINT8 zero point stored for each group.

Frame convention. Throughout the paper, *frame* refers to a decoded video frame, while *latent frame* refers to one temporal position in the VAE latent space. The evaluated models use a temporal downsampling factor of 4, such that L latent frames correspond to $4L-3$ video frames, or equivalently, F video frames correspond to $(F + 3)/4$ latent frames for valid generation lengths. For example, 717 and 1401 video frames correspond to 180 and 351 latent frames, respectively.

Pre-RoPE key representation. TempoKV predicts and quantizes normalized keys before RoPE. Since RoPE applies position-dependent rotations, operating on post-RoPE keys would entangle positional variation with the temporal dynamics modeled by the key predictor. We therefore reconstruct keys in the pre-RoPE space and apply RoPE at their original absolute positions only when they are used in attention. Values are stored directly, as RoPE is not applied to them. For all comparison methods, we preserve the key representation used in their official implementations, including whether caching is performed before or after RoPE.

B.1 Offline Calibration

Trace collection. For each backbone, we collect clean activation traces from the BF16 model using 8 calibration prompts. Keys are recorded after normalization and before RoPE, and values are sampled at the same spatial positions. We retain all attention layers and heads, while uniformly sampling eight spatial tokens from each latent frame. For Self-Forcing and Causal Forcing, this corresponds to eight out of 1,560 tokens per frame. The sampled K/V traces require approximately 1.05 GiB for Self-Forcing. Trace collection is performed once per backbone, and the same traces are reused when changing the residual precision, history length, or anchor precision, without an additional model rollout.

Reconstruction-aware fitting. Predictor fitting follows the same reconstruction path used at inference. The original BF16 caches are retained as regression targets, while every predictor input is obtained from the reconstructed cache under the target quantization configuration. We first fit temporal K predictor on the reconstructed K history, then reconstruct K with the fitted predictor and fit the cross predictor using the reconstructed aligned K and reconstructed V history. Reconstruction and ridge regression are alternated for a small number of iterations until the coefficients stabilize. The entire fitting procedure is lightweight, taking only about 1.5 s for each compression configuration.

Predictor parameters. TempoKV fits separate K and V predictors for each attention layer and head and shares them across spatial positions. The K predictor uses element-wise temporal coefficients over its history ($h_K = 2$). The V predictor uses a dense aligned $K \to V$ projection together with element-wise temporal coefficients ($h_V = 1$), so it accounts for most of the predictor parameters. Self-Forcing and Causal Forcing share the Wan2.1-T2V-1.3B backbone (30 layers, 12 heads, and head dimension 128). Their K and V predictors contain 0.28M and 5.99M parameters, respectively, or 6.27M in total and 25.1 MB in FP32. LongCat-Video-13B has 48 layers, 32 heads, and head dimension 128. Its K and V predictors contain 1.18M and 25.56M parameters, respectively, or 26.74M in total and 107 MB in FP32.

B.2 Triton Kernel Optimizations

TempoKV reconstructs the compressed KV cache before attention consumes it, so the decoder lies on the critical path of generation. We implement this path with custom Triton kernels, and no intermediate tensor is materialized between dequantization and prediction. Each launch decodes several consecutive latent frames and fuses low-bit unpacking, dequantization, predictor evaluation, residual addition, and BF16 rounding. For the value cache, the cross $K \to V$ projection is computed inside the decoding kernel. Each tile multiplies the reconstructed keys by the per-head projection held in registers, so the projected keys are never written to memory. The value kernel reads the reconstructed keys through a strided view, so no contiguous copy of the key cache is needed. Reconstructed keys and values are written directly in the layout expected by the attention backend,

which removes a separate layout conversion. Predictor parameters stay resident on the GPU, and the compiled kernels are reused throughout generation. These optimizations change only the execution path. They do not change the predictor, the quantization rule, or the reconstruction precision used in the reported experiments.

C Visual Quality Comparisons

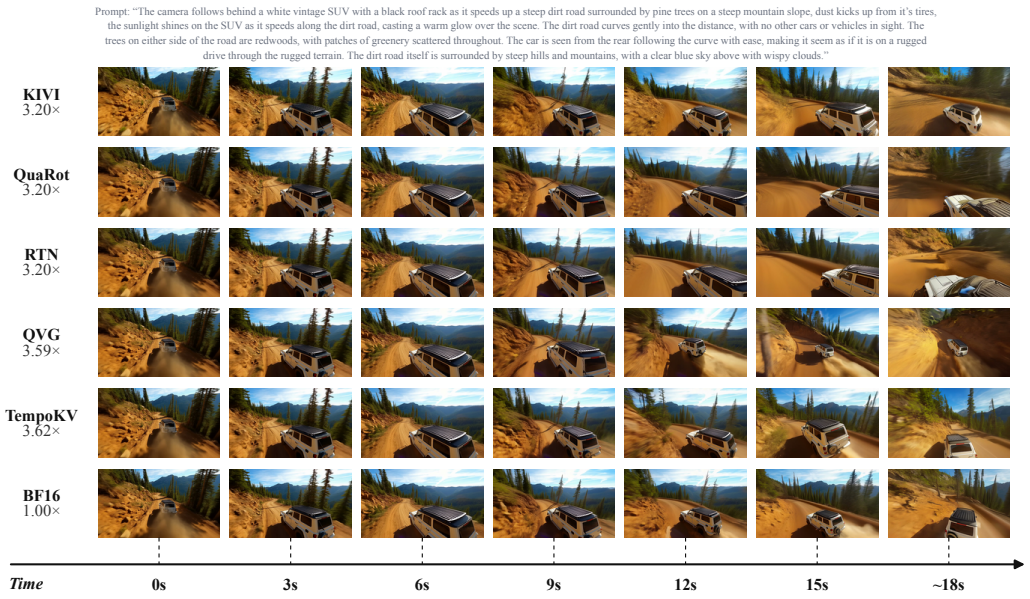


Figure 10: Qualitative comparison on LongCat-Video-13B at INT4. TempoKV maintains superior temporal coherence and visual fidelity at a higher compression ratio.



Figure 11: Qualitative comparison on Causal Forcing at INT2. TempoKV maintains superior temporal coherence and visual fidelity at a higher compression ratio.