TRACEGATE: ROUTING-ACTION HISTORY FOR BLOCK REUSE IN VIDEO DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Video diffusion transformers repeatedly evaluate costly block stacks and attention routes. Block caching skips stack evaluations; sparse attention reduces work within executed stacks. TraceGate connects these decisions: a planner reuses, repairs, or refreshes query and key clusters on executed calls, and its historical actions inform later residual reuse at the block level. The outer gate combines this delayed feedback with conditioning drift and residual magnitude history; skipped calls do not compute their own attention. A product quantization shortlist reduces planner candidate scoring. Settings focused on efficiency achieve 3.36×, 2.45×, and 3.02× overall speedups on Wan 2.1, Wan 2.2, and HunyuanVideo, respectively, with tradeoffs in task quality. On the Wan 2.1 calibration cohort of 108 videos, the setting with feedback has 0.13 dB higher mean PSNR and 0.003 higher mean SSIM than a retuned setting without feedback; both speedups round to 3.30× and LPIPS is equal at reported precision. This comparison is exploratory because the cohort also informed calibration. Substituting a stale map on 72 selected events tests propagation of inner routing error, not the outer execution versus reuse action.

1 INTRODUCTION

Video diffusion transformers produce high quality videos by repeatedly processing long spatiotemporal token sequences (Peebles & Xie, 2023; Kong et al., 2024; Wan Team, 2025a). Inference therefore presents two distinct opportunities for reuse: avoiding a redundant block stack evaluation and reducing the attention computation within an evaluation that is still needed. Their interaction matters because changing which steps execute also changes the age and distribution of the attention states available for reuse.

Existing methods provide effective building blocks for both opportunities. TeaCache and MagCache use input or residual statistics to select reuse intervals, and BWCache adapts reuse at the block level (Liu et al., 2025; Ma et al., 2025; Cui et al., 2026); sparse attention methods identify a subset of query and key interactions (Xi et al., 2025; Yang et al., 2025; Tan et al., 2026). Temporal state is useful inside sparse attention as well: SVG2 initializes clustering from cached centroids, and HEART uses query and key drift to control mask refresh (Yang et al., 2025; Zheng et al., 2026). These mechanisms motivate our hypothesis that historical state from an inner sparse attention planner can improve a later block reuse decision.

We address this question with TraceGate. Its outer controller either evaluates the DiT block stack or adds a cached stack residual to the current stack input. When evaluation is selected, an inner planner updates its clustering state through three actions: reuse, repair, or refresh. The planner summarizes these actions into a historical stability score. The next outer decision uses that score in two ways: stable history discounts accumulated conditioning risk, while sufficiently low stability vetoes residual reuse. Residual magnitude history further adapts the permissible risk. The resulting dependency is directed and delayed: the planner of an executed step influences later decisions, not the decision that would be needed to observe its own current query and key tensors.

To isolate feedback within this controller, we retain the inner planner and remove only its influence on the outer controller, comparing fixed thresholds and one point matched at reported speed on Wan 2.1. This comparison uses the calibration cohort and is exploratory. A separate route map substitution tests stale inner routing, not the outer execution versus reuse choice.

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.

Our contribution is an explicit feedback path from historical sparse routing actions to later block reuse decisions, with a clear distinction between current observations available before execution and cached state. A nested planner maintains clustering on executed steps. Measurements across three backbones characterize quality versus latency operating points; Wan 2.1 controls isolate feedback at one point matched at reported speed and distinguish effects on routing from comparisons of outer policies.

2 Related Work

Feature reuse across steps. DeepCache and FORA reuse features from diffusion models, while Δ-DiT uses offsets (Ma et al., 2024; Selvaraju et al., 2024; Chen et al., 2024). For video, PAB broadcasts attention outputs, FasterCache exploits feature and guidance branch redundancy, and AdaCache adapts caching to content (Zhao et al., 2024; Lv et al., 2024; Kahatapitiya et al., 2025). TeaCache, MagCache, and SeaCache use input, magnitude, or spectral reuse signals (Liu et al., 2025; Ma et al., 2025; Chung et al., 2026); BWCache triggers blockwise reuse across timesteps from a similarity indicator (Cui et al., 2026). TraceGate instead uses historical inner planner state to modulate later residual reuse for the entire stack.

Sparse attention and temporal route maintenance. SVG exploits structured spatiotemporal sparsity; SVG2 uses semantic clustering, centroid caching, and permutation (Xi et al., 2025; Yang et al., 2025). AdaCluster adapts query/key clustering and cluster selection within sparse attention (Tan et al., 2026); HEART uses query and key drift to control temporal mask reuse for each head (Zheng et al., 2026). Our tested distinction is the use of planner action history in a later decision to reuse the *entire stack*. The control without feedback retains the same planner and cuts this connection, but it does not represent every independently optimized combination of caching and sparse attention.

3 Block Reuse Informed by Routing Topology

3.1 Reuse Scope and Observation Timing

Let k index denoising calls in execution order, with noise level t_k, latent x_k, and conditioning c. We use H_k^{in} and H_k^{out} for the input and output of the Transformer block stack *within* this call:

$$H_k^{\text{out}} = \mathcal{B}_\theta(H_k^{\text{in}}, c, t_k). \tag{1}$$

The stack residual is

$$r_k = H_k^{\text{out}} - H_k^{\text{in}}. \tag{2}$$

If $a < k$ is the most recent executed call with a compatible cache, the reuse path forms

$$\widetilde{H}_k^{\text{out}} = H_k^{\text{in}} + r_a. \tag{3}$$

The prediction head and sampler then process the current call as usual. Equation 3 is residual reuse, not interpolation between video frames or removal of sampler timesteps. An executed stack may itself use sparse attention; executing the stack is therefore distinct from dense attention and from replaying the original dense trajectory.

Before the gate acts, the current timestep projection e_k is available. Current internal Q_k, K_k, and r_k are not. The controller instead uses the cached residual magnitude history and the planner score η_a from the most recent executed call. Residual caches are maintained separately for classifier free guidance branches. The implementation also records any inner topology sharing between branches separately from the reuse/repair/refresh counts.

3.2 An inner planner with three update actions

During an executed stack, the planner maintains query/key cluster assignments, centroids, cluster sizes, and a sparse route map $\mathcal{M}_k^{(\ell)}$ for layer ℓ. Head and guidance branch indices are suppressed. A sampled token set Ω estimates drift from the previous available state at that layer:

$$d_k^{(\ell)} = \frac{1}{2}\left(\frac{\text{RMS}(Q_{k,\Omega}^{(\ell)} - Q_{a,\Omega}^{(\ell)})}{\text{RMS}(Q_{a,\Omega}^{(\ell)}) + \epsilon} + \frac{\text{RMS}(K_{k,\Omega}^{(\ell)} - K_{a,\Omega}^{(\ell)})}{\text{RMS}(K_{a,\Omega}^{(\ell)}) + \epsilon}\right). \tag{4}$$

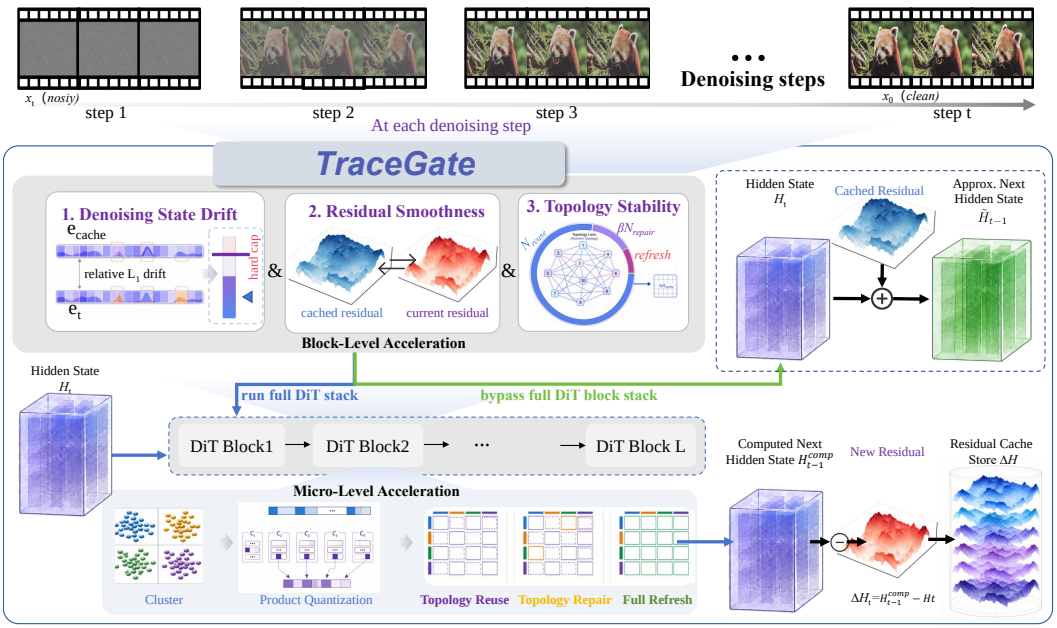

Figure 1: Overview of TraceGate. The outer gate decides whether to evaluate the DiT stack or reuse a cached stack residual; executed calls enter the inner planner, which maintains sparse attention through topology reuse, repair, or refresh. The signal panels summarize current conditioning together with statistics cached from previously executed calls. Planner state is fed back to *later* outer decisions; a skipped call does not compute its own current query and key tensors.

With a valid cache and a permitted reuse streak, small drift selects **reuse**: cached centroids provide a lightweight clustering update. Intermediate drift selects **repair**: current tokens are reassigned using cached centroids, followed by a small number of centroid updates before rebuilding the route map. Large drift or invalid history selects **refresh**: clustering is reconstructed from current tokens. In the recorded Wan 2.1 setting, the reuse/repair drift thresholds are 0.07/0.18, repair uses one centroid update iteration, and an inner reuse cap of four calls forces an update even when drift stays small (**Appendix A.2, Table 4**). Reuse here concerns the clustering state; it does not require the final attention mask to remain bitwise identical.

After the state update, centroid scores identify key cluster candidates. PQ (Jégou et al., 2011) provides an approximate shortlist, followed by exact centroid scoring *within* that shortlist and top p selection. The retained clusters define token pair blocks, which are permuted for sparse attention and then restored to the original order. The shortlist is not guaranteed to contain all keys selected by exact all cluster scoring. We consequently evaluate PQ against an exact control at a comparable expanded token pair budget in Section 4.6.

The planner summarizes its executed call actions as

$$\eta_k = \frac{N_{\text{reuse},k} + \beta N_{\text{repair},k}}{N_{\text{reuse},k} + N_{\text{repair},k} + N_{\text{refresh},k}}, \tag{5}$$

where $\beta \in [0, 1]$ discounts repair relative to reuse. This is an action history statistic, not a direct graph similarity measurement. The denominator counts only the three planner actions and excludes topology sharing between guidance branches. The score is stored for subsequent outer decisions; a missing score cannot be treated as evidence of a valid reusable state.

3.3 A GATE COMBINING DRIFT AND HISTORICAL FEEDBACK

Conditioning risk accumulation. The relative change in the timestep projection supplies a cheap input based signal:

$$\delta_k = \frac{\|e_k - e_{\text{ref}}\|_1}{\|e_{\text{ref}}\|_1 + \epsilon}. \tag{6}$$

Here e_{ref} denotes the stored conditioning reference used by the controller. A calibrated polynomial g maps this change into a nonnegative increment $s_k = \max(0, g(\delta_k))$. Historical planner feedback discounts the increment:

$$R_k = R_{k-1} + s_k \max(\epsilon_s, 1 - \lambda\eta_a). \tag{7}$$

The discount is a soft efficiency mechanism: larger η_a permits more reuse for the same input based change. It is not an estimate of current, unobserved attention topology.

Residual magnitude history. On an executed call, the ratio $\rho_k = \|r_k\|_2 / (\|r_a\|_2 + \epsilon)$ updates an exponential moving average. On a skipped call, the controller uses an effective ratio ρ_k^* formed from the historical estimate and an optional timestep prior; the prior formula is given in **Appendix A.2**, with the recorded constants summarized in **Table 4**. Within a reuse streak, the accumulated magnitude ratio and its risk score are

$$P_k = P_{k-1}\rho_k^*. \tag{8}$$

$$E_k = E_{k-1} + |1 - P_k|. \tag{9}$$

These quantities track predicted amplitude change. Equal residual norms do not imply equal residual vectors; E_k is therefore a risk proxy rather than a bound on output or video error.

Soft relaxation and hard veto. A small magnitude risk estimate allows a larger conditioning risk threshold:

$$\tau_k = \tau_r \left[1 + \gamma \operatorname{clip}\left(1 - \frac{E_k}{\tau_m}, 0, 1\right)\right]. \tag{10}$$

The complete feedback path also vetoes reuse if historical stability falls below $\eta_{\min}$. Let v_k indicate cache validity, n_k the proposed consecutive reuse length, and $\mathcal{P}$ the protected calls. The gate selects reuse exactly when

$$G_k = v_k \, \mathbb{I}[k \notin \mathcal{P}] \, \mathbb{I}[n_k \le K] \, \mathbb{I}[R_k < \min(\tau_k, \alpha\tau_r)] \, \mathbb{I}[E_k < \tau_m] \, \mathbb{I}[\eta_a \ge \eta_{\min}]. \tag{11}$$

Otherwise it executes the stack, updates r_a and planner history, and resets the streak accumulators. The update order is in **Appendix A, Algorithm 1**. Setting $\lambda = 0$ removes soft feedback; setting $\eta_{\min} = 0$ disables the low stability veto when a valid score exists. Disabling both produces the controller without feedback while retaining the planner. The soft path encourages reuse, whereas the hard path spends computation on low stability histories. Under the recorded Wan 2.1 settings, the $\alpha\tau_r$ cap is inactive (**Appendix A.2**).

Rationale for Combining the Signals. The observations enter at different points in the reuse loop. Conditioning drift is available before execution and supplies a risk increment; residual history estimates amplitude persistence; planner history summarizes recent routing updates. An unconstrained weighted sum would let a favorable signal cancel a large residual risk estimate or a low historical planner score. We instead retain the residual constraint and veto based on low stability while using graded history to alter risk accumulation. This is a design choice grounded in observation timing, not a guarantee that this coupling is uniquely optimal. The controls without feedback and with only soft or hard feedback below test its behavior within the recorded architecture.

3.4 Nested execution and cache lifecycle

At each sampler call, the outer gate first reads the current conditioning projection and valid historical caches. A reused call adds the stored residual to the current stack input and leaves the inner planner state unchanged. An executed call instead evaluates the stack, chooses reuse, repair, or refresh for eligible attention layers from current Q/K, and then stores the new residual and planner action score for later calls. Skipping a stack therefore postpones the inner planner update: the next executed call may compare Q/K with a state several sampler calls old. This bidirectional dependency motivates feedback without treating cached observations as measurements of the current uncomputed attention. The update order and guidance branch handling are in **Appendix A, Algorithm 1**. Rejecting reuse evaluates the current stack at the current latent; it cannot undo errors from earlier denoising calls.

4 EXPERIMENTS

4.1 EVALUATION PROTOCOL

We evaluate Wan 2.1-I2V-14B at 720p with 81 frames, Wan 2.2-I2V-A14B at 480p with 81 frames, and HunyuanVideo-T2V-13B at 720p with 129 frames (Wan Team, 2025a;b; Kong et al., 2024). Frame count is distinct from the denoising call count. The I2V evaluation uses paired VBench image and prompt inputs cropped to 16:9; the expanded comparison of task quality uses 355 pairs. HunyuanVideo uses the Penguin prompt set released with the official evaluation resources (Hunyuan Foundation Model Team, 2025). The component, feedback, and calibration analyses use the same Wan 2.1 cohort of 108 videos, separate from the cohort used for the main results; these results must not be pooled or treated as independent feedback validation.

Quality and latency. VBench and VBench++ provide task-level evaluation for video generation, including the I2V dimensions used here (Huang et al., 2024; 2026). VBench measures video properties at the task level. Table 1 reports descriptive unweighted means over eight I2V dimensions excluding camera for Wan and sixteen T2V dimensions for HunyuanVideo; the scores for each dimension are in **Appendix B, Tables 10 through 13**. These expanded dimension sets were reported during the earlier review after an initial summary of five dimensions, so the means are not prespecified primary endpoints or official normalized VBench Overall scores; they should not be compared across I2V and T2V. Paired PSNR, SSIM(Wang et al., 2004), and LPIPS (Zhang et al., 2018) instead measure deviation from the corresponding dense model output, not accuracy against a ground truth video. Speedups are measured for the complete inference run under the same hardware, resolution, frame count, and denoising schedule as each model's dense reference. The reported hardware is a server with four NVIDIA H100 GPUs and an Intel Xeon Platinum 8558 CPU.

Comparators and operating points. We compare SVG2, TeaCache, MagCache, and SeaCache where results are available. TraceGate-F denotes the setting focused on efficiency, and TraceGate-Q denotes the setting focused on quality. They are configurations of the same algorithm, not separately trained models. All speedups in the tables are measurements from this evaluation, not the maximum speedups reported by baseline papers. The settings depend on the backbone; calibration transfer without retuning was not tested. The available operating points for individual methods are not a benchmark of independently optimized combinations of caching and sparsity. We distinguish these comparisons from ablations at fixed thresholds and the retuned control without feedback.

4.2 QUALITY AND LATENCY OPERATING RANGE

Table 1 shows regimes focused on quality and efficiency. On Wan 2.1, TraceGate-F reaches 3.36× with 8.46% density, while SVG2 retains the highest paired PSNR/SSIM/LPIPS among the listed accelerated methods. On Wan 2.2, TraceGate-Q reaches 27.62 PSNR / 0.861 SSIM / 0.068 LPIPS at 1.51×, whereas TraceGate-F reaches 2.45× with 8.62% density. The VBench mean of TraceGate-F is below Dense on Wan 2.2 (0.8165 versus 0.8220). The Wan 2.2 dimension scores in **Appendix B, Table 11** show identical reported motion smoothness (0.9880) but lower dynamic degree (0.33333 versus 0.35772), so motion smoothness alone does not establish unchanged dynamics.

On HunyuanVideo, TraceGate-Q has lower LPIPS than MagCache (0.047 versus 0.051) and higher speedup (2.17× versus 1.99×), while MagCache has higher PSNR (32.73 versus 32.49). TraceGate-F extends the speed range to 3.02× with 11.54% density, but its VBench mean is below Dense (0.6974 versus 0.7007). These operating points show a configurable quality versus latency tradeoff rather than universal domination; the task specific dimension scores remain visible in **Appendix B, Tables 12 and 13**.

Figure 2 displays the evaluated quality versus speed operating ranges. TraceGate combines less work during each execution with residual reuse to reach the faster part of this range. Baseline measurements remain specific to the evaluated configurations and execution stack; the plot does not establish an upper bound on acceleration across architectures.

Table 1: Task quality, fidelity relative to the dense reference, and measured speed at the reported operating points. The VBench column is a retrospectively expanded, unweighted mean of dimensions specific to each task, not the official VBench Overall score. F and Q denote settings focused on efficiency and quality, respectively. For Wan 2.2, density is normalized to native dense MoE inference, with one of two experts active per step.

Method	Task quality and dense reference fidelity				Efficiency	
	VBench ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Density ↓	Speedup ↑
Wan 2.1 14B (720p, Image to Video)						
Dense	0.8353	–	–	–	100%	1.00×
SVG2	0.8351	**25.84**	**0.821**	**0.105**	29.89%	1.12×
MagCache	0.8346	19.83	0.664	0.287	35.00%	2.57×
TeaCache	0.8361	22.55	0.752	0.160	40.00%	2.72×
SeaCache	0.8403	21.34	0.709	0.215	55.00%	1.76×
TraceGate-F	**0.8436**	24.07	0.790	0.133	**8.46%**	**3.36**×
Wan 2.2 A14B (480p, Image to Video)						
Dense	**0.8220**	–	–	–	100%	1.00×
SVG2	0.8208	27.20	0.854	0.071	27.97%	1.45×
MagCache	0.8185	22.11	0.725	0.153	47.50%	2.03×
TeaCache	0.8201	22.29	0.729	0.145	95.00%	1.11×
SeaCache	0.8182	22.63	0.742	0.142	69.70%	1.39×
TraceGate-Q	0.8219	**27.62**	**0.861**	**0.068**	28.01%	1.51×
TraceGate-F	0.8165	25.69	0.823	0.091	**8.62%**	**2.45**×
HunyuanVideo 13B (720p, Text to Video)						
Dense	0.7007	–	–	–	100%	1.00×
SVG2	0.6984	29.93	0.908	0.085	23.94%	1.13×
MagCache	0.7011	**32.73**	**0.939**	0.051	38.00%	1.99×
TeaCache	0.7024	20.87	0.766	0.234	42.33%	1.53×
TraceGate-Q	**0.7086**	32.49	0.936	**0.047**	31.05%	2.17×
TraceGate-F	0.6974	31.39	0.924	0.060	**11.54%**	**3.02**×

Notes: Density measures attention pair workload over the execution trajectory (**Appendix D, Equation 15**); speedup is based on measured wall clock time. The displayed VBench values are rounded from the scores for each dimension in **Appendix B, Tables 10 through 13**.

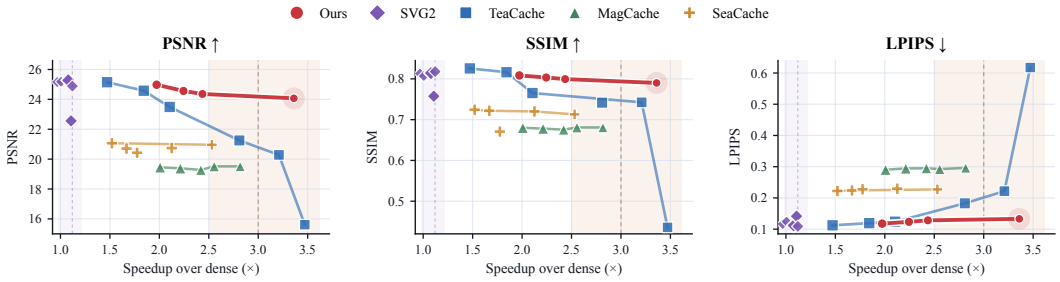

Figure 2: Wan 2.1 quality versus speed operating points. "Ours" in the source figure denotes TraceGate configurations. The curve connects measured configurations for readability; intermediate locations are not additional measurements. Methods overlap over different parts of the speed range.

4.3 Contributions of the two reuse levels

Table 2 separates the outer block reuse from the inner planner on the cohort of 108 videos. With block reuse disabled, the planner alone achieves 2.04×; the complete configuration reaches 3.30×. Their reported normalized latencies are $1/2.04$ and $1/3.30$, respectively, corresponding to a 38.2% latency reduction with block reuse enabled. PSNR changes from 23.71 to 23.51 dB. This is an observed tradeoff at those operating points, not a claim that the two speedups multiply independently. Removing conditioning drift, residual control, or topology feedback makes the fixed configuration faster and changes fidelity; these rows alone cannot establish each signal's value at equal latency. Section 4.4 examines feedback at comparable reported speed; Section 4.6 separately controls PQ's pair budget.

Table 2: Fixed configuration component ablation on 108 Wan 2.1 videos; rows differ in speed.

Configuration	Visual Quality			Efficiency	
	PSNR ↑	SSIM ↑	LPIPS ↓	D_{traj} ↓	Speedup ↑
w/o Block Reuse	**23.71**	**0.783**	**0.131**	15.64%	2.04×
w/o Conditioning Drift	23.35	0.776	0.140	–	3.70×
w/o Residual Control	23.47	0.779	0.140	16.79%	3.59×
w/o Topology Feedback	23.22	0.772	0.138	**7.73%**	**4.12**×
w/o PQ Shortlist	23.90	0.787	0.128	13.93%	3.08×
Complete TraceGate	23.51	0.779	0.137	8.46%	3.30×

Table 3: Topology feedback controls on 108 paired Wan 2.1 videos: fixed threshold decomposition and matched reported speed.

Configuration	Feedback		Visual Quality			Efficiency
	λ	$\eta_{\min}$	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑
Fixed threshold decomposition						
No feedback	0	0	23.22	0.772	0.138	4.12×
Soft only	0.20	0	23.07	0.766	0.143	**4.40**×
Hard only	0	0.05	**23.54**	**0.781**	**0.135**	3.05×
Complete feedback	0.20	0.05	23.51	0.779	0.137	3.30×
Matched reported speed						
No feedback, retuned	0	0	23.38	0.776	0.137	3.30×
Complete feedback	0.20	0.05	**23.51**	**0.779**	0.137	3.30×

4.4 Contribution of Planner Feedback

Table 3 separates two experiments on the same cohort of 108 videos. With the other settings fixed, removing the topology veto allows additional reuse and changes both quality and latency. Hard feedback alone emphasizes fidelity, while soft feedback alone is more aggressive. Combining both yields 23.51 PSNR at 3.30×, between these operating regimes.

To reduce confounding by computation budget, the controller without feedback is retuned by tightening only τ_r and τ_m, keeping the planner, PQ, reuse cap, and protected calls fixed. At $(\tau_r, \tau_m) = (0.11, 0.07)$, it reaches the same reported 3.30× as the complete controller at $(0.20, 0.12)$. Mean PSNR and SSIM are higher with feedback by 0.13 dB and 0.003; LPIPS is equal at three decimals. The two speedups are matched only at displayed precision. Because calibration and this comparison use the same 108 videos, the result is exploratory; aggregate means give neither an uncertainty bound for paired differences nor an independent test of the selected setting. The threshold rows are in **Appendix C, Table 14**.

This control disconnects planner feedback within the same nested system; it is not a fully optimized independent composition of sparse attention and caching. The complete operating point was selected from a scan of 18 configurations over seven parameters, while the recorded retuning without feedback covers four pairs of outer thresholds with other settings held fixed. The two searches therefore have different scope. The displayed 3.30× comparison tests this limited ablation at one operating point, rather than establishing a superior Pareto frontier over independently tuned compositions.

Sensitivity to nearby thresholds. The local $2 \times 2 \times 2$ crossed control in **Appendix C, Table 15** varies two conditioning thresholds, two residual thresholds, and feedback enabled or disabled. At all four fixed threshold pairs, feedback has higher PSNR by 0.26 to 0.29 dB and SSIM by 0.007 to 0.008, with lower LPIPS but lower speedup. This shows local sensitivity of the fixed threshold comparison, not four additional equal latency gains; Table 3 offers one check at matched reported speed.

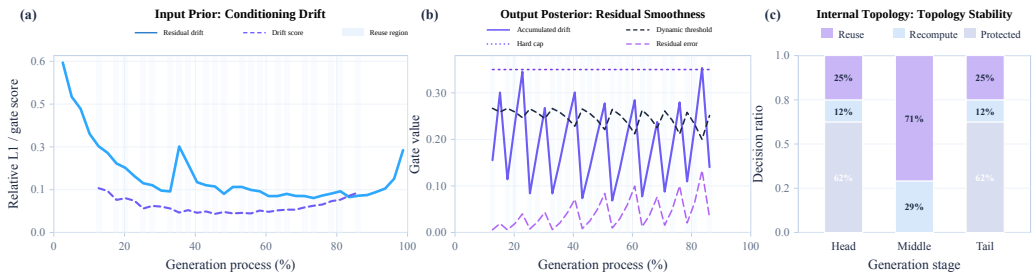

Figure 3: Runtime diagnostics for the three controller states and resulting stagewise decisions. Conditioning drift and residual smoothness vary along the trajectory, while reuse concentrates in the stable middle stage, reaching 71% of decisions there; protected and recomputed calls are more prominent toward the ends. These traces show where the gate acts, but do not bound error.

4.5 Stale route effects within executed stacks

We examine 72 events from 108 traces, with at most one event per video, under a screen fixed before viewing intervention outcomes: all criteria except topology permit reuse, the topology criterion vetoes it, and current and cached route maps differ. Each event has an unchanged replay, a protected execution using the current map, and an execution using the stale map. The stale arm changes only the route map, holding target Q/K/V fixed and replaying protected downstream decisions.

Unchanged replay reproduces 72/72 outputs. Map substitution changes the target block output in 72/72 events and propagates to the final video in 69/72. Protected execution has higher PSNR relative to the dense output in 54/72 selected events and improves mean PSNR by 0.23 dB, with the reported 95% interval $[0.14, 0.32]$; corresponding mean SSIM and LPIPS differences are $+0.006$ and -0.003. These controls show that stale routing can cause downstream error in selected events, not that protection helps every event. The screen conditions on a topology veto and a changed route map, so these events do not estimate how often the same error occurs across all calls. Both intervention arms execute the target stack; they do not replace execution with reuse of the cached residual. Section 4.4 compares outer policies at one point matched at reported speed; this intervention localizes inner routing error.

4.6 Distribution level robustness

Figure 4 reports paired PSNR and SSIM distributions over the Wan 2.1 evaluation cohort and representative categories. In this unmatched operating point comparison, differences from the stronger of two plotted cache baselines are +1.53 dB PSNR and +0.038 SSIM overall, and +1.87 dB and +0.077 at the 10th percentile. The lower tail differences are not confined to high score samples; the plot does not establish a matched latency advantage over these baselines.

Workload versus measured time. In the trace of 108 videos, removing residual magnitude control reduces executed stacks from 4,248 to 3,905 but raises attention density over the trajectory from 8.46% to 16.79%; speedup changes from 3.30× to 3.59×. Fewer evaluations of the full stack can therefore coexist with more modeled attention work. The definition weighted by counts and the action counts are in **Appendix D, Equation 15 and Tables 17 through 18**; this does not attribute runtime to individual kernels.

PQ at a comparable attention pair budget. PQ and exact ranking across all clusters at nearly equal attention pair budgets yield 8.4628% versus 8.4555% attention density over the trajectory and 3.30× versus 3.29×. The exact control retaining 90% of the score mass uses a larger pair budget and is not the matched comparison. PQ is an approximate planner implementation, not the dominant source of overall speedup; the exact controls and diagnostics are in **Appendix E, Table 19**.

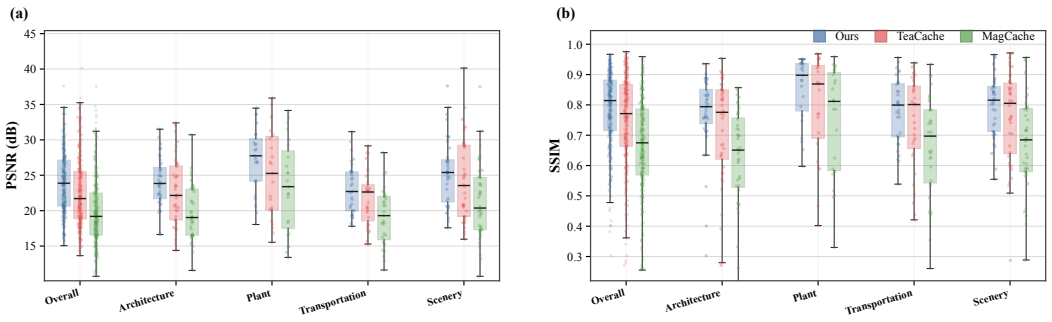

Figure 4: Wan 2.1 paired dense reference fidelity over the evaluation cohort and representative categories. "Ours" denotes TraceGate; other boxes show TeaCache and MagCache. Lower tails show low fidelity samples.

Qualitative results. Wan 2.1, Wan 2.2, and HunyuanVideo examples are in **Appendix F, Figures 8 through 13**. The dense outputs are generated references rather than ground truth future videos.

5 Discussion and Limitations

Across three backbones, nested stack reuse and sparse attention accelerate inference without retraining. The key design is a directed information path: conditioning drift is available before an outer decision, whereas query and key routing is observed only after execution. The planner summarizes past reuse, repair, and refresh actions into η for later decisions. Thus, a skipped call never needs to reveal its uncomputed route.

The Wan 2.1 controls isolate feedback within the recorded controller. Fixed thresholds reveal its quality versus speed cost, while the retuned control aligns speed only at reported precision. Retuning without feedback covers four threshold pairs; the complete controller was selected across 18 configurations and seven parameters. The comparison reuses the calibration cohort of 108 videos and lacks unrounded latency, uncertainty for paired differences, and task quality at matched speed. Its PSNR and SSIM differences therefore do not rank independently optimized caching and sparsity combinations.

Substituting stale routes affects selected final videos, but the screen requires a topology veto and differing maps; both arms execute the target stack. This tests routing approximation, not the outer execution versus residual reuse action. Moreover, η records actions rather than graph distance or residual error. Whether it adds information beyond cache age, skipped calls, timestep, or recent query and key drift remains unknown. Proxy controls and forced outer reuse would test that causal chain.

The three backbone results establish system operating points, not feedback gains on every model; each uses its own settings. The Wan 2.1 cap is inactive by default, planner history can lag change, residual magnitudes miss direction, and PQ may omit candidates. The 18 configuration sweep is descriptive. Reproduction needs the fitted polynomial, prior, reference update rule, protected calls, and execution presets. An independent test should freeze these settings, collect unrounded times and paired video differences on new samples, and compare with separately tuned caching and sparsity at matched budget.

6 Conclusion

TraceGate links temporal sparse attention maintenance to later residual reuse. The planner records routing actions on executed calls; the outer controller uses that history through soft and hard feedback. Three backbones show configurable quality versus latency operating points. Controls at matched reported speed and with stale routing clarify feedback within one Wan 2.1 calibration cohort. Because routing is recorded only after execution, the controller can use it without assuming that skipped calls reveal uncomputed attention routes.

AI USE STATEMENT

AI tools assisted manuscript restructuring, language editing, bibliographic lookup, LaTeX editing, and critical review of the method description and interpretation of already reported experiments. They did not generate experimental measurements. The authors are responsible for verifying AI assisted changes and for all claims, results, and the final submission.

ETHICS STATEMENT

This work accelerates pretrained video generators without adding new model level safety mechanisms. Lower generation cost may amplify both beneficial and harmful uses, so the safeguards and provenance practices of the underlying models remain applicable.

REPRODUCIBILITY STATEMENT

Observation timing and constants are in **Appendix A, Algorithm 1 and Table 4**; the Wan 2.1 calibration operating points are in **Appendix A.3, Tables 6 through 7**, the additional Wan 2.2 and HunyuanVideo profiles are in **Tables 8 through 9**, and task-level metric vectors are in **Appendix B, Tables 10 through 13**. Exact reproduction also requires the model dependent polynomial, prior, and execution configurations.

REFERENCES

Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-DiT: A training-free acceleration method tailored for diffusion transformers. *arXiv preprint arXiv:2406.01125*, 2024.

Jiwoo Chung, Sangeek Hyun, MinKyu Lee, Byeongju Han, Geonho Cha, Dongyoon Wee, Youngjun Hong, and Jae-Pil Heo. SeaCache: Spectral-evolution-aware cache for accelerating diffusion models. *arXiv preprint arXiv:2602.18993*, 2026.

Hanshuai Cui, Zhiqing Tang, Zhifei Xu, Zhi Yao, Wenyi Zeng, and Weijia Jia. BWCache: Accelerating video diffusion transformers through block-wise caching. In *International Conference on Learning Representations*, 2026.

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

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(3):3268–3285, 2026. doi: 10.1109/TPAMI.2025.3633890.

Hunyuan Foundation Model Team. Penguin video benchmark. `https://github.com/Tencent-Hunyuan/HunyuanVideo`, 2025.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S. Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15240–15252, 2025.

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.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7353–7363, 2025.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. FasterCache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 2024.

Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast video generation with magnitude-aware cache. *arXiv preprint arXiv:2506.09045*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. FORA: Fast-forward caching in diffusion transformer acceleration. *arXiv preprint arXiv:2407.01425*, 2024.

Haoyue Tan, Shengnan Wang, Yulin Qiao, Juncheng Zhang, Youhui Bai, Ping Gong, Zewen Jin, and Cheng Li. AdaCluster: Adaptive query-key clustering for sparse attention in video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43249–43259, 2026.

Wan Team. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025a. URL `https://arxiv.org/abs/2503.20314`.

Wan Team. Wan2.2: Open video generation models. `https://github.com/Wan-Video/Wan2.2`, 2025b.

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. doi: 10.1109/TIP.2003.819861.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, Jianfei Chen, Ion Stoica, Kurt Keutzer, and Song Han. Sparse VideoGen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, Jianfei Chen, Song Han, Kurt Keutzer, and Ion Stoica. Sparse VideoGen2: Accelerate video generation with sparse attention via semantic-aware permutation. *arXiv preprint arXiv:2505.18875*, 2025. Version 5, revised May 2026.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. *arXiv preprint arXiv:2408.12588*, 2024.

Xuzhe Zheng, Yuexiao Ma, Jing Xu, Xiawu Zheng, Rongrong Ji, and Fei Chao. HEART: Exploiting head heterogeneity in sparse attention for video diffusion. *arXiv preprint arXiv:2605.14513*, 2026. Version 2, revised August 2026.

A Controller Details and Calibration

A.1 Observation timing and state updates

Algorithm 1 separates the outer decision from the inner observations it may cause. At an outer decision, only the current stack input, timestep projection, and historical cache are available. Current layer query/key tensors and the new stack residual are observed only if the stack executes. During a residual reuse streak, the last executed planner score is retained rather than recomputed from unobserved tensors.

Algorithm 1 Nested residual reuse and planner feedback with cache state local to each guidance branch

Require: Ordered denoising calls; controller and planner settings; protected calls $\mathcal{P}$

Initialize an invalid residual/planner cache; $R \leftarrow 0$, $E \leftarrow 0$, $P \leftarrow 1$, $n \leftarrow 0$
for call k in the original sampler schedule **do**
  Obtain H_k^{in} and current timestep projection e_k
  if the residual cache and required historical score are valid **then**
    Evaluate conditioning drift using its stored reference (Eq. 6)
    Form ρ_k^* from available residual history and the configured prior
    Update proposed R_k, P_k, E_k and proposed streak length $n_k = n + 1$
    Evaluate Eq. 11 using η_a from the most recent executed call
  else
    $G_k \leftarrow 0$
  end if
  if $G_k = 1$ **then**
    $H_k^{\text{out}} \leftarrow H_k^{\text{in}} + r_a$; commit proposed accumulators and n_k
    Retain historical residual and planner observations
  else
    Execute the Transformer stack on the current input
    Within each eligible attention layer, compute current Q, K, V
    Select reuse, repair, or refresh from Q/K drift and cache validity
    Update cluster state; form a PQ shortlist and exact within the shortlist scores
    Construct sparse routes, permute tokens, execute attention, and restore the original order
    Collect planner action counts; form η_k using Eq. 5
    Compute $r_k = H_k^{\text{out}} - H_k^{\text{in}}$; update residual magnitude history
    Cache the executed call observations; reset R, E, P, n to $0, 0, 1, 0$
  end if
  Return the current branch prediction to the native guidance/sampler path
end for

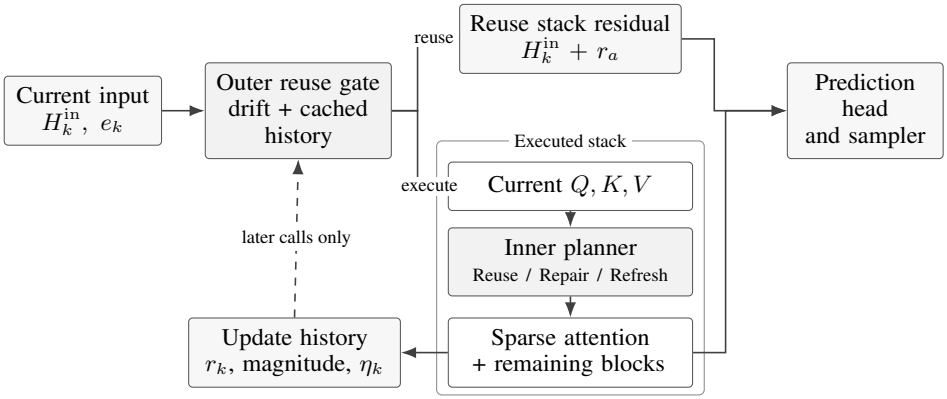

Figure 5: Observation timing for the outer gate and inner planner. Current query and key tensors become available only on executed calls; the return edge therefore updates future reuse decisions.

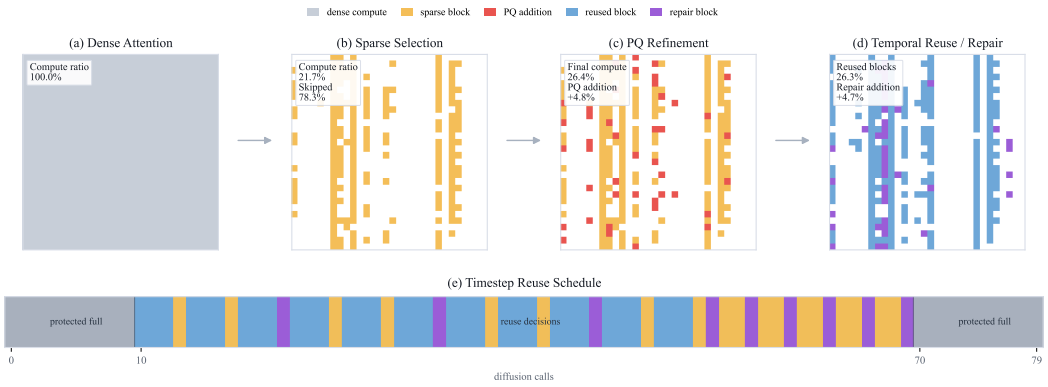

Figure 6: Recorded execution trace illustrating how dense attention is reduced by sparse selection, refinement with PQ, temporal reuse or repair, and the outer reuse schedule across timesteps. This trace visualizes the nested computation structure; its percentages describe the shown configuration rather than a universal decomposition.

The native classifier free guidance implementation combines branch predictions before its sampler update; Algorithm 1 does not update the diffusion latent independently inside each branch. The conditioning reference and residual anchor are conceptually distinct state variables. The projection reference used by the polynomial must follow the same update convention as its calibration. Likewise, the residual ratio prior must correspond to the backbone and timestep schedule. Changing these conventions while retaining fitted coefficients changes the controller, even if the nominal thresholds are unchanged. Dense fallback within attention, protection of selected layers/calls, and sharing between guidance branches are separate implementation choices; an executed stack is not synonymous with fully dense inference.

A.2 Recorded implementation constants

The residual ratio estimate is updated after an executed call using

$$\overline{\rho} \leftarrow \mu\overline{\rho} + (1-\mu)\rho_k. \tag{12}$$

The available implementation description combines this online estimate with a configured timestep prior:

$$\rho_k^* = \omega\rho_{\text{prior},k} + (1-\omega)\overline{\rho}. \tag{13}$$

These are calibration and control operations; they do not retrain the generative model. Table 4 lists constants recorded in the method description and the feedback controls. The polynomial coefficients, any prior table specific to a model, protected call definitions, and presets for each backbone are part of the configuration rather than implied by the claim that the method requires no training.

For these recorded Wan 2.1 defaults, $\gamma = 0.35$ gives $\tau_k \leq 1.35\tau_r$, below the $\alpha\tau_r = 1.75\tau_r$ cap. The cap is therefore inactive in this reported regime, and none of the reported Wan 2.1 comparisons identifies a benefit from it. It is retained in the general controller for other gain/cap combinations without claiming that those combinations were evaluated. The companion method note names the outer parameters ROUTE_MAGBOOST_ROUTE_THRESH, ROUTE_MAGBOOST_MAG_THRESH, ROUTE_MAGBOOST_K, ROUTE_MAGBOOST_ROUTE_GAIN, ROUTE_MAGBOOST_HARD_ROUTE_MULT, ROUTE_MAGBOOST_TABLE_BLEND, and ROUTE_MAGBOOST_ONLINE_MOMENTUM; the three planner controls are CARE_REUSE_THRESHOLD, CARE_REPAIR_THRESHOLD, and CARE_MAX_REUSE_STEPS. These names aid configuration matching but do not replace the missing executable run configuration. In particular, the planner's repair discount β, polynomial coefficients, reference update rule, and residual ratio prior must be obtained from the actual run.

Table 4: Recorded constants for the described Wan 2.1 controller and planner. The soft/hard settings are those used in the feedback decomposition. A listed default is not a claim that every backbone uses the same value.

Quantity	Role	Recorded value
τ_r	Base conditioning risk threshold	0.20
τ_m	Residual magnitude risk threshold	0.12
K	Maximum outer reuse streak	3
γ	Threshold gain from residual history	0.35
α	Conditioning risk cap multiplier	1.75
ω	Mixture weight for timestep prior	0.35
μ	Online residual ratio momentum	0.80
λ	Soft stability discount	0.20
$\eta_{\min}$	Hard low stability veto	0.05
$\lvert\Omega\rvert$	Sampled Q/K token count	64
τ_{reuse}	Planner reuse threshold	0.07
τ_{repair}	Planner repair threshold	0.18
K_{attn}	Maximum inner reuse streak	4
Repair iterations	Cached initialization centroid updates	1
PQ candidate ratio	Ordinary / higher risk layers	0.20 / 0.45

A.3 Descriptive Calibration Across 18 Configurations

The Wan 2.1 sweep varies seven parameters at three levels and evaluates 18 configurations on the same cohort of 108 videos used for the feedback controls, rather than the full 3^7 grid. The reported efficiency operating point was selected by considering multiple objectives in the quality versus speed tradeoff, not a prespecified scalar objective; the feedback comparison at matched reported speed on this cohort is therefore exploratory. Each parameter level appears six times, but balance within each column does not imply an orthogonal array of strength two. For example, the pair of residual threshold and key cluster ratio has the counts in Table 5. The sweep displays observed operating points, not independent causal main effects or weak interactions over the full domain. No claim of global optimality is made.

Table 5: Joint counts for two columns of the actual sweep of 18 configurations. An orthogonal design of strength two would place two observations in every cell.

Residual threshold	Cluster ratio 0.08	0.10	0.12
0.070	3	3	0
0.095	0	3	3
0.120	3	0	3

A.4 Additional backbone configurations

The main table settings focused on quality correspond to the recorded QualityExtreme (Q) profile below, denoted TraceGate-Q in the main table. The additional profiles illustrate operating ranges; they are not independent tests of a universal set of numerical thresholds. Table 8 preserves the recorded Wan 2.2 profiles, and Table 9 preserves the HunyuanVideo calibration results. The default settings focused on efficiency in Table 1 must not be silently identified with a different quality profile row.

Table 6: Actual Wan 2.1 calibration configurations. p is the cluster selection threshold; r_{KC} is the minimum key cluster ratio. f_t and f_ℓ are the protected time/layer fractions. Fast is a historical aggressive configuration, not a claimed Pareto optimum.

ID	K	τ_r	τ_m	p	r_{KC}	f_t	f_ℓ
Fast	4	0.24	0.150	0.90	0.08	0.30	0.02
TraceGate-F	3	0.20	0.120	0.90	0.10	0.35	0.03
E01	1	0.14	0.070	0.90	0.08	0.35	0.03
E02	1	0.17	0.095	0.93	0.10	0.40	0.04
E03	1	0.20	0.120	0.95	0.12	0.45	0.05
E04	2	0.14	0.095	0.95	0.10	0.35	0.05
E05	2	0.17	0.120	0.90	0.12	0.40	0.03
E06	2	0.20	0.070	0.93	0.08	0.45	0.04
E07	3	0.14	0.120	0.93	0.12	0.35	0.04
E08	3	0.17	0.070	0.95	0.08	0.40	0.05
E09	3	0.20	0.095	0.90	0.10	0.45	0.03
E10	1	0.14	0.095	0.93	0.12	0.40	0.03
E11	1	0.17	0.120	0.95	0.08	0.45	0.04
E12	1	0.20	0.070	0.90	0.10	0.35	0.05
E13	2	0.14	0.120	0.90	0.08	0.45	0.05
E14	2	0.17	0.070	0.93	0.10	0.35	0.03
E15	2	0.20	0.095	0.95	0.12	0.40	0.04
E16	3	0.14	0.070	0.95	0.10	0.45	0.03
E17	3	0.17	0.095	0.90	0.12	0.35	0.04
E18	3	0.20	0.120	0.93	0.08	0.40	0.05

Table 7: Wan 2.1 sweep results on the same 108 valid videos. These are calibration cohort results, distinct from the main result cohort in Table 1. Latency values and speedups are retained at their reported precision.

ID	PSNR	SSIM	LPIPS	D_{traj}	Speedup	Time (s)
Dense	–	–	–	100.00%	1.00×	1525.40
Fast	21.3063	0.7355	0.1674	7.58%	3.62×	421.50
TraceGate-F	23.5071	0.7791	0.1366	8.46%	3.30×	461.82
E01	23.2033	0.7705	0.1416	9.63%	2.80×	544.95
E02	23.6794	0.7822	0.1302	11.31%	2.62×	582.65
E03	23.9080	0.7866	0.1262	12.92%	2.49×	612.65
E04	23.5903	0.7797	0.1337	10.88%	2.86×	533.93
E05	23.6897	0.7826	0.1322	9.57%	3.04×	501.36
E06	23.5493	0.7789	0.1339	9.23%	3.05×	500.51
E07	23.6744	0.7824	0.1328	10.63%	2.95×	516.91
E08	23.6263	0.7804	0.1328	10.09%	2.99×	509.63
E09	23.6675	0.7825	0.1342	8.58%	3.13×	486.71
E10	23.7872	0.7847	0.1276	11.95%	2.62×	581.96
E11	23.7395	0.7828	0.1294	11.93%	2.52×	604.33
E12	23.3238	0.7736	0.1387	10.30%	2.72×	561.54
E13	23.4904	0.7757	0.1404	8.81%	2.85×	535.13
E14	23.4857	0.7785	0.1338	9.52%	3.15×	484.38
E15	23.7927	0.7851	0.1293	10.86%	2.99×	510.78
E16	23.7758	0.7846	0.1291	11.21%	2.79×	546.22
E17	23.4985	0.7802	0.1395	9.45%	3.12×	489.16
E18	23.4090	0.7759	0.1403	8.90%	3.21×	474.87

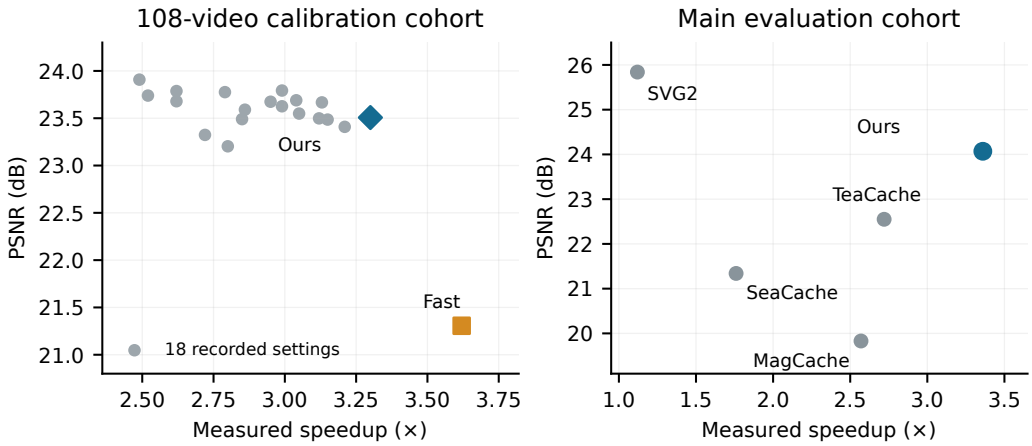

Figure 7: Wan 2.1 quality versus speed operating points. "Ours" marks the TraceGate-F configuration in each panel. The left panel shows the 18 recorded settings and two selected profiles on the calibration cohort of 108 videos; the right panel shows the reported baseline points from the main cohort. The panels use different cohorts and do not constitute comparisons at matched speed.

Table 8: Recorded Wan 2.2 quality profile configurations and results. Cutoff is the recorded schedule cutoff.

Profile	p	r_{KC}	f_t	f_ℓ	K	τ_r	τ_m	Cutoff
QualityFast	.93	.12	.40	.04	3	.20	.10	36
QualityGuard	.95	.12	.45	.04	2	.18	.08	35
QualityUpper	.97	.14	.50	.05	2	.16	.06	34
QualityFastPlus	.95	.12	.42	.04	3	.19	.09	36
QualityExtreme (Q)	.99	.16	.60	.06	2	.14	.05	32

Profile	PSNR	SSIM	LPIPS	Reported density	Speedup
QualityFast	26.3454	.8390	.0807	19.05%	1.98×
QualityGuard	26.4978	.8420	.0790	20.06%	1.84×
QualityUpper	27.0202	.8518	.0733	22.88%	1.66×
QualityFastPlus	26.4295	.8410	.0797	20.06%	1.92×
QualityExtreme (Q)	27.6219	.8607	.0676	28.01%	1.51×

Table 9: Recorded HunyuanVideo quality profiles. Calibration results use the same 946 valid videos.

Profile	K	τ_r	τ_m	p	r_{KC}	f_t	f_ℓ
QualityExtreme (Q)	2	.16	.08	.95	.12	.20	.04
QualityGuard	3	.20	.12	.90	.10	.10	.03
QualityFast	6	.24	.16	.90	.10	.06	.03

Profile	PSNR	SSIM	LPIPS	Reported density	Speedup
QualityExtreme (Q)	32.4934	.935938	.047185	31.05%	2.1732×
QualityGuard	29.1948	.908080	.077824	24.56%	2.6056×
QualityFast	24.2128	.840772	.151410	24.81%	2.7943×

B VIDEO QUALITY BY TASK

VBench dimensions (Huang et al., 2024) and the I2V extension in VBench++ (Huang et al., 2026) are retained individually so that a compact mean cannot conceal changes in content preservation, dynamics, or appearance. The Wan evaluation comprises eight I2V dimensions excluding camera, not the complete I2V protocol with camera control. HunyuanVideo uses sixteen T2V dimensions. The VBench value reported in Table 1 is

$$\mathrm{VBench} = \frac{1}{m}\sum_{j=1}^{m} V_j. \tag{14}$$

No official VBench Overall normalization or weighting is applied to this descriptive mean. The original NeurIPS review exchange first discussed five selected VBench dimensions; the eight I2V and sixteen T2V dimension vectors below were supplied in the subsequent author response. Consequently, the displayed means summarize that expanded retrospective report rather than a prespecified primary task quality endpoint. The tables retain the reported dimension values, including dimensions on which an accelerated setting is below its dense reference; per video uncertainty for these dimensions is not available in the supplied record.

Table 10: Wan 2.1: eight I2V dimensions excluding camera. I2V-S/B measure input image subject/background preservation; SC/BC are subject/background consistency; MS is motion smoothness; IQ/AQ are imaging/aesthetic quality; DD is dynamic degree.

Method	I2V-S	I2V-B	SC	BC	MS	IQ	AQ	DD
Dense	.97652	.99077	.9436	.9657	.9834	.7077	.6105	.50407
TeaCache	.97641	.99063	.9438	.9657	.9840	.7067	.6095	.51220
SeaCache	.97501	.98998	.9435	.9662	.9830	.7005	.6357	.52846
SVG2	.97720	.99036	.9434	.9648	.9833	.7076	.6101	.50407
MagCache	.97558	.99064	.9362	.9621	.9813	.6949	.6260	.51000
TraceGate-F	.97515	.99271	.9412	.9670	.9846	.7029	.6487	.53659

Table 11: Wan 2.2: eight excluding camera I2V dimensions, using the same abbreviations as Table 10.

Method	I2V-S	I2V-B	SC	BC	MS	IQ	AQ	DD
Dense	.98088	.99294	.9608	.9737	.9880	.7076	.6141	.35772
TeaCache	.97953	.99225	.9603	.9727	.9876	.7064	.6125	.34959
SeaCache	.97868	.99191	.9556	.9711	.9872	.7057	.6111	.34431
SVG2	.97915	.99184	.9596	.9722	.9879	.7068	.6111	.35772
MagCache	.97280	.98848	.9482	.9670	.9872	.7035	.6110	.36992
TraceGate-F	.97852	.99143	.9585	.9711	.9880	.7035	.6080	.33333
TraceGate-Q	.98026	.99250	.9604	.9728	.9880	.7065	.6114	.36366

Table 12: HunyuanVideo T2V, dimensions 1 to 8. TF: temporal flickering; OC: overall consistency. Other abbreviations follow Table 10.

Method	SC	BC	MS	TF	DD	AQ	IQ	OC
Dense	.90880	.94630	.99080	.99614	.76389	.65990	.64790	.27204
TeaCache	.90440	.94810	.98990	.99578	.80556	.65450	.62800	.27100
MagCache	.90850	.94680	.99140	.99631	.75000	.65600	.63250	.27102
SVG2	.90918	.94651	.99111	.99617	.73611	.65712	.62802	.26982
TraceGate-F	.90610	.94540	.99110	.99628	.75000	.65350	.62540	.27152
TraceGate-Q	.90790	.94660	.99090	.99618	.81556	.65740	.63810	.27196

Table 13: HunyuanVideo T2V, dimensions 9 to 16. HA: human action; TS/AS: temporal/appearance style; Obj.: object class; Multi: multiple objects; SR: spatial relationship. Tables 12 and 13 together comprise the sixteen dimensions.

Method	HA	TS	AS	Obj.	Multi	Color	SR	Scene
Dense	.95000	.25476	.22297	.86946	.57088	.86013	.78514	.51235
TeaCache	.96000	.25535	.22307	.86737	.50915	.87112	.77074	.58360
MagCache	.94000	.25448	.22220	.86234	.58384	.87801	.77540	.54942
SVG2	.93000	.25230	.22182	.86155	.58765	.87245	.78192	.53270
TraceGate-F	.95000	.25535	.22226	.84968	.57470	.86245	.77967	.52544
TraceGate-Q	.96000	.25441	.22237	.86738	.58622	.85173	.78382	.58672

C Feedback and Component Controls

C.1 Threshold Sweep at Matched Reported Speed

The following sweep removes both topology feedback factors but keeps the inner planner, PQ configuration, protected calls, maximum reuse length, inputs, and seeds unchanged. Tightening the remaining thresholds reduces speed. Its purpose is to compare the complete controller to a controller without feedback at a comparable reported wall clock operating point, rather than to interpret fixed threshold removal as a free improvement.

Table 14: Threshold sweep without feedback on the cohort of 108 videos. The final row retains feedback. Matching is at the displayed speedup precision; no interval of equivalent timings is inferred.

Configuration	τ_r	τ_m	PSNR	SSIM	LPIPS	Speedup
No feedback, original	.20	.12	23.22	.772	.138	4.12×
No feedback, tuned 1	.16	.10	23.30	.774	.138	3.70×
No feedback, tuned 2	.13	.08	23.36	.776	.137	3.45×
No feedback, rounded speed match	.11	.07	23.38	.776	.137	3.30×
Complete feedback	.20	.12	23.51	.779	.137	3.30×

C.2 Local crossed control

Table 15 crosses two conditioning thresholds, two residual thresholds, and feedback enabled or disabled. The differences are on minus off at the *same threshold pair*. The runs with feedback are slower in every row, so these are not additional matched speed results. They describe local sensitivity and do not establish weak interactions among all seven sweep parameters.

All other planner, PQ, protection, and reuse cap settings are held fixed for this comparison.

Table 15: Local $2 \times 2 \times 2$ feedback control. Differences are on minus off at the same threshold pair; negative ΔLPIPS favors feedback.

τ_r	τ_m	Speedup off / on	ΔPSNR	ΔSSIM	ΔLPIPS
.16	.10	3.70× / 2.95×	+.28	+.007	-.003
.16	.12	3.84× / 3.08×	+.26	+.007	-.002
.20	.10	3.91× / 3.16×	+.28	+.008	-.002
.20	.12	4.12× / 3.30×	+.29	+.007	-.001

C.3 Component removal at fixed configurations

These ablations remove individual operations or feedback paths from the complete configuration. Removing the inner PQ shortlist can also change the retained pair budget, so the w/o PQ row is not the similar budget exact control of Section 4.6. Likewise, the residual and topology removals change execution decisions and are not quality comparisons at fixed latency.

Table 2 in Section 4.3 reports these configurations alongside the complete method. The row without block reuse retains the attention planner, whereas the row without topology feedback retains both the planner and residual reuse but removes the connection from planner to gate. They therefore isolate different architectural changes.

C.4 Stale map intervention details

The event screen was fixed before viewing intervention outcomes and selects at most one event per video from 108 traces. The 72 selected events cover eleven categories; the other 36 videos are not converted into positive intervention cases. Eligibility requires that conditioning and residual conditions allow reuse, topology alone vetoes it, and the current and cached route maps differ. Each target intervention holds $Q/K/V$ fixed and substitutes the cached map for the current map. Later

decisions follow the protected replay, so the intervention is not confounded by a separately adaptive downstream policy.

Table 16: Selected event stale routing controls. Intervals are the reported 95% intervals; they are not transferred to the distinct matched speed controller experiment.

Quantity	Reported result
Unchanged replay reproduces the output	72/72
Map substitution changes target block output	72/72
Change propagates to the final video	69/72
Mean target block relative L_2 difference	18.9% [17.4, 20.4]%
Protected has higher PSNR relative to dense output	54/72 selected events (75.0%)
Protected minus stale: PSNR	+0.23 [0.14, 0.32] dB
Protected minus stale: SSIM	+0.006 [0.003, 0.009]
Protected minus stale: LPIPS	-0.003 [-0.005, -0.001]
Protected minus stale: temporal PSNR	+0.19 [0.10, 0.28] dB

D WORKLOAD ACCOUNTING AND PLANNER ACTIONS

For call k, let I_k indicate stack execution, a_k its active attention pair count, and b_k the corresponding dense count. Attention density over the trajectory is

$$D_{\text{traj}} = \frac{\sum_k I_k a_k}{\sum_k b_k}. \tag{15}$$

For equal dense attention shapes, let r_{exec} be the fraction of stack calls actually executed and D_{exec} the mean attention density within those calls. Then $D_{\text{traj}} = r_{\text{exec}} D_{\text{exec}}$; unequal shapes, heads, or experts require the definition based on counts above. Guidance branches enter the same denominator through their actual calls; topology sharing is recorded separately. MoE accounting refers to execution by the experts active in the native model.

Table 17: Wan 2.1 trace accounting over 8,640 outer decisions per configuration. Removing residual control changes both execution frequency and the density of the remaining computations.

Statistic	Complete method	Without residual control
Executed stacks	4,248	3,905
Skipped stacks	4,392	4,735
Execution rate	49.17%	45.20%
Skip rate	50.83%	54.80%
Within stack attention density	17.21%	37.15%
Trajectory aligned density	8.46%	16.79%
Overall speedup	3.30×	3.59×

Table 18: Planner action counts for the same traces. Percentages use only refresh/repair/reuse as their denominator. CFG sharing is a separate operation.

Action	Complete method	Without residual control
Full refresh	38,641 (63.84%)	37,876 (68.00%)
Repair	16,657 (27.52%)	13,925 (25.00%)
Reuse	5,230 (8.64%)	3,899 (7.00%)
Three action total	60,528	55,700
CFG share	37,752	34,625

A complete arithmetic accounting would sum all actually executed operations, rather than scale attention alone:

$$F_{\text{total}} = F_{\text{init}} + \sum_k \left[F_{\text{base},k} + I_k \left(F_{\text{nonattn},k} + F_{\text{attn},k} + F_{\text{planner},k} \right) \right]. \tag{16}$$

This separates initial encoding/decoding, unavoidable operations on every call, stack work outside attention, attention, and planning. We do not replace a count of these operations across the trace with a "total FLOPs" estimate based only on density. Arithmetic also does not account for memory traffic, indexing, launch overhead, or synchronization. The present trace counts explain the changed workload composition; a component timing profile is needed for a quantitative causal decomposition of wall clock savings.

E PQ RETRIEVAL CONTROLS

PQ (Jégou et al., 2011) partitions centroid vectors into subspaces and uses quantized key representations for approximate candidate scoring. The reported kernel accumulates subspace contributions and includes a key cluster size weight in centroid importance. Exact scores are then computed for shortlisted keys. The use of exact reranking within a shortlist does not make retrieval exact over all keys.

For query cluster i with n_i^Q tokens, the expanded pair target is $n_i^Q \sum_{j \in \mathcal{M}_i} n_j^K$. Exact ranking takes the highest scoring prefix nearest the PQ target. A row can differ by an indivisible key cluster, and

Table 19: PQ controls on the cohort of 108 videos. The exact row at a similar budget matches expanded token pair workload for each query cluster as closely as indivisible key clusters permit. Exact top p uses a different coverage budget.

Selection mode	Trajectory aligned density	Speedup
PQ shortlist + exact reranking	8.4628%	3.30×
Exact ranking, near PQ pair budget	8.4555%	3.29×
Exact top p, 90% cluster score mass	14.8779%	3.01×

changed routes can affect later planner decisions. Consequently, a local budget match does not force identical overall density.

The reported approximation audit of 1,944 records gives precision 0.9893, recall 0.5372, representation cosine 0.99831, and relative L_2 difference 0.0452. Across 108 outputs, the reported similarity between PQ and exact ranking is 29.102 dB PSNR, 0.9030 SSIM, and 0.1547 LPIPS. These compare implementations rather than measure gains against the dense video reference. Precision, set recall, and representation agreement measure different properties; in particular, the recall value is not evidence of nearly complete retrieval. The overall speedups at similar budgets are close at reported precision, so PQ is treated as an optional planner implementation rather than the central empirical contribution.

F QUALITATIVE EXAMPLES

The examples retain the original rendered outputs and their configuration annotations. Labels “Ours” and “Ours-slow,” where present, identify TraceGate configurations; embedded speed labels remain as recorded rather than being replaced with aggregate table values. Dense denotes the original model’s generated reference rather than a ground truth future video. These examples illustrate appearance differences; the paired aggregate and task level measurements are reported separately.

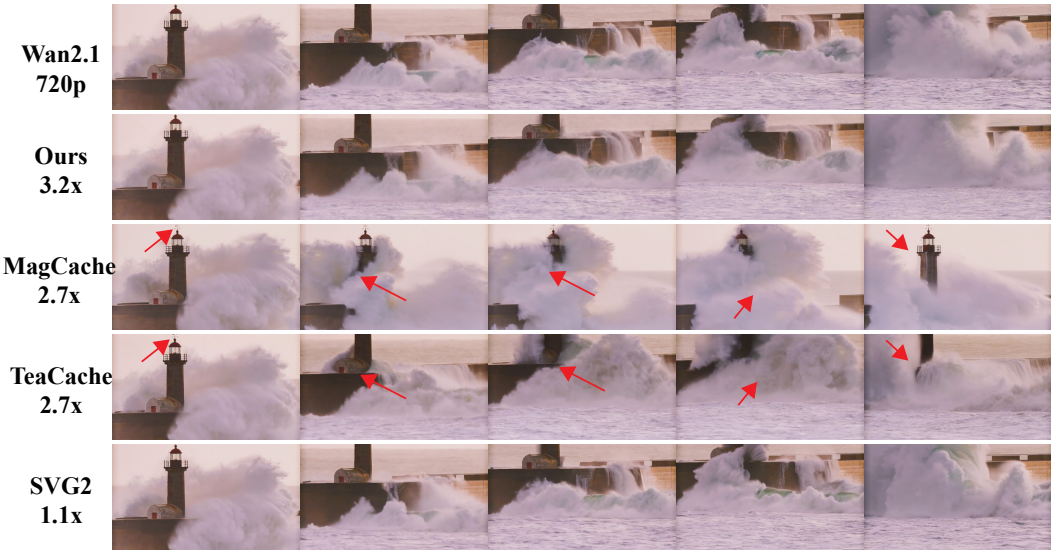

Figure 8: Wan 2.1 I2V qualitative comparison using the original generated outputs.



Figure 9: Additional Wan 2.1 I2V example.

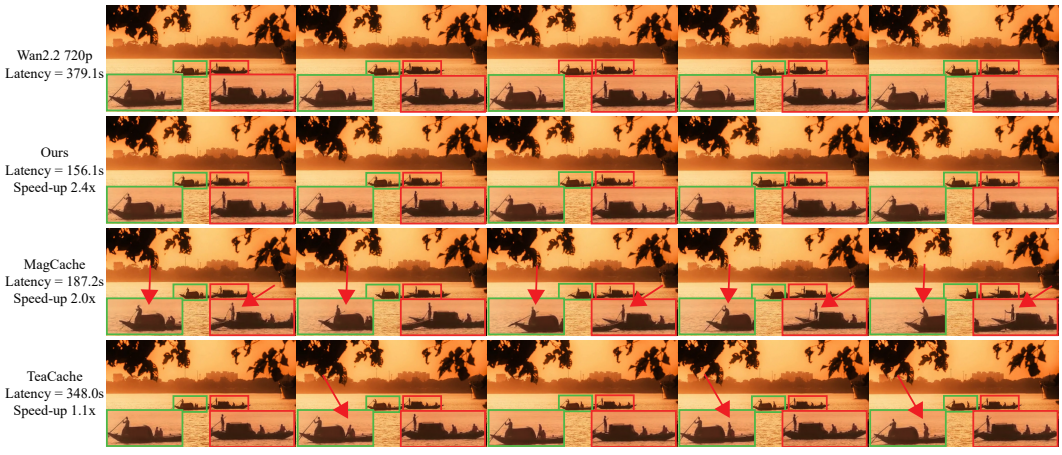

Figure 10: Wan 2.2 I2V qualitative comparison.



Figure 11: Additional Wan 2.2 I2V example.

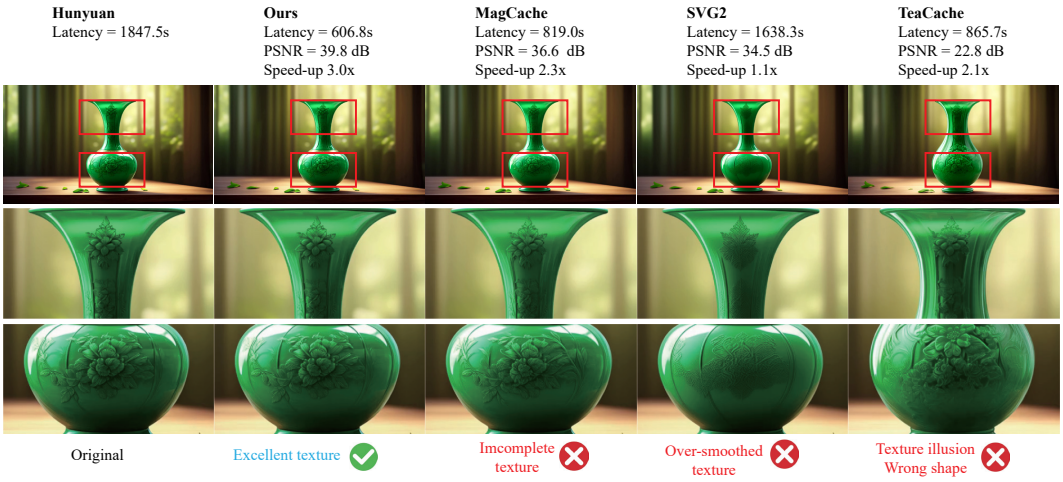

Figure 12: HunyuanVideo T2V qualitative comparison with detail crops.

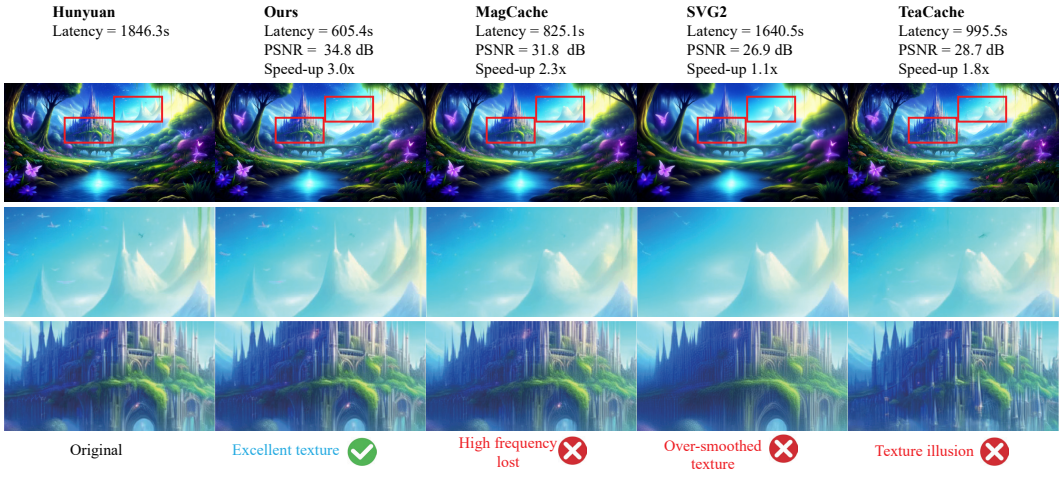

Figure 13: Additional HunyuanVideo T2V example with detail crops.