MAX-CACHE: MOTION-ALIGNED ACTIVATION REUSE WITH MISMATCH-GUIDED SELECTIVE RECOMPUTATION FOR EFFICIENT AUTOREGRESSIVE VIDEO DIFFUSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Interactive driving simulation requires video world models to generate future scenes continuously and with low latency. Existing acceleration methods reduce denoising steps, sparsify attention, or optimize key-value caches. We exploit a complementary source of redundancy: neighboring video chunks share scene content and have similar intermediate activations. Reusing these activations at unchanged token locations, however, is sensitive to ego motion, independently moving objects, parallax, and occlusion. We propose MAX-CACHE, an inference-acceleration framework with lightweight post-hoc calibration for motion-aligned activation reuse across chunks. MAX-CACHE evaluates a safety gate before either reuse mechanism. When the gate passes, it directly reuses the entire motion-aligned output at b_r, bypasses the prefix through b_r, and repairs that feature with a lightweight residual corrector calibrated post hoc from training-split teacher pairs while the pretrained generator remains frozen. Otherwise, the executed blocks use a partial token mask: low-mismatch tokens reuse warped cached outputs, while high-mismatch tokens retain fresh block updates. Conservative reuse criteria and baseline K/V-state maintenance limit error accumulation. On the 21-frame single-view nuScenes setting, the resulting configuration increases end-to-end throughput from 3.90 to 5.48 FPS (1.41×). Across both 21- and 211-frame settings, VBench, FID, and FVD remain close to the baseline, while the measured 211-frame end-to-end speedup reaches 1.30×.

Keywords: autoregressive video diffusion; driving world models; inference acceleration; motion-aligned cross-chunk reuse; token-level routing.

1 INTRODUCTION

Video world models are increasingly used as interactive simulators rather than one-shot content generators. In driving, a model must extend the visual future as the environment and control signal evolve; closed-loop evaluation then repeats this process over long rollouts and many initial conditions. Generation quality remains essential, but latency also determines how often an agent can interact with the simulator and how many scenarios can be tested. This makes efficient online generation a first-order systems requirement, not merely a convenience at deployment time.

Autoregressive video diffusion is well suited to this setting because it extends a clean history one chunk at a time. Recent causal distillation pipelines compress the denoising trajectory to a small number of steps and bring streaming generation much closer to real time (Zhao et al., 2026). If a rollout contains N chunks, however, the denoiser still executes K steps and L Transformer blocks for each new chunk. Distillation reduces K; it does not exploit the fact that successive chunks repeatedly process the same road, background, and persistent actors. As spatial resolution, causal context, and rollout length grow, this repeated block computation remains expensive.

Adjacent chunks provide an obvious source of redundancy, but not a trivial cache hit. Ego translation and yaw move scene content across token positions. Depth-dependent parallax, newly visible regions, and independent object motion then violate any single global correspondence. A cache

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.

built for the previous chunk can therefore be semantically useful while being spatially misplaced. The central question is not whether to cache an activation, but how to align it and decide where the aligned value is still trustworthy.

X-CACHE showed that compact same-step/same-block fingerprints can gate cross-chunk residual reuse (Zeng et al., 2026). On our view–temporal-factorized backbone, all 19,720 eligible fingerprint cosines fall below the documented threshold (maximum 0.778; Appendix B.1, Figure 1), showing that the original same-position similarity gate does not transfer directly. Together with the activation-level analysis in Figure 3, this motivates explicit alignment before assessing reuse reliability.

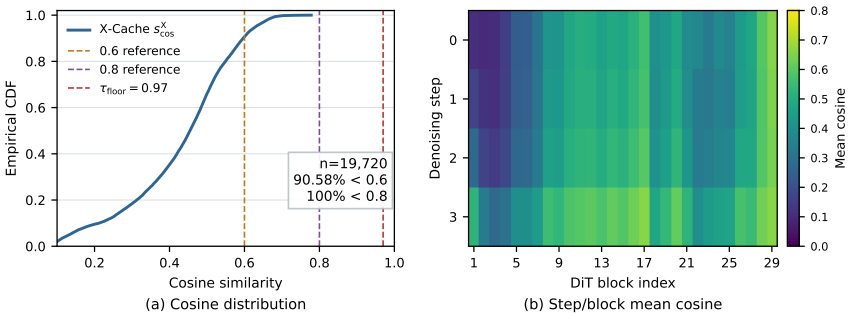


Figure 1: Compatibility diagnostic for a direct X-CACHE gate transfer on ten 211-frame nuScenes clips. (a) All 19,720 eligible same-step/same-block fingerprint cosines lie below 0.8; 90.58% are below 0.6, while the red line marks the documented threshold floor. (b) The same pattern persists across denoising steps and non-anchor blocks.

MAX-CACHE uses one safety gate and two mutually exclusive paths. A pass warps and corrects the cached output at b_r, replacing the current output and bypassing the prefix. Otherwise, the prefix executes and Mismatch-Guided Selective Recomputation retains fresh updates for high-mismatch tokens while reusing aligned outputs elsewhere; incompatible caches fall back to dense computation. Dense K/V context and the baseline's clean recache preserve the causal state. The warp-only variant requires no learned adaptation. The final configuration reported in the main tables adds a lightweight post-hoc corrector calibrated once for the target backbone; the pretrained generator remains frozen and no end-to-end generator retraining is performed. Appendix C reports its calibration and transfer settings.

Across the single-view benchmark, the corrected configuration improves throughput at both 21 and 211 frames while retaining close quality. End-to-end results establish the latency–quality operating point, and targeted diagnostics verify alignment, routing, and correction.

Our contributions are:

- **Motion-aligned cross-chunk reuse with feature correction.** We reuse an intermediate activation only after mapping it into the current chunk's coordinates, enabling prefix skipping. We further export full-compute teacher pairs from the training split and post-train a residual corrector at the reuse point while keeping the base generator frozen.
- **Mismatch-guided selective recomputation.** We turn the residual alignment error into a token-level compute allocation rule: low-mismatch tokens reuse cached outputs, while high-mismatch tokens retain fresh block updates and dense K/V history.
- **Latency–quality evaluation.** On single-view nuScenes, the corrected configuration provides 1.41× and 1.30× end-to-end speedups over Causal Forcing++ at 21 and 211 frames while retaining comparable VBench, FID, and FVD.

2 RELATED WORK

Driving world models and generative simulation. Driving-video models generate future observations from structured conditions. DriveDreamer adds traffic constraints (Wang et al., 2024a), Drive-WM couples multi-view prediction with planning (Wang et al., 2024b), and Panacea uses

BEV and panoramic representations for controllable generation (Wen et al., 2024). Epona and OmniNWM extend this direction with autoregressive generation and state–action–reward awareness (Zhang et al., 2025b; Li et al., 2026a), while DriveArena and Point as Skeleton study closed-loop or state-updated simulation (Yang et al., 2025b; Wong et al., 2026).

Autoregressive and long video diffusion. Bidirectional models such as CogVideoX and Wan jointly denoise a clip (Yang et al., 2025c; Wan et al., 2025), whereas causal models generate it sequentially. Diffusion Forcing enables causal rollouts (Chen et al., 2024), and SkyReels-V2 and MAGI-1 scale autoregression to long videos (Chen et al., 2025; Teng et al., 2025). CausVid performs few-step causal distillation (Yin et al., 2025); Self Forcing and Causal Forcing++ address training–inference mismatch and causal adaptation (Huang et al., 2026; Zhao et al., 2026). Rolling Forcing, LongLive, Self-Forcing++, and Reward Forcing further improve real-time or long-horizon rollouts (Liu et al., 2026; Yang et al., 2025a; Cui et al., 2026; Lu et al., 2026). These advances reduce sampling cost and rollout degradation; cross-chunk Transformer reuse provides a complementary acceleration axis.

Efficient video diffusion and autoregressive inference. Feature-caching methods reuse denoising activations through timestep prediction, learned layer policies, or training–inference alignment (Liu et al., 2025; Ma et al., 2024; Huang et al., 2025); sparse-attention methods instead reduce interactions with spatial–temporal, radial, or dynamic patterns (Xi et al., 2025; Li et al., 2026b; Zhang et al., 2025a). These designs mainly operate within a denoising trajectory. Autoregressive methods additionally exploit causal structure: FlowCache combines chunk-wise reuse with K/V compression (Ma et al., 2026), while X-Cache gates same-step/same-block residual reuse with compact input fingerprints (Zeng et al., 2026). Forcing-KV compresses historical K/V (Ji et al., 2026), and Light Forcing allocates sparse-attention budgets and retrieves history hierarchically (Lv et al., 2026). X-Cache is closest to our method along the cross-chunk reuse axis: it authorizes block-residual reuse from same-position fingerprint similarity, whereas we explicitly align cached activations and use the post-alignment mismatch to select whole-prefix reuse or token-level recomputation.

3 PRELIMINARIES

Autoregressive video diffusion. Let $x_{1:N}$ denote N clean video chunks and $y = (y_1, \dots, y_N)$ their chunk-wise controls. With $x_{<c}$ denoting the generated history, an autoregressive model factorizes

$$p_\theta(x_{1:N} \mid y) = \prod_{c=1}^{N} p_\theta(x_c \mid x_{<c}, y_c). \tag{1}$$

Each latent chunk contains T frames on an $H \times W$ grid and is generated in K denoising steps indexed by $k \in \{0, \dots, K-1\}$. We use a completed causal-consistency-distillation model as the base generator. MAX-CACHE leaves this pretrained generator frozen and introduces no end-to-end generator retraining; its final configuration adds a lightweight post-hoc residual calibration module at the reuse point.

Chunk-wise causal Transformer computation. Let $P = HW$ be the spatial tokens per latent frame and $Q = TP$ the current query tokens per view. Retaining R historical chunks gives $(R+1)T$ causal context frames. We use zero-based block indices $b \in \{0, \dots, L-1\}$ and token indices $i \in \{1, \dots, Q\}$. For batch size B and hidden width D, $h^k_{c,b}, u^k_{c,b} \in \mathbb{R}^{B\times Q\times D}$ denote the input and complete output of block b; in particular, $h^k_{c,0}$ is the Transformer input. We write

$$u^k_{c,b} = \mathrm{Block}_b(h^k_{c,b}) = h^k_{c,b+1}. \tag{2}$$

Token-level equations are per sample and per view, with batch and view indices suppressed unless v is shown; hence $h^k_{c,b,i}$ and $u^k_{c,b,i}$ are D-dimensional vectors. The reusable intermediate output is distinct from the persistent K/V cache, even if an implementation stores a residual and reconstructs the same output.

We use b_r for the whole-prefix reuse block and $\mathcal{B}_{\mathrm{tok}} \subseteq \{0, \dots, L-1\}$ for the blocks eligible for token-level routing. A prefix-reuse decision replaces the output of b_r and therefore bypasses blocks $0, \dots, b_r$; otherwise, the prefix is executed and selective recomputation may operate only for

$b \in \mathcal{B}_{\text{tok}}$. These symbols describe the method independently of a particular implementation; their instantiated values are given in Appendix C (*Main reuse configuration*).

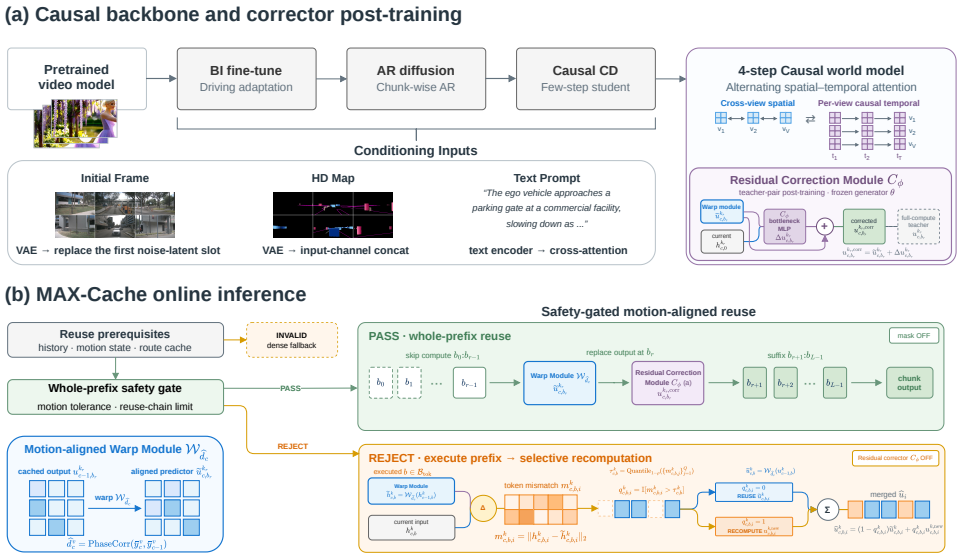


Figure 2: Overview of MAX-CACHE. (a) Causal backbone, conditioning inputs, view–temporal attention, and frozen-generator corrector calibration. (b) Online inference with mutually exclusive whole-prefix reuse and mismatch-guided selective recomputation.

4 METHOD

4.1 PROBLEM FORMULATION AND OVERVIEW

Causal distillation reduces K, yet every new chunk still traverses L Transformer blocks at each denoising step, with cost

$$\mathcal{C}_{\text{chunk}} \approx KL\,\mathcal{C}_{\text{block}}. \tag{3}$$

The remaining cost therefore lies inside the Transformer, motivating reuse of its intermediate activations; Figure 5 reports the measured operator-level decomposition.

Suppose a reuse policy $\mathcal{R}$ constructs an approximate block output $\widehat{u}^k_{c,b,i}$ from the previous chunk. A local statement of the problem is

$$\begin{aligned} \min_{\mathcal{R}} \quad & \mathcal{C}_{\text{current}}(\mathcal{R}) + \mathcal{C}_{\text{overhead}}(\mathcal{R}), \\ \text{s.t.} \quad & \mathbb{E}_{c,k,b,i}\left[\left\|\widehat{u}^k_{c,b,i} - u^k_{c,b,i}\right\|_2\right] \le \varepsilon, \end{aligned} \tag{4}$$

where the overhead includes motion estimation, warping, routing, and cache maintenance. Equation 4 requires reuse to account for both alignment error and its own overhead, while keeping approximate outputs out of persistent causal state. Figure 2 realizes this objective with two mutually exclusive reuse granularities.

Backbone and conditioning adaptation. Throughout this paper, Causal Forcing++ denotes our view–temporal-factorized implementation adapted to multi-view driving conditions. We replace joint space–time self-attention with alternating cross-view spatial attention and per-view causal temporal attention. The initial frame and per-view HD maps are VAE-encoded and combined with the video latent, while scene text is injected through cross-attention. Figure 2(a) summarizes the backbone, conditioning layout, and frozen-generator post-training of the residual corrector. These adaptation choices define the conditional causal generator and remain separate from the cross-chunk

reuse mechanisms; their tensor layout and attention complexity are given in Appendix A (*Backbone adaptation and attention complexity*).

Mutually exclusive reuse paths. Figure 2(b) summarizes the online branch order. After chunk $c-1$ is generated, the system caches intermediate activations and evaluates a whole-prefix safety gate for chunk c; its exact predicate is defined in Section 4.2. A pass invokes Motion-Aligned Activation Reuse (Section 4.2) to align and correct a cached state and bypass the prefix through b_r. Otherwise, the prefix executes and compatible caches enable Mismatch-Guided Selective Recomputation (Section 4.3); the dense block is the fallback when those prerequisites are unavailable. The gate therefore selects one reuse granularity, and state maintenance is described at the end of Section 4.3.

4.2 Motion-Aligned Activation Reuse

Road layout, background, and persistent actors create temporal redundancy in intermediate activations. Their token coordinates nevertheless move with the camera and scene, so same-coordinate copying incurs motion-dependent mismatch, as quantified in Figure 3.

On the gated whole-prefix reuse path, Motion-Aligned Activation Reuse follows a predictor–corrector design: motion warping first maps the cached activation into the current chunk coordinate system, and a lightweight residual corrector then compensates for discrepancies not captured by the estimated per-view global translation.

Motion-aligned predictor. At selected denoising step k_r and block b_r, let $\mathcal{W}_d$ warp the historical activation by per-view displacement d and then restore the original token layout. At inference time, we estimate displacement from adjacent control latents by phase correlation and form the aligned feature

$$\begin{aligned} \widehat{d}_c^{\,v} &= \mathrm{PhaseCorr}(\bar{y}_c^v, \bar{y}_{c-1}^v), \\ \widetilde{u}_{c,b_r}^{k_r} &= \mathcal{W}_{\widehat{d}_c}(u_{c-1,b_r}^{k_r}). \end{aligned} \tag{5}$$

where $\bar{y}_c^v$ is the control latent for view v, averaged over channel and time. Appendix A (*Displacement estimation*) gives the full displacement definition, subpixel refinement, and token-grid sampling procedure. This first places the historical activation in the current chunk's spatial coordinates, so reuse no longer relies on same-coordinate correspondence.

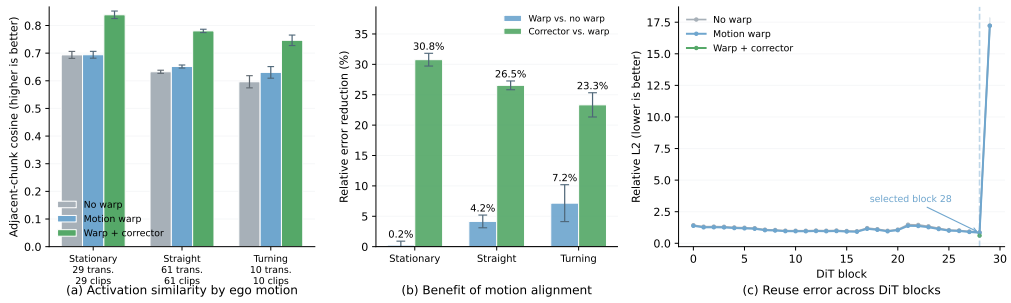


Figure 3: Activation-level evidence for motion alignment and correction. (a) Adjacent-chunk cosine similarity across motion groups. (b) Relative ℓ_2 error reductions from warping and correction. (c) Reuse error across DiT blocks, with the corrected feature in green. Statistics are aggregated at the clip level.

Figures 3(a,b) show that alignment becomes more useful as motion increases: relative ℓ_2 error falls by 4.2% and 7.2% for straight and turning transitions, respectively. Thus warping removes a component of cross-chunk coordinate shift that can be explained by a global translation; Appendix A (*Metrics and error bound*) analyzes the remaining error.

These observations motivate a conservative eligibility rule for whole-prefix reuse. The gate passes only when the previous output at (k_r, b_r) and the required pose, control-latent, and token-grid metadata are available and compatible; the absolute yaw change between adjacent chunk-start ego poses is below θ_{yaw}; and the current whole-prefix reuse streak is below $S_{\max}$. We extract yaw from the

planar rotation of the relative ego-to-global transform and measure it in radians; it is used only for this eligibility check, while the spatial warp still comes from phase correlation. Eligibility begins after the first chunk and requires a compatible cache plus successful pose and displacement estimates; a new sample clears this state. A rejected decision resets the whole-prefix streak to zero. The numerical values of θ_{yaw} and $S_{\max}$ are listed in Appendix C (*Main reuse configuration*).

When the gate passes, the method forms the warped predictor at (k_r, b_r), applies the residual correction defined below, substitutes the corrected feature for the current output of block b_r, bypasses blocks $0, \ldots, b_r$, and resumes normal computation at block $b_r + 1$. Figure 3(c) shows the measured trade-off between motion-warp error and reusable prefix length used to select b_r; the concrete block index and execution range are listed in Appendix C (*Main reuse configuration*).

Residual feature correction. Warping restores the dominant spatial correspondence between adjacent chunks, but interpolation, occlusion, local dynamics, and reuse feedback leave a feature residual. The corrector refines the warped predictor directly at the selected reuse point,

$$\begin{aligned} \Delta u^{k_r}_{c,b_r} &= C_\phi(\widetilde{u}^{k_r}_{c,b_r}, h^{k_r}_{c,0}), \\ u^{k_r,\text{corr}}_{c,b_r} &= \widetilde{u}^{k_r}_{c,b_r} + \Delta u^{k_r}_{c,b_r}. \end{aligned} \tag{6}$$

Here $h^{k_r}_{c,0}$ is the current-chunk Transformer input. With the generator frozen, C_ϕ is trained against the full-compute target $u^{k_r}_{c,b_r}$. Appendix A (*Teacher-pair construction*) specifies how the predictor, condition, and full-compute target are exported, while *Corrector objective* gives the training loss. Across the 100 evaluated transitions in Figure 3, the corrector further reduces relative ℓ_2 error over the warped predictor by 23.3–30.8% across the three motion groups, verifying residual fitting at the reuse point.

4.3 Mismatch-Guided Selective Recomputation

When the gate rejects whole-prefix reuse, prefix execution exposes a second opportunity: with a layout-compatible aligned cache, only locally unreliable regions need fresh computation, while blocks without compatible cache entries follow the dense path. For an executed block $b \in \mathcal{B}_{\text{tok}}$ with valid token-level cache, define the aligned cached output and output-side reuse error by

$$\begin{aligned} \widetilde{u}^k_{c,b} &:= \mathcal{W}_{\widehat{d}_c}(u^k_{c-1,b}), \\ e^k_{c,b,i} &:= \left\| u^k_{c,b,i} - \widetilde{u}^k_{c,b,i} \right\|_2. \end{aligned} \tag{7}$$

An ideal router would recompute the tokens with the largest e_i, but evaluating e_i itself requires the full block output and is therefore not available online. We therefore need an observable score that identifies these high-risk locations before the expensive transformation is executed.

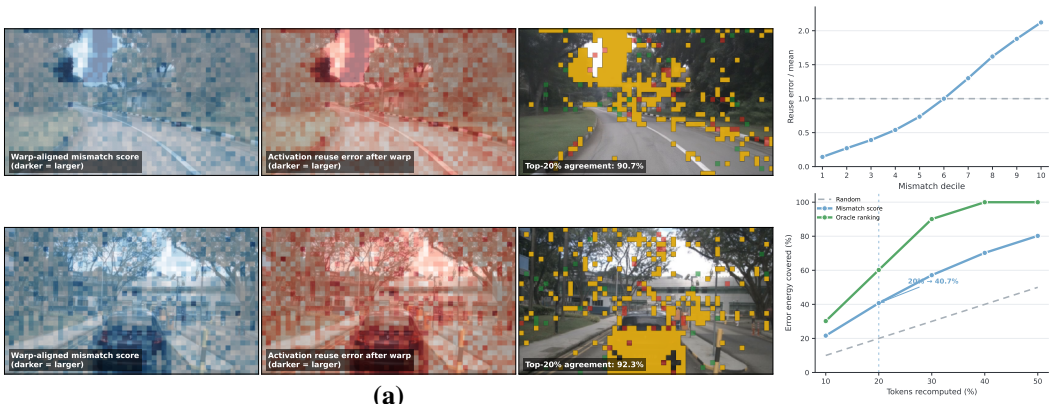
The block input is available before its expensive transformation and can be compared with the historical input under the same alignment. We define

$$\begin{aligned} \widetilde{h}^k_{c,b} &:= \mathcal{W}_{\widehat{d}_c}(h^k_{c-1,b}), \\ m^k_{c,b,i} &:= \left\| h^k_{c,b,i} - \widetilde{h}^k_{c,b,i} \right\|_2. \end{aligned} \tag{8}$$

The same warp aligns the cached input for computing m_i and the cached output used as the reuse candidate, keeping the score and candidate in the same coordinate system. Since m_i is available before the block output is formed, it can be used online; Figure 4 tests whether this input-side score localizes tokens with high output-side error.

The observed overlap and error-energy coverage motivate using m_i as the routing score: if its ranking preserves the output-side risk ranking, a fixed budget can be concentrated on high-risk locations. For a target recomputation fraction $\rho \in [0, 1]$, each executed block $b \in \mathcal{B}_{\text{tok}}$ obtains its own quantile threshold and mask:

$$\begin{aligned} \tau^k_{c,b} &:= \text{Quantile}_{1-\rho}\left(\{m^k_{c,b,j}\}_{j=1}^{Q} \right), \\ q^k_{c,b,i} &:= \mathbb{I}[m^k_{c,b,i} > \tau^k_{c,b}]. \end{aligned} \tag{9}$$



(a) **(b)**

Figure 4: Token-level routing evidence. (a) Qualitative localization of high reuse-error tokens for straight ego motion (top) and a stationary ego vehicle with a moving lead vehicle (bottom); the selected examples obtain 90.7% and 92.3% top-20% overlap. (b) Routing diagnostics on selected clips: reuse error across mismatch-ordered token deciles and reuse-error energy coverage as the recomputation ratio varies.

where $q^k_{c,b,i} = 1$ denotes recomputation and zero denotes reuse. The threshold is computed independently within each block.

The recompute mask defines a token-level routing and merge rule for executed blocks $b \in \mathcal{B}_{\text{tok}}$. Let $u^{k,\text{new}}_{c,b,i}$ be the current block output. Its final approximate output is

$$\widehat{u}^k_{c,b,i} = (1 - q^k_{c,b,i})\widetilde{u}^k_{c,b,i} + q^k_{c,b,i} u^{k,\text{new}}_{c,b,i}. \tag{10}$$

Thus high-mismatch tokens retain the current block output, whereas low-mismatch tokens take the warped cached output. This merge defines the rejected-gate path; the accepted path remains whole-prefix reuse.

At the final 60% recomputation budget, mismatch routing covers more than 80% of the reuse-error energy (Figure 4(b)). Appendix A gives the proxy-bound interpretation; Appendix B.3 compares acceleration and curvature scores.

State maintenance and safety controls. After each chunk, a dense pass on the clean denoised latent replaces the persistent K/V tensors; its cost is included in end-to-end timing. The token mask changes only block outputs, while K/V history remains dense. A new sample clears activation, pose, and streak state. Yaw- or streak-based rejection resets the whole-prefix streak and preserves compatible token caches; incompatible prerequisites select dense computation. Appendix C (*Main reuse configuration*) specifies the update.

5 Experiments

5.1 Experimental setup

We evaluate on nuScenes driving video (Caesar et al., 2020). The single-view model uses Wan2.1-T2V-1.3B and generates at 480×832. The causal base model is trained on 16 GPUs and evaluated from its 2,000-step checkpoint. Inference uses one NVIDIA H20 with batch size one. For a fixed output length, all methods share the checkpoint, prompts, HD maps, random seeds, and evaluation examples. Forcing-KV uses checkpoint-specific offline profiling, while MAX-Cache calibrates a 294,912-parameter corrector (0.023% of the generator) from 9,800 teacher pairs drawn from 700 training-split clips; generator weights remain frozen. Its 10,000-update budget is fixed before evaluation, and corrector execution is included in both Core and end-to-end timing. Appendix C reports the complete corrector and baseline settings.

We report model-core and end-to-end FPS. End-to-end timing starts with sample preparation and stops once the decoded video is available in memory; it includes conditioning, denoising, VAE encode/decode, K/V refresh, and data movement. Model-core time removes VAE encode/decode time from the inference call, and Core FPS divides the output frame count by that duration. Quality is measured with VBench Total (Huang et al., 2024), FID (Heusel et al., 2017), and FVD (Unterthiner et al., 2018). Frame lengths are never mixed in a comparison.

The baselines are Causal Forcing++, FlowCache-CD, Forcing-KV, and Light Forcing, all using the same trained view–temporal-factorized generator. FlowCache-CD denotes the cross-chunk residual-reuse adaptation detailed in Appendix C. The direct X-CACHE transfer is reported as a compatibility diagnostic in Appendix B.1 and Figure 1, rather than as a matched operating point in Table 1.

5.2 MAIN RESULTS

We test whether cross-chunk reuse improves end-to-end throughput while preserving long-horizon quality. All methods are compared jointly on speed and quality at matched output lengths.

Table 1: Single-view efficiency and quality at 480×832, compared within each output length. E2E speedup is relative to Causal Forcing++ at the same length; arrows indicate the preferred direction.

Method	Frames	Core FPS ↑	E2E FPS ↑	E2E Speedup ↑	VBench ↑	FID ↓	FVD ↓
Causal Forcing++	21	8.86	3.90	1.00×	0.774	12.11	135.03
	211	6.70	4.27	1.00×	0.780	**25.05**	126.86
FlowCache-CD	21	9.83	5.30	1.36×	0.776	12.15	134.56
	211	9.13	5.14	1.20×	0.713	65.99	129.24
Forcing-KV	21	10.05	5.38	1.38×	0.776	12.16	134.44
	211	10.15	5.46	1.28×	0.777	30.32	126.44
Light Forcing	21	**10.87**	**5.60**	**1.44**×	0.776	12.67	136.97
	211	10.34	5.50	1.29×	0.774	32.79	127.50
MAX-CACHE	21	10.71	5.48	1.41×	0.776	**12.09**	135.13
	211	**10.76**	**5.57**	**1.30**×	0.778	**25.57**	**125.74**

Table 1 shows that corrected MAX-CACHE improves end-to-end throughput over Causal Forcing++ from 3.90/4.27 to 5.48/5.57 FPS at 21/211 frames, corresponding to $1.41\times$ and $1.30\times$. VBench, FID, and FVD remain close to the baseline at both lengths, indicating that quality is maintained. At 211 frames, MAX-CACHE also provides stronger quality metrics than Light Forcing and Forcing-KV.

Runtime decomposition. Table 1 establishes the end-to-end operating point; Figure 5 explains where the model-core time is spent. Temporal attention with output projection and the FFN dominate the measured latency. Relative to Causal Forcing++, MAX-CACHE reduces both components, consistent with skipping a block prefix on the whole-reuse path and retaining fresh updates only for selected tokens on the recomputation path. The residual segment contains routing, warping, packing, cache maintenance, and dispatch work and accounts for the gap between component-level savings and end-to-end speedup.

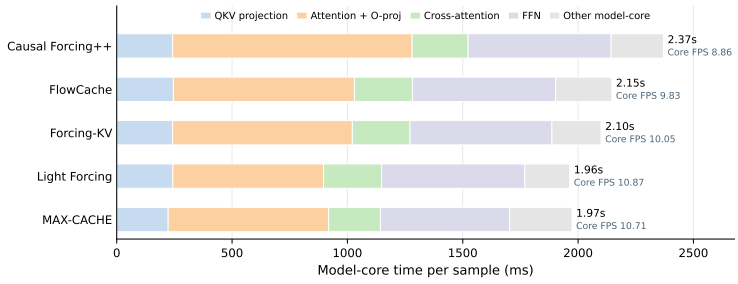


Figure 5: Operator-level model-core latency for 150 successful 21-frame samples; stacks show components and end labels report totals.



Figure 6: Paired 211-frame comparison on nuScenes scenes 0015 and 0093. Columns show 0, 8, 16, and 20 seconds; rows compare the five methods. Colored boxes mark scene-specific regions, with square crops in the final column.

Qualitative long-horizon comparison. Figure 6 complements the distributional metrics in Table 1: across both 211-frame scenes and highlighted regions, MAX-CACHE remains visually close to Causal Forcing++.

5.3 CORE COMPONENT ABLATIONS

To attribute the system-level operating point to its core mechanisms, we retain four leave-one-component-out controls in the main paper: motion alignment, feature correction, mismatch-guided token routing, and clean state refresh.

Table 2 reports 21-frame results with 150 evaluation samples. Each ablation changes only the named component. Threshold, schedule, reuse-streak, mask, and corrector-design controls are reported in Appendix B.2 (*Additional control ablations*).

Table 2: Core 21-frame component ablations (150 samples).

Control	Variant	Core FPS ↑	FID ↓	FVD ↓
Full method	MAX-CACHE (warp + corrector)	10.71	**12.09**	**135.13**
Motion alignment	No phase correlation	10.60	12.19	136.10
Feature correction	Warp only (training-free)	11.29	14.41	145.07
Token routing	No token routing	12.08	17.82	251.21
Clean refresh	No clean refresh	**13.41**	13.20	148.88

All four leave-one-component-out controls increase FID/FVD, with the largest changes caused by removing token routing or feature correction. The warp-only row is the training-free variant and isolates the learned corrector while holding the reuse and routing policy fixed; the full-method row is the final post-hoc-calibrated variant used in the main comparison.

6 CONCLUSION

Few-step autoregressive diffusion shortens the denoising trajectory but still repeats much of the Transformer for every generated chunk. MAX-CACHE first uses a safety gate to choose reuse granularity: a gate pass directly reuses the motion-aligned whole output at b_r, skips the prefix, and applies a post-trained residual corrector, whereas the alternative path executes the prefix and selectively reuses low-mismatch tokens while retaining fresh updates for high-mismatch tokens. Selected queries preserve dense K/V history. On single-view nuScenes, the resulting configuration improves end-to-end throughput from 3.90 to 5.48 FPS at 21 frames while maintaining quality; it also remains faster than the baseline with close quality at 211 frames. These results show that aligned cross-chunk activations can be reused at either block or token granularity for quality-aware acceleration.

AI USE STATEMENT

Generative AI tools were used to aid or polish writing, including improving grammar, clarity, organization, and consistency. All AI-assisted edits were reviewed and revised by the authors, who take full responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The method and routing definitions are specified in Sections 4 and 4.3, with the attention-complexity and error analyses in Appendix A. Appendix C records the backbone, conditioning inputs, block indices, routing budget, denoising steps, hardware, evaluation lengths, complete corrector post-training recipe, and per-baseline implementation, budget, backend, and operating-point selection rule. The reported 21-frame controls and 211-frame long-horizon comparison use the fixed settings described there, while the routing and component-ablation tables state their evaluation budgets and metrics. Together, these descriptions provide the information needed to reproduce the method and the reported comparisons from the identified public dataset and model configurations.

REFERENCES

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11618–11628. IEEE, 2020.

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

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

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.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

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

Yushi Huang, Zining Wang, Ruihao Gong, Jing Liu, Xinjie Zhang, Jinyang Guo, Xianglong Liu, and Jun Zhang. Harmonica: Harmonizing training and inference for better feature caching in diffusion transformer acceleration. In *ICML*, 2025.

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

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.

Bohan Li, Zhuang Ma, Dalong Du, Baorui Peng, Zhujin Liang, Zhenqiang Liu, Xianda Guo, Zheng Zhu, Chao Ma, Yueming Jin, et al. Omninwm: Omniscient driving navigation world models. In *European Conference on Computer Vision*, pp. 187–208. Springer, 2026a.

Xingyang Li, Muyang Li, Tianle Cai, Haocheng Xi, Shuo Yang, Yujun Lin, Lvmin Zhang, Songlin Yang, Jinbo Hu, Kelly Peng, et al. Radial attention: $\mathcal{O}(n \log n)$ sparse attention with energy decay for long video generation. *Advances in Neural Information Processing Systems*, 38:16822–16852, 2026b.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7353–7363. IEEE, 2025.

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

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

Chengtao Lv, Yumeng Shi, Yushi Huang, Ruihao Gong, Shen Ren, and Wenya Wang. Light forcing: Accelerating autoregressive video diffusion via sparse attention. *arXiv preprint arXiv:2602.04789*, 2026.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *Advances in Neural Information Processing Systems*, 37: 133282–133304, 2024.

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

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.

Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

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

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European conference on computer vision*, pp. 55–72. Springer, 2024a.

Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14749–14759. IEEE, 2024b.

Yuqing Wen, Yucheng Zhao, Yingfei Liu, Fan Jia, Yanhui Wang, Chong Luo, Chi Zhang, Tiancai Wang, Xiaoyan Sun, and Xiangyu Zhang. Panacea: Panoramic and controllable video generation for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6902–6912, 2024.

Songbur Wong, Xiaosong Jia, Junqi You, Bo Zhang, Pei Xu, Renqiu Xia, Yuping Qiu, Shaofeng Zhang, Zelin Zhao, Xuechao Yan, et al. Point as skeleton: Accumulated point cloud enhanced autoregressive generation for closed-loop autonomous driving simulation. *arXiv preprint arXiv:2607.06516*, 2026.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.

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

Xuemeng Yang, Licheng Wen, Tiantian Wei, Yukai Ma, Jianbiao Mei, Xin Li, Wenjie Lei, Daocheng Fu, Pinlong Cai, Min Dou, et al. Drivearena: A closed-loop generative simulation platform for autonomous driving. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26933–26943. IEEE, 2025b.

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

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

Yixiao Zeng, Jianlei Zheng, Chaoda Zheng, Shijia Chen, Mingdian Liu, Tongping Liu, Tengwei Luo, Yu Zhang, Boyang Wang, Linkun Xu, et al. X-cache: Cross-chunk block caching for few-step autoregressive world models inference. *arXiv preprint arXiv:2604.20289*, 2026.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattention: Accurate and training-free sparse attention accelerating any model inference. *arXiv preprint arXiv:2502.18137*, 2025a.

Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, et al. Epona: Autoregressive diffusion world model for autonomous driving. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27220–27230. IEEE, 2025b.

Min Zhao, Hongzhou Zhu, Kaiwen Zheng, Zihan Zhou, Bokai Yan, Xinyuan Li, Xiao Yang, Chongxuan Li, and Jun Zhu. Causal forcing++: Scalable few-step autoregressive diffusion distillation for real-time interactive video generation. *arXiv preprint arXiv:2605.15141*, 2026.

A Method Theory and Derivations

Backbone adaptation and attention complexity. Let V be the number of views, $S = P$ the number of spatial tokens per frame and view, and $T_{\mathrm{ctx}} = (R+1)T$ the causal temporal context. Cross-view spatial attention processes a sequence of length VS at each time index, whereas per-view temporal attention forms TS current queries and reads $T_{\mathrm{ctx}}S$ causal-context tokens. Thus the attention-interaction count changes from

$$\mathcal{O}((TVS)(T_{\mathrm{ctx}}VS)) \longrightarrow \mathcal{O}\big(T(VS)^2\big) + \mathcal{O}(V(TS)(T_{\mathrm{ctx}}S)) \,. \tag{11}$$

The initial-frame latent and per-view HD-map latents are arranged with the video latent before Transformer processing; scene-text embeddings enter each block through cross-attention.

Displacement estimation. At denoising step k_r and block b_r, let $d = (d^1, \ldots, d^V)$ denote per-view image-plane translations and let $\mathcal{W}_d$ warp each view independently before restoring the original token layout. For analysis, define

$$d_c^\star = \arg\min_{d \in \mathcal{D}^V} \left\| u_{c,b_r}^{k_r} - \mathcal{W}_d(u_{c-1,b_r}^{k_r}) \right\|_2, \qquad u_{c,b_r}^{k_r} = \mathcal{W}_{d_c^\star}(u_{c-1,b_r}^{k_r}) + \xi_c, \tag{12}$$

where ξ_c contains independent motion, occlusion, rotation, and depth-dependent parallax. For view v, the control latent $y_c^v \in \mathbb{R}^{B\times C_y\times T\times H_y\times W_y}$ is reduced over its channel and temporal axes. Explicitly, for batch index n and spatial location (p, q),

$$[\bar{y}_c^v]_{n,p,q} = \frac{1}{C_y T}\sum_{j=1}^{C_y}\sum_{t=1}^{T}[y_c^v]_{n,j,t,p,q}, \qquad \widehat{d}_c^v = \mathrm{PhaseCorr}(\bar{y}_c^v, \bar{y}_{c-1}^v). \tag{13}$$

The reference maps are mean-centered and Hann-windowed. The peak of the inverse normalized cross-power spectrum gives an integer shift, refined by three-point parabolic interpolation. The shift maps chunk $c-1$ to chunk c; positive vertical and horizontal values move content down and right. Before bilinear sampling, latent-pixel displacement is divided by the spatial patch size. Warping uses border padding and leaves time coordinates unchanged.

Metrics and error bound. For candidate activation a and full-compute reference b, we use

$$\mathrm{Cos}(a,b) = \frac{\langle a,b\rangle}{\max(\|a\|_2\|b\|_2,\epsilon_{\mathrm{met}})}, \qquad E_{\mathrm{rel}}(a,b) = \frac{\|a-b\|_2}{\max(\|b\|_2,\epsilon_{\mathrm{met}})}, \tag{14}$$

with $\epsilon_{\mathrm{met}} = 10^{-8}$ and no per-channel normalization. The activation decomposition yields

$$\begin{aligned}\left\|u_{c,b_r}^{k_r} - \widetilde{u}_{c,b_r}^{k_r}\right\|_2 \le \|\xi_c\|_2 \\ + \left\|\mathcal{W}_{d_c^\star}(u_{c-1,b_r}^{k_r}) - \mathcal{W}_{\widehat{d}_c}(u_{c-1,b_r}^{k_r})\right\|_2.\end{aligned} \tag{15}$$

If the warp is locally Lipschitz in displacement and the suffix $\mathcal{F}_c^{>b_r}$ is locally Lipschitz around the reuse point, then

$$\begin{aligned}\left\|\mathcal{F}_c^{>b_r}(u_{c,b_r}^{k_r}) - \mathcal{F}_c^{>b_r}(\widetilde{u}_{c,b_r}^{k_r})\right\|_2 \\ \le L_{\mathcal{F}}\left(\|\xi_c\|_2 + L_{\mathcal{W}}\|d_c^\star - \widehat{d}_c\|_2\right).\end{aligned} \tag{16}$$

This local bound characterizes error propagation around the reuse point rather than asserting global contraction.

Teacher-pair construction. For each adjacent training-split chunk pair selected for corrector post-training, the frozen generator is run without activation reuse. At the same denoising step k_r and reuse block b_r used at inference, we record the full-compute previous-chunk output $u_{c-1,b_r}^{k_r}$, the current-chunk Transformer input $h_{c,0}^{k_r}$, and the full-compute target $u_{c,b_r}^{k_r}$. The previous output is aligned with the estimated displacement to form $\widetilde{u}_{c,b_r}^{k_r} = \mathcal{W}_{\widehat{d}_c}(u_{c-1,b_r}^{k_r})$. Thus each supervised example consists of predictor and condition $(\widetilde{u}_{c,b_r}^{k_r}, h_{c,0}^{k_r})$ paired with target $u_{c,b_r}^{k_r}$. The base generator remains frozen, and the exported previous-chunk state is always full-compute and never a recursively reused activation.

Corrector objective. Let $\mathcal{D}_{\mathrm{corr}}$ denote the training-split teacher pairs. For token i, define the target-energy scale $s_{c,i} = \max(D^{-1}\|u_{c,b_r,i}^{k_r}\|_2^2, \epsilon_{\mathrm{scale}})$. We optimize

$$\begin{aligned}\mathcal{L}_{\mathrm{corr}} = \mathbb{E}_{(c,i)\sim\mathcal{D}_{\mathrm{corr}}}\Bigg[\frac{D^{-1}\left\|u_{c,b_r,i}^{k_r,\mathrm{corr}} - u_{c,b_r,i}^{k_r}\right\|_2^2}{s_{c,i}} + \lambda_{\cos}\Big(1-\cos(u_{c,b_r,i}^{k_r,\mathrm{corr}}, u_{c,b_r,i}^{k_r})\Big) \\ + \lambda_\Delta\frac{D^{-1}\|\Delta u_{c,b_r,i}^{k_r}\|_2^2}{s_{c,i}}\Bigg].\end{aligned} \tag{17}$$

We use $\epsilon_{\mathrm{scale}} = 10^{-6}$, $\lambda_{\cos} = 0.1$, and $\lambda_\Delta = 10^{-4}$.

Conditional proxy-bound interpretation for mismatch routing. Let $\mathcal{S}^k_{c,b} = \{i : q^k_{c,b,i} = 0\}$ be the reused token set. In the idealized setting where recomputed tokens recover their exact block outputs, the remaining block-output error is

$$\left\|\widehat{u}^k_{c,b} - u^k_{c,b}\right\|_2^2 = \sum_{i \in \mathcal{S}^k_{c,b}} (e^k_{c,b,i})^2. \tag{18}$$

If a nondecreasing function g locally satisfies $e_i \le g(m_i)$, then

$$\left\|\widehat{u}^k_{c,b} - u^k_{c,b}\right\|_2^2 \le \sum_{i \in \mathcal{S}^k_{c,b}} g(m^k_{c,b,i})^2. \tag{19}$$

For a fixed reuse count, removing the tokens with the largest mismatch values minimizes this conditional proxy bound. The result supplies a local ranking interpretation, while Figure 4(b) directly tests the ranking empirically.

B Additional Results and Analysis

B.1 Compatibility of Similarity-Gated Cross-Chunk Caching

X-Cache is the closest prior method along the cross-chunk reuse axis: it compares matching denoising steps and DiT blocks across chunks, then reuses a cached block residual when a compact input-fingerprint gate passes. We directly port this gate to the same causal generator with MAX-Cache disabled. For denoising step t and block b, the cosine term is

$$s^{\mathrm{X}}_{c,t,b} = \mathrm{Cos}(F(x_{c,t,b}), F(x_{\widehat{c},t,b}))\,, \qquad \widehat{c} < c, \tag{20}$$

where F contains 32 compact spatial samples and a global mean. Because a skipped block does not refresh the full-compute residual, $\widehat{c}$ can precede $c-1$. We use the documented settings: one warmup chunk, one front anchor block, no back anchor or staleness cap, $\tau_{\mathrm{floor}} = 0.97$, EMA margin 0.02, EMA coefficient 0.3, deviation threshold $\tau_{\mathrm{dev}} = 2.0$, no step-0 protection, and full-compute protection for the clean K/V-refresh pass. The local backbone encodes control in the block input before patch embedding, so the transfer uses the spatial and global-mean fingerprint channels.

The diagnostic covers ten 211-frame clips, all four denoising steps, and 29 non-anchor blocks. Excluding warmup, anchor, and clean K/V-refresh calls yields 19,720 eligible decisions, summarized in Figure 1.

The mean/median cosine is $0.429/0.455$, the 95th percentile is 0.629, and the maximum is 0.778, all below the 0.97 cosine floor. The minimum observed deviation is 17.875 versus $\tau_{\mathrm{dev}} = 2.0$, yielding no active gate decisions under the documented configuration. This diagnostic motivates explicit alignment; relaxed-threshold operating points are outside its scope.

Figures 1 and 3 provide complementary evidence: the former evaluates unaligned fingerprint stability, whereas the latter measures the benefit of explicitly aligning complete activations. Together they motivate alignment before reliability estimation.

B.2 Additional control ablations

Table 3 reports parameter sensitivity and design choices separated from the core component ablations in the main paper. To avoid presenting several independently measured rows as different "main" results, the shared Table 1 reference is stated once in the first row. Its complete setting is yaw threshold 0.20, reuse steps `[0]`, reuse-streak cap 3, dynamic mask on, and a bottleneck-64 corrector conditioned on the current feature. All remaining rows change the parameter named in the first column.

Relative to the shared reference, stricter yaw thresholds slightly reduce Core FPS, while reusing both steps increases throughput at a quality cost. Removing the dynamic mask likewise raises Core FPS but degrades FID/FVD. The reuse-streak and corrector-width alternatives remain comparatively close to the reference.

Table 3: Additional 21-frame control ablations. The first row is the single shared reference configuration from Table 1 (E2E FPS 5.48, speedup $1.41\times$, and VBench 0.776). Higher is better for Core FPS; lower is better for FID and FVD.

Control	Variant	Core FPS ↑	FID ↓	FVD ↓
Shared reference	**MAX-CACHE: yaw 0.20; steps `[0]`; streak 3; mask on; bottleneck 64 + current condition**	**10.71**	**12.09**	**135.13**
Yaw threshold (rad)	0.05	10.58	12.19	**133.95**
	0.10	10.60	**12.12**	136.77
Reuse steps	`[1]`	10.84	14.54	146.02
	`[0,1]`	**12.12**	14.55	145.22
Reuse streak	1	10.73	**12.10**	134.90
	Unlimited	10.75	12.15	136.08
Dynamic mask	Off	**11.79**	14.60	144.21
Corrector design	Bottleneck 128	**10.74**	12.19	135.53
	Without current condition	10.72	**12.14**	136.43

B.3 ADDITIONAL ROUTING COMPARISON

This comparison isolates the routing rule while Motion-Aligned Activation Reuse, denoising steps, clean K/V recache, evaluation samples, target reuse budget, and `history_len=3` are fixed; only Token Routing or its score changes.

Let $x^k_{c,b,i}$ denote the current block input token and let $\mathcal{W}_{\widehat{d}_c}$ denote the warp induced by the estimated displacement. The motion mismatch score used in the main configuration is

$$s^{\text{motion}}_{c,b,i} = \left\| x^k_{c,b,i} - \left[\mathcal{W}_{\widehat{d}_c}(x^k_{c-1,b}) \right]_i \right\|_2 . \tag{21}$$

For the three-chunk alternatives, define

$$d^1_{c,b,i} = x^k_{c-1,b,i} - x^k_{c-2,b,i}, \qquad d^2_{c,b,i} = x^k_{c,b,i} - x^k_{c-1,b,i}, \qquad a_{c,b,i} = d^2_{c,b,i} - d^1_{c,b,i}. \tag{22}$$

Acceleration and curvature scores are then

$$s^{\text{accel}}_{c,b,i} = \|a_{c,b,i}\|_2, \qquad s^{\text{curv}}_{c,b,i} = \frac{\|a_{c,b,i}\|_2}{\|d^2_{c,b,i}\|_2^2 + \epsilon}. \tag{23}$$

Here ϵ is a small numerical-stability constant. For every score, tokens above the within-block quantile are recomputed and the rest reuse the warped output. The random baseline samples a mask uniformly at the same budget, while the oracle ranks tokens by the measured output error e_i. The main paper uses the motion score; the table below reports the end-to-end comparison of these routing choices.

Table 4: Additional routing comparison at 21 frames.

Method	Core FPS ↑	E2E FPS ↑	VBench ↑	FID ↓	FVD ↓
MAX-CACHE w/o Token Routing	12.08	5.89	0.768	17.82	251.21
MAX-CACHE (Motion Routing)	11.29	5.73	**0.773**	**14.41**	**145.07**
MAX-CACHE with Acceleration Routing	**12.46**	5.97	0.767	19.64	282.92
MAX-CACHE with Curvature Routing	**12.46**	**5.98**	0.767	19.64	281.99

Against the no-routing control, mismatch routing incurs a 2.7% end-to-end cost while improving overall quality, especially FID and FVD. Acceleration and curvature routing are faster but produce lower quality across the reported metrics.

C EXPERIMENTAL CONFIGURATION

Model and evaluation configuration. Table 5 collects the model, conditioning, and evaluation settings used for the reported results.

Table 5: Model and evaluation configuration.

Name	Meaning	Value
Backbone	Base video diffusion generator	Wan2.1-T2V-1.3B
Output resolution	Generated video resolution	480×832
Base checkpoint	Causal-generator checkpoint used for evaluation	2,000 steps
Base training hardware	Hardware count used to train the causal base	16 GPUs
Inference hardware	Device used for throughput measurement	One NVIDIA H20
Inference batch size	Samples processed per inference call	1
Conditioning inputs	Inputs supplied to the conditional generator	Initial frame, per-view HD map, and scene text
Evaluation lengths	Output lengths reported in the main comparison	21 and 211 frames
Corrector data	Data source for post-training teacher pairs	Training split; generator frozen

Table 6: Residual-corrector architecture, data, and post-training cost. The 10,000-update budget is fixed before evaluation.

Item	Reported setting
Architecture	Two non-affine LayerNorm branches project the warped block-28 feature and current chunk input from 1,536 to 64 dimensions; their sum passes through SiLU and a bias-free $64 \to 1{,}536$ projection, whose output is added to the warped feature.
Input and parameters	Input width 1,536; bottleneck width 64; three bias-free linear matrices; 294,912 trainable parameters (about 0.023% of the 1.3B generator).
Teacher pairs	9,800 pairs from 700 nuScenes training clips: seven adjacent chunk transitions per clip at denoising steps 0 and 1. Each pair contains the warped previous output, current input, and full-compute current target at block 28.
Train/validation split	Clip-grouped 90/10 split with seed 0: 630 clips / 8,820 pairs for optimization and 70 clips / 980 pairs for post-training validation; no official validation clip is used.
Optimizer	AdamW, learning rate 10^{-4}, weight decay 10^{-4}; loss weights $\lambda_{\cos} = 0.1$ and $\lambda_{\Delta} = 10^{-4}$; no learning-rate schedule.
Batching	One pair is loaded at a time; 4,096 token rows are sampled from it and optimized as four minibatches of 1,024 tokens, without gradient accumulation.
Updates / epochs	The checkpoint budget is fixed in advance at 10,000 optimizer updates, during epoch 1 (0.28 passes over the optimization pairs). Main-table and evaluation-set metrics are not used for checkpoint selection.
Hardware and time	One NVIDIA H20. Full-compute teacher export takes approximately 10.4 hours; corrector optimization to 10,000 updates takes approximately 8.7 minutes (0.15 GPU-hours), for approximately 10.6 GPU-hours including export.
Transfer protocol	The generator remains frozen; a new backbone, hidden width, view layout, or reuse point uses a fresh pair export and corrector calibration, without end-to-end generator training.

Corrector post-training and adaptation scope. Table 6 reports the complete recipe used by the corrector checkpoint in all main results. Pair export and corrector optimization are timed separately because the former runs the frozen 1.3B generator whereas the latter trains only the small residual module.

Baseline implementations and operating budgets. All comparison methods use the same frozen causal checkpoint and clean K/V recache. Table 7 distinguishes local adaptations from official implementations and records the configured operating budgets and backend choices used for comparison.

Scope of FlowCache-CD. FlowCache-CD isolates the cross-chunk residual-reuse component. Integrating the full K/V-compression system additionally requires cache-range and positional-index bookkeeping plus compressed-K/V support in the clean-context recache path and attention backend; that extension is outside this evaluation.

All baseline operating points use the nearest checkpoint-compatible paper-recommended setting and remain fixed across 21 and 211 frames; the MAX-CACHE setting selected by the 21-frame controls is likewise fixed for the 211-frame evaluation. Table 7 reports each method's operating budget, while the X-CACHE transfer is retained as the gate compatibility diagnostic in Figure 1.

Main reuse configuration. Table 8 consolidates the reported MAX-CACHE configuration.

The code configuration `maxcache_v3` corresponds to Motion-Aligned Activation Reuse, while the internal term `motion mask` denotes the activation-mismatch route used by Mismatch-Guided Selective Recomputation. High-mismatch tokens retain the current block output, and low-mismatch rows receive the warped cached block output. Because corrector teacher pairs use a full-compute

Table 7: Reproduction details for the baselines in Table 1 and the X-CACHE compatibility study. The gate diagnostic uses ten 211-frame clips, while the main baseline results use 150 examples.

Method	Implementation source and backbone modification	Hyperparameters and sparsity/compression/cache budget	Kernel / backend
Causal Forcing++	Repository implementation of the cited method, adapted to the Wan2.1 driving condition interface; this is the unchanged checkpoint and dense reference path.	Four denoising steps, 30 DiT blocks, local attention window 24, and dense clean K/V recache; no sparsity or compression.	BF16 dense FlashAttention-2 on one H20; dense PyTorch linear/FFN operators.
FlowCache-CD	Local, paper-inspired adaptation of FlowCache to the few-step causal pipeline. It reuses a block residual at the same step/block across chunks; we did not port FlowCache's K/V-compression component.	Relative-L1 threshold 0.1, one warmup chunk, first and last denoising steps always dense, no reuse-streak cap.	Computed blocks use the dense baseline backend; a hit performs cached-residual addition. No custom cache-compression kernel.
Forcing-KV	Local adaptation of Forcing-KV. A checkpoint-specific offline profile partitions the 30-layer attention heads; generator weights are unchanged.	Static-head threshold 0.8; switch after generated chunk 1; one sink frame. Static heads retain one recent frame (2 stored frames including sink); dynamic heads retain one recent plus one selected frame (3 including sink), versus the dense 24-frame cap.	Separate BF16 FlashAttention-2 calls for static and dynamic head groups; PyTorch top-k patch selection.
Light Forcing	Local implementation of the cited CAG and HSA rules in the causal temporal-attention path; projections, FFNs, weights, and clean recache remain unchanged.	Target sparsity 0.70, base sparsity 0.80, keep 3 retrieved frames plus 1 sink and 1 near frame, local window 24. The first chunk is dense; 21-frame sparsities are $[0, 0.70]$, and the 211-frame schedule grows from 0.583 to 0.728 with 0.70 compute-weighted target sparsity.	FlashAttention-4 block sparse attention on SM90 H20 with 128×128 Q/K blocks.
X-CACHE	Local direct adaptation of the cited same-step/same-block fingerprint gate to the causal block-residual interface; no MAX-CACHE alignment or correction.	One warmup chunk; front anchor 1, no back anchor or staleness cap; 32 spatial samples plus global mean; $\tau = \max(0.97, \mathrm{EMA} - 0.02)$, EMA 0.3, deviation cap 2.0, step-0 protection off, clean recache protected. The separate action vector is not exposed by this backbone and its control is already included before patch embedding.	Dense FlashAttention-2 for computed blocks; cached-residual addition on a hit.

Table 8: Main MAX-CACHE configuration. Indices are zero-based.

Name	Meaning	Main value
Transformer blocks L	Total number of DiT blocks.	30
Reuse block b_r	Block whose output is replaced by the aligned and corrected cached output; blocks $0, \ldots, b_r$ are bypassed.	28; resume at 29
Token-routing blocks $\mathcal{B}_{\mathrm{tok}}$	Blocks eligible for mismatch-guided token routing.	$\{2, \ldots, 27\}$
Whole-prefix denoising step k_r	Denoising step at which whole-prefix reuse and residual correction are applied.	0
Token-routing steps	Denoising steps at which token routing is enabled.	0 and 1
Recomputation fraction ρ	Fraction of highest-mismatch tokens retaining fresh block outputs.	60%
Yaw threshold θ_{yaw}	Absolute adjacent-chunk ego-yaw limit in radians.	0.2 rad
Routing history length	History length fixed in routing comparisons.	`history_len=3`
Corrector bottleneck width	Internal bottleneck width of the residual corrector C_ϕ.	64
Maximum reuse streak $S_{\max}$	Maximum number of consecutive chunks allowed to take the whole-prefix reuse path.	3
Persistent K/V maintenance	Update rule for long-lived causal state.	Dense clean recache

previous chunk, a reuse streak of one is the strictly matched teacher-pair setting; the main value of three additionally tests rollout feedback.

D LIMITATIONS

The per-view translation model captures dominant image-plane displacement, while local mismatch routing handles rotation, depth-dependent parallax, occlusion, and non-rigid motion. More expressive alignment may further expand the reliable reuse region. Current measurements use single-view nuScenes on an H20; additional hardware, datasets, multi-view generation, and closed-loop simulation provide natural directions for broader evaluation.