KINETRA: FAST VIDEO DIFFUSION BY GENERATING THE WEIGHTS OF LATENT-FRAME TRANSITIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Video diffusion transformers generate every latent frame as a full grid of spatial tokens, so their cost grows with the number of frames times the frame size, although consecutive frames share most of their content. We propose KineTra, in which the diffusion transformer generates how latent frames change and a small network carries their content forward. KineTra splits the latent video into short segments. For each segment, the transformer generates 256 tokens that form the channel-mixing matrices of a shared executor network; the executor applies these matrices to the last latent frame of the previous segment and returns the latent frames of the current segment, whose last frame starts the next segment. The generated weights therefore encode motion and newly visible content, while appearance travels with the carried frame. A slot encoder that maps videos to these weights is trained with the executor through their own predicted boundary frames, and the transformer then learns to generate the weights with flow matching. A rank bound shows that the number of generated values must grow with the number of independently changing segments, and an error recursion bounds how errors in the carried boundary frame, the only tensor passed between segments, grow along the video. Relative to Dense on SANA-Video 2.0, LTX-Video, and CogVideoX, KineTra generates 129-frame videos 2.81–$4.07\times$ faster with VBench Quality changes of -0.05 to $+0.15$ points, and 961-frame videos up to $7.76\times$ faster on a single GPU.

1 INTRODUCTION

Video diffusion transformers (Yang et al., 2025; HaCohen et al., 2024; Kong et al., 2024; Team Wan, 2025) generate a video by denoising all its latent frames jointly. A video VAE compresses the clip into T latent frames of S spatial positions, and the transformer processes the ST positions, grouped into patch tokens, at every denoising step, so cost scales with frame count times frame size. For a one-minute, 961-frame video at 512×768, LTX-Video processes 46,464 tokens per step and takes 51 minutes on an A100 GPU, 96% of it in the transformer (Tables 4 and 18).

Most of these tokens describe content that earlier frames already contain. In Figure 2, a panning camera films a cyclist. Copying the earlier frame leaves errors across the whole image, because everything has moved; warping it with estimated optical flow removes most of it, leaving error only around the cyclist and along the border where new background enters. A later frame is thus earlier content at new positions plus a little new content, yet a dense video transformer generates every latent frame from noise as a complete grid.

Existing accelerations keep this grid: sparse attention, caching, and distillation (Xi et al., 2025; Bu et al., 2026; Yin et al., 2024b) lower the per-token cost or the step count but still process S tokens per latent frame. Stronger temporal compression yields fewer latent frames (Mahapatra et al., 2025; Yuan et al., 2025), yet each compressed token still encodes full frame appearance: raising ProMAG's compression from $4\times$ to $16\times$ costs 3.4–3.8 dB PSNR (Table 11). Content–motion models (Yu et al., 2024c; Tian et al., 2025) condition video decoding on a content frame and compressed motion latents.

We propose KineTra, in which the diffusion transformer generates how the latent frames change and a small network carries their content forward. KineTra splits the latent video into K segments of R latent frames. For each segment, the transformer generates 256 tokens that are reshaped into the channel-mixing matrices of a small shared transformer, the *executor* (Figure 3). The executor applies these matrices to the last latent frame of the previous segment and returns the segment's R

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: **KineTra samples.** A running dog, a dance turn, trail cycling, and a horse jump (frames left to right). Pose and visible content change while the scene is carried from frame to frame.

frames, the last of which starts the next segment. Appearance travels with the carried frame, and the generated weights encode motion and new content (Figure 1). Because the generated matrices change the features that the executor's shared attention uses in its next layer, one weight set controls both where content moves and what new content appears; a flow field specifies only the first, leaving the residual of Figure 2. For 961-frame LTX-Video, the transformer thus processes 8,064 tokens instead of 46,464, a $5.8\times$ reduction.

Two properties of this representation shape the design. First, at the default $R = 4$ a segment covers 32 video frames in SANA and LTX, within which an object can turn, occlude another, or reappear; the executor therefore returns a nonlinear trajectory from the carried frame to an endpoint shared with the next segment (Section 3.3). Second, each segment starts from the previous segment's output, so we train the executor and a slot encoder on their own predicted boundary frames (Section 3.4). A rank bound shows that the generated values must grow with the number of independently changing segments, so each segment receives its own weights (Section 3.5).

We compare KineTra with Dense at matched transformer, VAE, data, and budget on SANA-Video 2.0, LTX-Video, CogVideoX, and a causal Wan model, and with external accelerations on LTX and CogVideoX. At 129 frames, relative to Dense, KineTra is 2.81–$4.07\times$ faster with VBench Quality changes of -0.05 to $+0.15$ points; at 961 frames, it is $4.42\times$ faster on SANA and $7.76\times$ on LTX, where subject consistency also rises from 91.78 to 93.27. Our contributions are:

- a representation that defines each segment by generated executor weights acting on the carried latent frame;
- predicted-boundary training, with a rank bound and an error recursion;
- matched experiments on four video transformers that explain when the time is saved.

2 Related Work

Reducing the cost of video diffusion transformers. Video diffusion moved from pixel space (Ho et al., 2022; Singer et al., 2023) to the latent space of an autoencoder (Rombach et al., 2022; Blattmann et al., 2023b;a), and then to transformers over spatio-temporal patches (Peebles & Xie, 2023; Ma et al., 2025; Zheng et al., 2024; Yang et al., 2025; Kong et al., 2024; Team Wan, 2025; HaCohen et al., 2024) and flow-matching formulations (Lipman et al., 2023; Liu et al., 2023; Esser et al., 2024). Cost then grew with the number of patch tokens. Later work lowers this cost along three axes. Efficient kernels, sparse attention, and linear attention reduce the cost of each token (Dao, 2024; Zhang et al., 2025a; Xi et al., 2025; Zhang et al., 2025d; Li et al., 2025; Zhang et al., 2025b; Chen et al., 2025b; Katharopoulos et al., 2020; Xie et al., 2025; Chen et al., 2026b;a). Feature caching (Ma et al., 2024; Selvaraju et al., 2024; Zhao et al., 2025; Liu et al., 2025; Bu et al., 2026; Liu et al., 2026; Haghighi & Alahi, 2026) and distillation (Salimans & Ho, 2022; Song et al., 2023; Yin et al., 2024b;a), also for causal students (Yin et al., 2025; Huang et al., 2025), reduce how often each token is processed, at a cost in quality or retraining. Chunk-wise and streaming generation manage past context (Chen et al., 2024a; Henschel et al., 2025; Zhang et al., 2025c; Dalal et al., 2025) while predicting latent frames or chunks. With a fixed tokenizer, these accelerations still predict the latent-frame grids; this count is the common bottleneck. KineTra reduces it, replacing RS latent positions per segment by 256 weight tokens, and combines with per-token and per-step acceleration (Section 4.4).

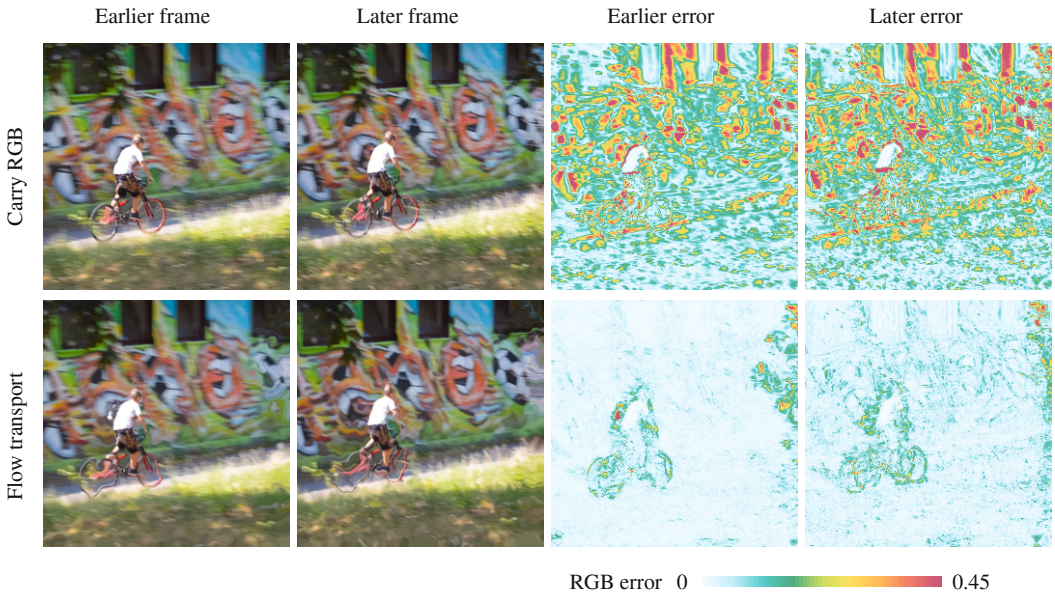

Figure 2: **Most of a frame is the previous frame moved.** DAVIS *bmx-trees*, frames 32–34, predicted from the preceding frame by copying it (top) or warping it with optical flow (bottom); maps show per-pixel RMS RGB error (0–0.45).

Compact video representations for generation. Image and video autoencoders (van den Oord et al., 2017; Esser et al., 2021; Yu et al., 2023a; 2024a; NVIDIA et al., 2025; Tang et al., 2024) compress space and time. Stronger spatial (Chen et al., 2025a) or temporal (Mahapatra et al., 2025) compression, adaptive frame rates (Yuan et al., 2025), variable-length codes (Yu et al., 2024b; Yan et al., 2025; Atanov et al., 2026), coordinate-based decoding (Jang et al., 2025; Guo et al., 2026), and generative priors (Wang et al., 2025) further shorten representations. PVDM factorizes video latents (Yu et al., 2023b); content–motion methods separate appearance from changing motion (Tulyakov et al., 2018; Yu et al., 2024c; Tian et al., 2025; Li et al., 2024). Neural representations encode signals in weights or embeddings (Chen et al., 2021; 2023; Dupont et al., 2022), and generative models sample implicit fields (Yu et al., 2022; Erkoç et al., 2023; Ren et al., 2026) or network checkpoints (Peebles et al., 2022; Wang et al., 2024). TeCoNeRV predicts patch-tubelet weights and codes differences between consecutive segments for compression (Padmanabhan et al., 2026). HyperNetworks and CondConv condition weights on inputs (Ha et al., 2017; Yang et al., 2019); fast-weight and test-time-training models update weights along a sequence (Schlag et al., 2021; Sun et al., 2025b). Spatial transformers, flow predictors, and dynamic filters transform input features or preceding frames (Jaderberg et al., 2015; Finn et al., 2016; Brabandere et al., 2016; Liu et al., 2017). KineTra samples segment-specific transition weights with a diffusion transformer and applies them to the carried boundary frame. Theorem 1 characterizes this representation's capacity under its local-rank assumptions.

3 KineTra: Generating Transition Weights

3.1 Setting

A frozen video VAE $(\mathcal{E}, \mathcal{D})$ maps an RGB video X to latent frames $Z_{0:T-1} = \mathcal{E}(X)$ with $Z_t \in \mathbb{R}^{S\times C}$, where S counts spatial positions and C channels. The condition c is text, a class label, a first image with text, or a sequence of robot actions (for VBench, UCF101, Physics-IQ, and IRASim). A dense diffusion transformer generates all T latent frames and processes their ST positions at each of its N denoising steps. We split the frames after Z_0 into K segments of R frames, so $T = 1 + KR$ ($K = 30$ for 961-frame LTX at $R = 4$). KineTra computes the last latent frame of segment k as a boundary frame B_k (Section 3.3), with B_0 the first frame. KineTra keeps the VAE, the conditioning, and the pretrained transformer body, and changes what the transformer generates: the first latent frame and per-segment weights instead of all latent frames.

3.2 Representing a Segment by the Weights of a Transition

Figure 2 suggests describing segment k as a transformation of B_{k-1}, the frame that already holds the scene; the reconstructions in Section 4.2 confirm that this holds over whole 32-frame segments. KineTra computes the frames of segment k with one network, the executor, whose parameters ψ are shared by all videos and segments; its only segment-specific parameters are weights $O_k \in \mathbb{R}^{n_w \times d_w}$ with $n_w = 256$ tokens of width $d_w = 512$. The transformer therefore generates $Q = (B_0, O_1, \ldots, O_K)$, or $S + Kn_w$ tokens instead of $S(1 + KR)$ (Figure 3a).

Executor. The executor must do the two things that Figure 2 separates: move existing content to new positions and change the features where content is new. It embeds B together with spatial positions and a local time $\tau \in [0, 1]$ into $V^0 \in \mathbb{R}^{S \times d}$ and applies $L = 8$ layers of width $d = 512$. Each layer splits the channels into $G = 16$ groups of width $d_g = 32$ and uses one generated matrix per group, $M_{k,\ell,g} = \gamma_\ell \tanh(\mathrm{reshape}_{\ell,g}\, O_k) \in \mathbb{R}^{d_g \times d_g}$, with a learned scale γ_ℓ. Writing $\mathbf{M}_{k,\ell} = \mathrm{blockdiag}(M_{k,\ell,1}, \ldots, M_{k,\ell,G})$ and $\widetilde{V}^\ell = \mathrm{LN}_\ell(V^\ell)$, layer ℓ computes

$$V^{\ell+1} = \Pi_\ell\Big(V^\ell + \mathrm{FFN}_\ell\big(\mathrm{LinAttn}_\ell(\widetilde{V}^\ell) + \widetilde{V}^\ell\,\mathbf{M}_{k,\ell}\big)\Big). \tag{1}$$

The shared linear attention $\mathrm{LinAttn}_\ell$ moves information across positions; the product $\widetilde{V}^\ell \mathbf{M}_{k,\ell}$ applies the generated channel mixing at every position; FFN_ℓ is a shared SwiGLU block; and the fixed channel permutation Π_ℓ reassigns channels to groups, so that successive block-diagonal matrices mix all channels. An output projection of V^L gives the residual $U_\psi(B, w_k, \tau) \in \mathbb{R}^{S \times C}$, where $w_k = \mathrm{vec}(O_k) \in \mathbb{R}^{D_w}$ and $D_w = n_w d_w = LGd_g^2 = 131{,}072$.

These weights suit generation: they specify an operation on content the carried frame already holds; each token has a fixed role (half the matrix of one group in one layer), so the transformer learns a distribution over a fixed layout, as over spatial positions; and $\tanh$ is 1-Lipschitz, so a weight error changes $M_{k,\ell,g}$ by at most $|\gamma_\ell|$ times that error (Appendices A.1 and B.2).

3.3 A Segment Trajectory with a Shared Endpoint

One weight set must produce R latent frames under two requirements: adjacent segments must meet at a common frame, so that one tensor carries the scene forward, and the frames between two boundaries must show the events inside the segment. Linear interpolation between B_{k-1} and B_k meets the first requirement and fails the second: in Figure 4, interpolating a horse jump between frames 0 and 16 superimposes the two poses and loses the motion between them. KineTra therefore defines the endpoint map $F(B, w) = B + U_\psi(B, w, 1)$ and the trajectory

$$H(B, w, \tau) = B + \tau\big(F(B, w) - B\big) + \tau(1 - \tau)\, U_\psi(B, w, \tau). \tag{2}$$

The first two terms fix the ends, $H(B, w, 0) = B$ and $H(B, w, 1) = F(B, w)$, and the last term vanishes at both ends and describes the path between them. Segment k returns $\widehat{Z}_{(k-1)R+j} = H(B_{k-1}, w_k, j/R)$ for $j = 1, \ldots, R - 1$ and its endpoint $B_k = F(B_{k-1}, w_k) = \widehat{Z}_{kR}$, computed once. B_k is thus both the last output frame of segment k and the input of segment $k + 1$, the only quantity passed between segments.

3.4 Training on the Boundary Frames Used at Generation

Stage 1: learning the weights through predicted boundaries. At generation, segment k receives the boundary B_{k-1} produced by the executor, which differs from the VAE latent $Z_{(k-1)R}$. Weights fitted to true latents would meet inputs at generation that they never saw in training, the train–test mismatch of recurrent predictors (Bengio et al., 2015; Huang et al., 2025), so we train on the executor's own boundaries. Starting from $B_0 = Z_0$, a slot encoder computes $O_k = E_\phi(B_{k-1}, Z_{(k-1)R+1:kR})$ from the predicted boundary and the target frames, the executor returns $\widehat{Z}$ for the segment and B_k, and we minimize $\mathcal{L}_{\mathrm{rep}} = \|\widehat{Z} - Z\|_F^2/(TSC)$ with gradients through every predicted boundary. The encoder sees the target frames only to construct training weights; at generation the transformer supplies them.

Stage 2: generating the weights. We then freeze (ϕ, ψ), so that the transformer learns a fixed target distribution, and encode each training video with the same rollout into $\mathbf{q} = \mathrm{vec}(Q) \in \mathbb{R}^{D_Q}$,

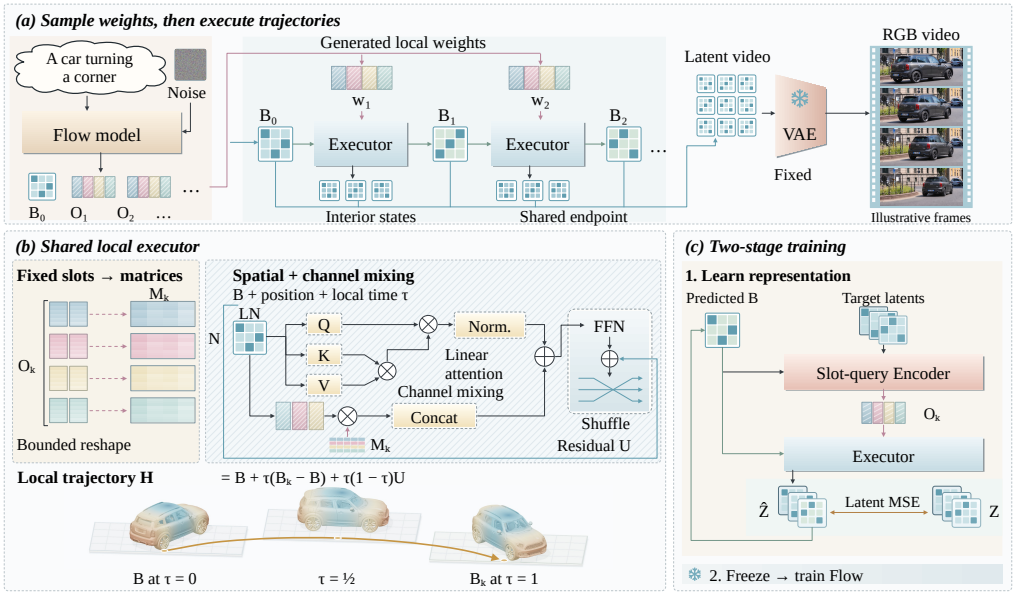

Figure 3: **KineTra.** (a) The diffusion transformer (flow model) generates the first latent frame B_0 and weight tokens O_k per segment; the executor applies O_k to the carried frame B_{k-1}, returns the interior frames, and passes the endpoint B_k on; the frozen VAE decodes. (b) Weight tokens become bounded 32×32 channel matrices $M_{k,\ell,g}$; shared linear attention moves content across positions, and the generated matrices change features at each position. (c) Two-stage training.

$D_Q = SC + KD_w$. A binary mask $\mathbf{m}$ marks the generated coordinates: all of them for text-to-video, and only the weights when B_0 is given. With flow time $s \sim \mathcal{U}[0,1]$, noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{q}_s = \mathbf{q} + s\,\mathbf{m} \odot (\boldsymbol{\epsilon} - \mathbf{q})$, the rectified-flow objective (Lipman et al., 2023; Liu et al., 2023) is

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{\mathbf{q},c,s,\boldsymbol{\epsilon}}\Big[\big\|\mathbf{m} \odot \big(v_\theta(\mathbf{q}_s, s, c) - (\boldsymbol{\epsilon} - \mathbf{q})\big)\big\|_2^2 \,/\, \|\mathbf{m}\|_1\Big]. \tag{3}$$

The pretrained transformer body receives new input and output projections for frame and weight tokens, and the position of each weight token encodes its segment, layer, group, and matrix half. Sampling integrates v_θ from $s = 1$ to $s = 0$ with the observed coordinates fixed, executes the segments in order, and decodes all frames with $\mathcal{D}$. Appendix A gives the complete algorithm.

3.5 When the Representation Saves Computation and What It Must Hold

Cost. Let κ_{tr} be the transformer cost per token and step and κ_{ex} the executor cost per position and output frame. Counting the token-linear work $\mathcal{W}$ before native patching,

$$\frac{\mathcal{W}_{\text{KineTra}}}{\mathcal{W}_{\text{dense}}} = \frac{N\kappa_{\text{tr}}(S + Kn_w) + KRS\kappa_{\text{ex}}}{N\kappa_{\text{tr}}S(1 + KR)} \xrightarrow{K\to\infty} \frac{n_w}{RS} + \frac{\kappa_{\text{ex}}}{N\kappa_{\text{tr}}}. \tag{4}$$

For LTX at 512×768 ($S = 384$) and $R = 4$, the first term is $1/6$. The second term is small for a large transformer run for many steps and grows when distillation shrinks N; when native patching already gives the dense model as few tokens as KineTra, only this executor term remains (Section 4.4). Full attention, quadratic in the token count, enlarges the saving (Appendix B.3).

Capacity. One weight set shared by the whole video would have a fixed size; the following bound shows why this fails when segments change independently.

Theorem 1 (Code size for independent segments)**.** *Fix B_0 and the executor, and let segment k's weights w_k depend smoothly on local coordinates $u_1, \dots, u_k$. Let Φ_k stack the frames of segment k, $\Phi = (\Phi_1, \dots, \Phi_K)$, and $r_k = \operatorname{rank} \mathrm{D}_{u_k}\Phi_k(0)$. Any smooth encoder–decoder pair that reproduces Φ exactly near 0 through an n-dimensional code satisfies*

$$n \;\geq\; \operatorname{rank} \mathrm{D}\Phi(0) \;\geq\; \textstyle\sum_{k=1}^{K} r_k. \tag{5}$$

Because segment k cannot affect earlier frames, $\mathrm{D}\Phi(0)$ is block lower triangular and every segment adds its own r_k directions, so the code must grow with the number of independently changing segments; KineTra adds D_w coordinates per segment (proof in Appendix B.1).

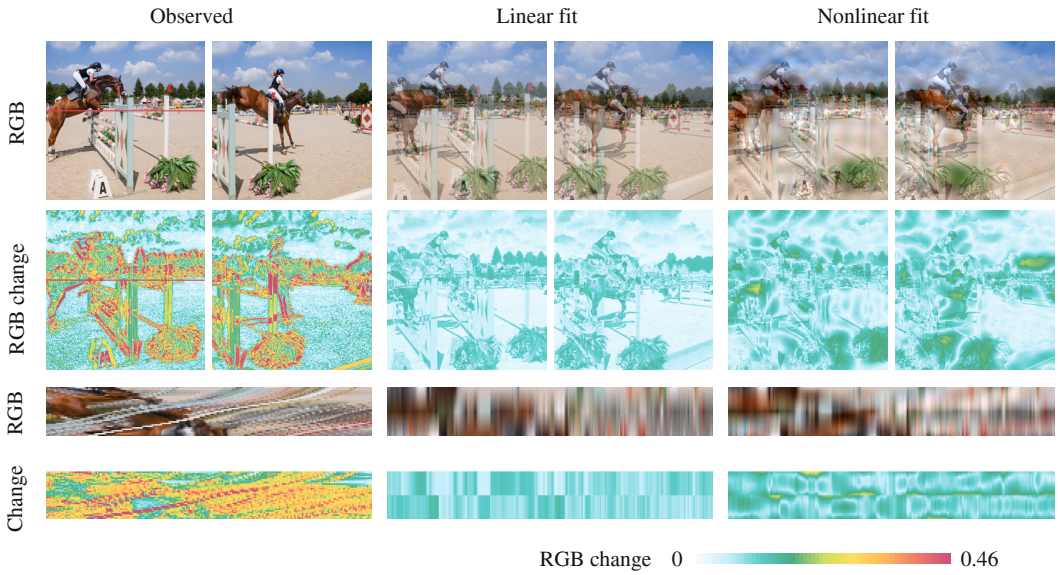

Figure 4: **The frames between two endpoints carry their own motion.** DAVIS *horsejump-high*, endpoints shared at frames 0, 16, and 32. Observed RGB versus linear interpolation and an endpoint-vanishing interior fitted with 16×16 DCT modes; rows show frames, adjacent-frame change, and a space–time strip.

Error propagation. Later segments receive an error only through B_k, the only tensor passed between them.

Theorem 2 (Boundary error recursion)**.** *For fixed reference and perturbed weight sequences, let e_k be the distance between their boundary frames B_k, ω_k the distance between their weights w_k, and ν_k a bound on numerical error. With path gains α_k of F in the frame and β_k in the weights,*

$$e_k \;\le\; \alpha_k\, e_{k-1} + \beta_k\, \omega_k + \nu_k. \tag{6}$$

An interior-frame error affects only the decoded video, but an error in B_k enters all later segments, scaled by products of later gains α_j. R sets the boundary count and the gains (Appendix B.2).

4 Experiments

Matched setup. Per model family, Dense and KineTra share the pretrained transformer body, frozen VAE, conditioning, flow objective, data, and total training budget, of which KineTra spends one quarter on Stage 1. We use SANA-Video 2.0 5B (Chen et al., 2026a), LTX-Video 13B 0.9.8-dev (HaCohen et al., 2024), CogVideoX1.5-5B (Yang et al., 2025), and a causal model from Self Forcing's Wan2.1-T2V-1.3B (Huang et al., 2025; Team Wan, 2025), with $R = 4$, 50 Euler steps, and guidance one. Each trained setting uses three seeds; accelerations reuse these checkpoints. Tables report means; intervals use 2,000 paired cluster-bootstrap resamples. Times are wall-clock on A100-SXM 80GB GPUs, from conditioning to saved video. Appendices A and D–G give all protocols.

4.1 How Much Time Does KineTra Save, and at What Quality?

At 129 frames, KineTra reduces complete-video time from 103.9 to 37.0 seconds on SANA, from 237.4 to 69.0 on LTX, and from 635.2 to 156.0 on CogVideoX, speedups of $2.81\times$, $3.44\times$, and $4.07\times$ (Table 1). VBench Quality changes by $+0.15$, -0.02, and -0.05 points, Semantic and VMBench motion scores by at most 0.17 (Huang et al., 2024; Ling et al., 2025), and peak memory falls by 23–43% because the transformer holds fewer token activations. The speedup is smallest on SANA-Video, whose attention is mostly linear, and larger on LTX and CogVideoX with full attention (Appendix A.2). The matched Dense blocks also include D2Cache and SenCache on LTX, and SenCache and Sparse-vDiT on CogVideoX, at conservative/faster (C/F) workpoints (Appendix G.3). KineTra is $1.52\times$ and $1.91\times$ faster than the fastest listed external workpoint on LTX and CogVideoX. Sparse-vDiT C has higher CogVideoX Quality (84.32 versus 84.18), at 422.6 versus 156.0 seconds.

Table 1: **Text-to-video: matched and external comparisons.** Complete-video means on VBench and VMBench. Dense and KineTra share training budgets; external accelerations reuse Dense weights. C/F settings are defined in Appendix G.3. Native VideoFlexTok (Atanov et al., 2026), DLFR-Gen (Yuan et al., 2025), and ProMAG (Mahapatra et al., 2025) retain their formats and samplers. Bold marks the best mean within a matched body or the ProMAG pair; red additionally marks LTX optima. Blue identifies KineTra.

Method	Output format			VBench					VMBench	Pipeline	
	Frames	$H \times W$	fps	Quality ↑	Sem. ↑	Subj. ↑	Backg. ↑	Dyn. ↑	Avg. ↑	Sec. ↓	GiB ↓
SANA-Video 2.0 (Chen et al., 2026a) / 5B; matched 129-frame outputs											
Dense	129	512 × 768	16	85.37	82.24	**96.42**	97.73	71.30	71.45	103.9	23.71
KineTra	129	512 × 768	16	**85.52**	**82.26**	96.25	**97.83**	**71.94**	**71.49**	**37.0**	**17.62**
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev; matched 129-frame outputs											
Dense	129	512 × 768	16	83.28	79.36	95.18	**97.01**	63.33	**69.02**	237.4	46.20
DiCache $\delta = 0.1$ (Bu et al., 2026)	129	512 × 768	16	83.21	79.25	95.22	96.83	63.24	68.76	164.2	47.05
DiCache $\delta = 0.2$ (Bu et al., 2026)	129	512 × 768	16	82.95	79.06	95.07	96.50	62.96	68.55	126.5	47.00
D2Cache C (Liu et al., 2026)	129	512 × 768	16	83.29	79.33	95.15	96.95	**63.43**	68.88	145.8	47.36
D2Cache F (Liu et al., 2026)	129	512 × 768	16	82.99	79.19	95.02	96.73	62.50	68.70	104.6	47.42
SenCache C (Haghighi & Alahi, 2026)	129	512 × 768	16	**83.32**	79.37	95.24	96.91	**63.43**	68.94	151.2	46.58
SenCache F (Haghighi & Alahi, 2026)	129	512 × 768	16	82.99	79.10	95.10	96.69	62.50	68.78	119.8	46.61
KineTra	129	512 × 768	16	83.26	**79.53**	**95.29**	96.79	62.78	68.92	**69.0**	**35.74**
CogVideoX1.5 (Yang et al., 2025) / 5B; matched 129-frame outputs											
Dense	129	512 × 768	16	84.23	**80.30**	**95.51**	**97.36**	67.04	**68.37**	635.2	41.86
SenCache C (Haghighi & Alahi, 2026)	129	512 × 768	16	84.23	80.27	95.46	97.30	66.67	68.28	372.8	42.57
SenCache F (Haghighi & Alahi, 2026)	129	512 × 768	16	83.84	80.08	95.30	97.12	66.20	68.06	297.5	42.63
Sparse-vDiT C (Chen et al., 2025b)	129	512 × 768	16	**84.32**	80.28	95.50	97.31	**67.13**	68.36	422.6	39.14
Sparse-vDiT F (Chen et al., 2025b)	129	512 × 768	16	84.12	80.20	95.37	97.20	66.67	68.20	382.1	38.29
KineTra	129	512 × 768	16	84.18	80.22	95.37	97.30	66.57	68.34	**156.0**	**23.92**
External native systems at their own output formats											
VideoFlexTok (Atanov et al., 2026)	17	256 × 256	4	78.87	75.31	92.49	95.81	47.87	66.01	114.7	28.49
DLFR-Gen (Yuan et al., 2025)	97	544 × 960	24	84.43	80.32	95.26	97.09	69.35	70.06	390.0	57.07
ProMAG (Mahapatra et al., 2025) 4×	68	192 × 360	24	**84.35**	**80.65**	**95.52**	**97.32**	**68.33**	**70.53**	160.4	34.87
ProMAG (Mahapatra et al., 2025) 16×	68	192 × 360	24	83.11	78.89	94.77	96.71	65.56	68.75	**82.8**	**32.75**

Table 2: **Long-video matched comparison across four bodies.** All outputs have 961 frames, 512×768, and 16 fps. Flicker denotes the official temporal-flickering score (higher is better). Memory is the maximum allocated GiB per GPU; GPUs counts devices per video. CogVideoX Dense and its accelerations, and causal Dense, use four GPUs versus one for KineTra. C/F settings are in Appendix G.3. All values are means; bold is within each body.

Method	VBench-Long							Complete pipeline		
	Quality ↑	Sem. ↑	Subj. ↑	Backg. ↑	Flicker ↑	Smooth ↑	Dyn. ↑	Sec. ↓	GiB ↓	GPUs
SANA-Video 2.0 (Chen et al., 2026a) / 5B										
Dense	**83.31**	79.82	93.41	**96.21**	**98.36**	**97.66**	68.37	1108.2	35.79	1
KineTra	83.21	**79.87**	**93.42**	96.05	98.25	97.53	**68.60**	**250.8**	**23.43**	1
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev										
Dense	81.38	**77.28**	91.78	95.43	**98.12**	97.18	60.37	3060.9	67.13	1
D2Cache C (Liu et al., 2026)	81.22	77.19	91.70	95.37	98.05	97.05	60.38	1678.2	71.32	1
D2Cache F (Liu et al., 2026)	80.91	76.87	91.54	95.18	97.89	96.90	59.51	1196.4	71.41	1
SenCache C (Haghighi & Alahi, 2026)	81.32	77.24	91.80	95.36	98.10	97.10	**60.66**	1746.8	68.27	1
SenCache F (Haghighi & Alahi, 2026)	80.99	76.99	91.62	95.22	97.95	96.91	59.77	1421.7	68.35	1
KineTra	**81.92**	77.27	**93.27**	**96.07**	97.98	**97.57**	60.63	**394.6**	**43.52**	1
CogVideoX1.5 (Yang et al., 2025) / 5B										
Dense	**81.99**	77.41	**92.25**	**95.63**	98.17	**97.31**	63.63	6347.2	69.08	4
SenCache C (Haghighi & Alahi, 2026)	81.92	77.36	92.23	95.60	98.13	97.27	63.66	3818.4	70.14	4
SenCache F (Haghighi & Alahi, 2026)	81.61	77.12	92.06	95.42	97.92	97.04	63.65	3164.8	70.22	4
Sparse-vDiT C (Chen et al., 2025b)	81.96	77.39	92.24	95.58	98.15	97.26	**63.95**	3726.1	65.81	4
Sparse-vDiT F (Chen et al., 2025b)	81.78	77.28	92.09	95.49	98.01	97.15	63.40	3372.6	64.49	4
KineTra	81.93	**77.52**	92.18	95.62	**98.28**	97.23	63.37	**993.5**	**46.13**	1
Self Forcing (Huang et al., 2025; Team Wan, 2025) / Wan2.1-T2V / 1.3B										
Dense	**80.92**	75.81	91.10	**94.81**	**97.73**	**97.06**	**67.27**	7275.7	62.50	4
KineTra	80.76	**75.90**	**91.47**	94.69	97.67	96.78	67.24	**491.1**	**38.04**	1

Longer videos enlarge the saving (Table 2): at 961 frames, the single-GPU speedup is $4.42\times$ on SANA and $7.76\times$ on LTX, where the transformer phase alone becomes $11.2\times$ faster for a $5.8\times$ token reduction because full attention costs more than linearly in the token count (Table 18, Appendix B.3). Dense CogVideoX and the dense causal model need four GPUs here, KineTra one. Against D2Cache F, long LTX takes 394.6 versus 1196.4 seconds ($3.03\times$, one GPU each). CogVideoX takes 993.5 versus 3164.8 seconds for SenCache F ($3.19\times$ wall-clock ratio, one versus four GPUs).

On long LTX videos, KineTra raises subject consistency from 91.78 to 93.27 (95% CI $[0.87, 2.11]$) and background consistency from 95.43 to 96.07 at every clip start (Figure 5), while dynamic degree

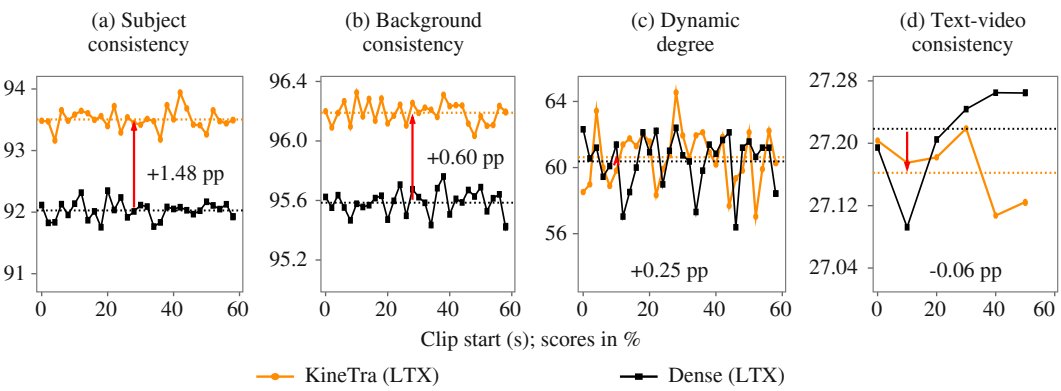

Figure 5: **The consistency gain holds over the whole minute.** LTX, 961 frames, three seeds; scores per two-second (a–c) or ten-second (d) clip start, with means (dotted) and differences (arrows).

is 60.37 versus 60.63: motion is kept while the carried frame copies appearance. On other bodies, long-video Quality changes by -0.16 to -0.06 (Table 5). KineTra is also within 0.13 of Dense on Physics-IQ (Motamed et al., 2026) at 2.8–3.9× speed, with no detectable MovieGen preference gap (Polyak et al., 2024) and close T2V-CompBench scores (Sun et al., 2025a) (Appendix D).

4.2 What Does Each Part of the Representation Contribute?

Information kept by the weights. With encoded weights, KineTra reconstructs within 0.17–0.40 dB PSNR of its frozen VAE on UCF101, Kinetics-600, MSR-VTT, and DAVIS for all three bodies (Table 11), and generated weights match dense latents in class-conditional generation: UCF101 gFVD 54.2–59.9, against 53.2–61.9 for Dense (Table 14).

Which design choice supplies which information. Each control in Table 3 removes one design choice with matched updates and data order, and the removals hurt different quantities. Feeding B_0 to every segment instead of the latest boundary causes the largest loss, 3.20 dB PSNR on SANA and 2.86 dB on LTX: every segment must then rebuild the changes accumulated since the start. Training on true boundaries lowers SANA PSNR from 34.06 to 31.98 dB and Quality from 85.52 to 83.50, the cost of the train–test mismatch that Stage 1 removes. A linear interior loses the least PSNR, 1.41 dB on SANA, but the most dynamic binding, 13.00 to 9.05: PSNR is dominated by appearance, which the carried frame preserves, while dynamic binding scores attributes that change during the video, which only the interior term expresses (Figure 4). LTX shows the same ordering.

Table 3: **Component removals.** DAVIS reconstruction, VBench Quality, and dynamic binding on T2V-CompBench (10^{-2} units). Updates and source order are matched. Bold is within each body.

Variant	DAVIS			VBench	Comp.
	PSNR ↑	LPIPS ↓	rFVD ↓	Qual. ↑	Bind. ↑
SANA-Video 2.0 (Chen et al., 2026a) / 5B					
KineTra	**34.06**	**3.2%**	**105.6**	**85.52**	**13.00**
Linear interior	32.65	4.7%	151.5	84.54	9.05
Initial state only	30.86	7.4%	232.2	83.34	9.94
Teacher boundaries	31.98	5.8%	187.9	83.50	10.75
Tied segment code	32.05	5.6%	170.6	83.93	12.20
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev					
KineTra	**32.81**	**4.0%**	**130.4**	**83.26**	**8.99**
Linear interior	31.80	5.2%	168.0	82.67	6.59
Initial state only	29.95	7.8%	249.6	81.71	6.91
Teacher boundaries	31.08	6.6%	213.2	81.71	7.52
Tied segment code	31.08	6.1%	179.1	81.85	8.09

Tying one weight set across segments tests Theorem 1: it lowers SANA Quality by 1.59 at $K = 4$, and LTX subject consistency from 93.27 to 83.19 at $K = 30$; the loss grows with the number of segments that one fixed-size code must describe.

Segment length. R sets how much change one weight set must describe and how many boundaries an error passes. For long LTX videos, $R = 2, 4, 8$ take 604.9, 394.6, and 285.7 seconds with subject consistency 92.45, 93.27, and 91.12 (Figure 6). $R = 8$ saves 27.6% of the time, but one weight set must describe 64 video frames, and with a fixed interior budget the error between endpoints grows with segment length (Figure 7); DAVIS PSNR drops from 32.81 to 32.22 dB. $R = 2$ doubles the boundaries to 60, adds 53% time, and still lowers consistency by 0.82. Per-segment capacity (Theorem 1) thus limits long segments, and the error recursion (Theorem 2) penalizes many short ones; width trades code size against executor cost (Appendix E.3). The default $R = 4$ lies between these two limits and gives the highest consistency of the sweep.

4.3 How Do Errors Pass Between Segments?

Theorem 2 states that later segments see an upstream error only through the fed boundary frame. We test this with fixed generated weights, replaying the rollout while replacing, at every internal boundary, either the frame fed to the next segment (state) or the frame returned to the VAE (output) by its reference value (Appendix F). Replacing the state lowers raw latent MSE on Physics-IQ from 0.247 to 0.204 for LTX and raises the Physics-IQ Original score from 21.06 to 25.57; SANA and CogVideoX gain 5.1 and 5.8 points. Replacing only the output leaves every later executor state, and hence latent MSE, unchanged, yet raises Original by 0.3 to 1.0 points through decoding, since the VAE decodes neighboring frames jointly. The two effects add: their interaction intervals include zero. Accuracy is thus decided at the carried boundary, whose error enters every later segment (Appendix B.4).

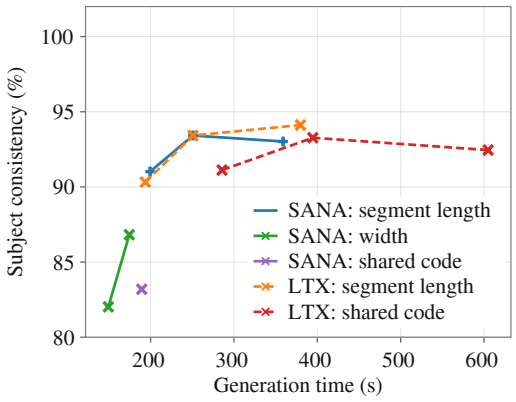

Figure 6: **Segment length and width trade time for consistency.** 961 frames, three seeds; lines join $R = 2, 4, 8$ (width 512), SANA widths 256/512/1024, or shared-code widths ($R = 4$).

4.4 What Limits the Saving?

Unchanged token count. In IRASim action-conditioned prediction (Zhu et al., 2025) on RT-1, Bridge, and Language Table, native patching gives both representations the same token count, so the executor only adds work: KineTra is 22.9–35.3% slower, with PSNR within 0.06 dB (Table 17). As Equation (4) states, the saving comes from the transformer tokens that the weights replace.

Combination with other accelerations. We apply DiCache, D2Cache, SenCache, sparse attention (Zhang et al., 2025b), four-step distribution-matching distillation (Yin et al., 2024b;a), and VAE compilation separately to both 961-frame LTX models (Table 19). KineTra is faster in every case: D2Cache F takes 269.4 versus 1196.4 seconds ($4.44\times$) at Quality 81.48 versus 80.91. With distillation it is $19.7\times$ faster than 50-step Dense, and the VAE takes 113.4 of its 155.4 seconds (Table 18).

5 Conclusion

KineTra has a video diffusion transformer generate, per segment, the weights of a small executor acting on the carried latent frame, so the transformer generates $S+256K$ tokens instead of $S(1+KR)$ latent positions. Relative to Dense, text-to-video generation is 2.81–$4.07\times$ faster at 129 frames and 4.42–$7.76\times$ at 961 frames on one GPU, with small VBench Quality changes. As the carried boundary frame sets the accuracy, more accurate boundaries and faster decoders are the next steps.

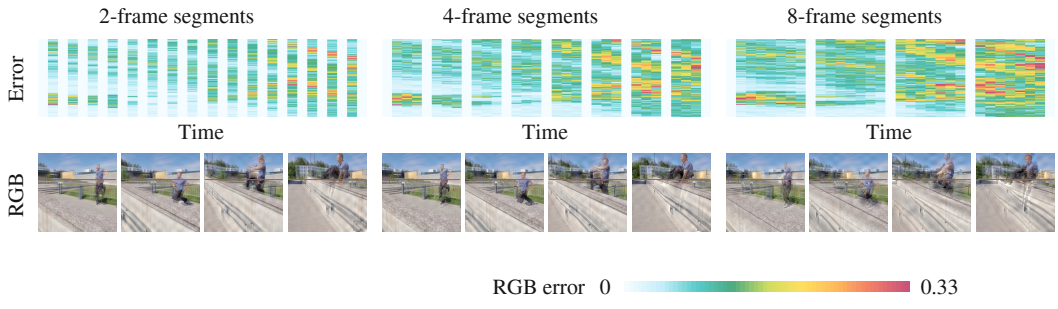

Figure 7: **Longer segments leave more error between their endpoints.** DAVIS *parkour*, frames 0–32, fitted with two-, four-, or eight-frame segments, each with exact endpoints and the same 768-coefficient interior. Strips show RMS RGB error of rows $y = 83$–89 over time (0–0.33); lower views show frames 5, 13, 21, and 29.

REFERENCES

Andrei Atanov, Jesse Allardice, Roman Bachmann, Oğuzhan Fatih Kar, R Devon Hjelm, David Griffiths, Peter Fu, Afshin Dehghan, and Amir Zamir. VideoFlexTok: Flexible-Length Coarse-to-Fine Video Tokenization. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2604.12887.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks. In *Advances in Neural Information Processing Systems*, 2015. URL https://arxiv.org/abs/1506.03099.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets. *arXiv preprint arXiv:2311.15127*, 2023a. URL https://arxiv.org/abs/2311.15127.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023b. URL https://arxiv.org/abs/2304.08818.

Bert De Brabandere, Xu Jia, Tinne Tuytelaars, and Luc Van Gool. Dynamic Filter Networks. In *Advances in Neural Information Processing Systems*, 2016. URL https://arxiv.org/abs/1605.09673.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. DiCache: Let Diffusion Model Determine Its Own Cache. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2508.17356.

Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A Short Note about Kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018. URL https://arxiv.org/abs/1808.01340.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion Forcing: Next-token Prediction Meets Full-Sequence Diffusion. In *Advances in Neural Information Processing Systems*, 2024a. URL https://arxiv.org/abs/2407.01392.

Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. NeRV: Neural Representations for Videos. In *Advances in Neural Information Processing Systems*, 2021. URL https://arxiv.org/abs/2110.13903.

Hao Chen, Matthew Gwilliam, Ser-Nam Lim, and Abhinav Shrivastava. HNeRV: A Hybrid Neural Representation for Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL https://arxiv.org/abs/2304.02633.

Junsong Chen, Jincheng Yu, Yitong Li, Shuchen Xue, Haozhe Liu, Jingyu Xin, Yuyang Zhao, Tian Ye, Zhangjie Wu, Zian Wang, Daquan Zhou, Ping Luo, Song Han, and Enze Xie. SANA-Video 2.0: Hybrid Linear Attention with Attention Residuals for Efficient Video Generation. *arXiv preprint arXiv:2607.21553*, 2026a. URL https://arxiv.org/abs/2607.21553.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, Haozhe Liu, Hongwei Yi, Hao Zhang, Muyang Li, Yukang Chen, Han Cai, Sanja Fidler, Ping Luo, Song Han, and Enze Xie. SANA-Video: Efficient Video Generation with Block Linear Diffusion Transformer. In *International Conference on Learning Representations*, 2026b. URL https://arxiv.org/abs/2509.24695.

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, and Song Han. Deep Compression Autoencoder for Efficient High-Resolution Diffusion Models. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f062da1973ac9ac61fc6d44dd7fa309f-Abstract-Conference.html.

Pengtao Chen, Xianfang Zeng, Maosen Zhao, Peng Ye, Mingzhu Shen, Wei Cheng, Gang Yu, and Tao Chen. Sparse-vDiT: Unleashing the Power of Sparse Attention to Accelerate Video Diffusion Transformers. *arXiv preprint arXiv:2506.03065*, 2025b. URL https://arxiv.org/abs/2506.03065.

Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, and Sergey Tulyakov. Panda-70M: Captioning 70M Videos with Multiple Cross-Modality Teachers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13320–13331, 2024b. URL https://arxiv.org/abs/2402.19479.

Karan Dalal, Daniel Koceja, Jiarui Xu, Yue Zhao, Shihao Han, Ka Chun Cheung, Jan Kautz, Yejin Choi, Yu Sun, and Xiaolong Wang. One-Minute Video Generation with Test-Time Training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Dalal_One-Minute_Video_Generation_with_Test-Time_Training_CVPR_2025_paper.html.

Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.08691.

Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *International Conference on Machine Learning*, 2022. URL https://proceedings.mlr.press/v162/dupont22a.html.

Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. HyperDiffusion: Generating Implicit Neural Fields with Weight-Space Diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2303.17015.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming Transformers for High-Resolution Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. URL https://arxiv.org/abs/2012.09841.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised Learning for Physical Interaction through Video Prediction. In *Advances in Neural Information Processing Systems*, 2016. URL https://arxiv.org/abs/1605.07157.

Jialong Guo, Ke Liu, Mengxuan Li, Jiajun Bu, and Haishuai Wang. CoANeRV: Coordinate-Aware Token-Space Neural Video Representation. *arXiv preprint arXiv:2608.13938*, 2026. URL https://arxiv.org/abs/2608.13938.

David Ha, Andrew M. Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1609.09106.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. LTX-Video: Realtime Video Latent Diffusion. *arXiv preprint arXiv:2501.00103*, 2024. URL https://arxiv.org/abs/2501.00103.

Yasaman Haghighi and Alexandre Alahi. SenCache: Accelerating Diffusion Model Inference via Sensitivity-Aware Caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14295–14304, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Haghighi_SenCache_Accelerating_Diffusion_Model_Inference_via_Sensitivity-Aware_Caching_CVPR_2026_paper.html.

Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. StreamingT2V: Consistent, Dynamic, and Extendable Long Video Generation from Text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2403.14773.

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. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video Diffusion Models. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2204.03458.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self Forcing: Bridging the Train-Test Gap in Autoregressive Video Diffusion. In *Advances in Neural Information Processing Systems*, volume 38, pp. 167283–167308, 2025. URL https://arxiv.org/abs/2506.08009.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive Benchmark Suite for Video Generative Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024. URL https://arxiv.org/abs/2311.17982.

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. URL https://arxiv.org/abs/2411.13503.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*, 2015. URL https://arxiv.org/abs/1506.02025.

Huiwon Jang, Sihyun Yu, Jinwoo Shin, Pieter Abbeel, and Younggyo Seo. Efficient Long Video Tokenization via Coordinate-based Patch Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22853–22863, 2025. URL https://arxiv.org/abs/2411.14762.

Xuan Ju, Yiming Gao, Zhaoyang Zhang, Ziyang Yuan, Xintao Wang, Ailing Zeng, Yu Xiong, Qiang Xu, and Ying Shan. MiraData: A Large-Scale Video Dataset with Long Durations and Structured Captions. In *Advances in Neural Information Processing Systems*, volume 37, pp. 48955–48970, 2024. URL https://arxiv.org/abs/2407.06358.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *International Conference on Machine Learning*, 2020. URL https://arxiv.org/abs/2006.16236.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A Systematic Framework For Large Video Generative Models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

Xingyang Li, Muyang Li, Tianle Cai, Haocheng Xi, Shuo Yang, Yujun Lin, Lvmin Zhang, Songlin Yang, Jinbo Hu, Kelly Peng, Maneesh Agrawala, Ion Stoica, Kurt Keutzer, and Song Han. Radial Attention: $\mathcal{O}(n \log n)$ Sparse Attention with Energy Decay for Long Video Generation. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.19852.

Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative Image Dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2309.07906.

Xinran Ling, Chen Zhu, Meiqi Wu, Hangyu Li, Xiaokun Feng, Cundian Yang, Aiming Hao, Jiashu Zhu, Jiahong Wu, and Xiangxiang Chu. VMBench: A Benchmark for Perception-Aligned Video Motion Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13087–13098, 2025. URL https://arxiv.org/abs/2503.10076.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Enhuai Liu, Yunke Wang, Changming Sun, and Chang Xu. D2Cache: Second-Order Delta Caching for Higher Video Diffusion Acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43589–43599, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Liu_D2Cache_Second-Order_Delta_Caching_for_Higher_Video_Diffusion_Acceleration_CVPR_2026_paper.html.

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*, 2025. URL https://arxiv.org/abs/2411.19108.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.03003.

Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video Frame Synthesis using Deep Voxel Flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017. URL https://arxiv.org/abs/1702.02463.

Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent Diffusion Transformer for Video Generation. *Transactions on Machine Learning Research*, 2025. URL https://arxiv.org/abs/2401.03048.

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*, 2024. URL https://arxiv.org/abs/2312.00858.

Aniruddha Mahapatra, Long Mai, David Bourgin, Yitian Zhang, and Feng Liu. Progressive Growing of Video Tokenizers for Temporally Compact Latent Spaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17629–17639, 2025. URL https://arxiv.org/abs/2501.05442.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do Generative Video Models Understand Physical Principles? In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 948–958, 2026. URL https://arxiv.org/abs/2501.09038.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025. URL https://arxiv.org/abs/2501.03575.

Namitha Padmanabhan, Matthew Gwilliam, and Abhinav Shrivastava. TeCoNeRV: Leveraging Temporal Coherence for Compressible Neural Representations for Videos. *arXiv preprint arXiv:2602.16711*, 2026. URL https://arxiv.org/abs/2602.16711.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2212.09748.

William Peebles, Ilija Radosavovic, Tim Brooks, Alexei A. Efros, and Jitendra Malik. Learning to Learn with Generative Models of Neural Network Checkpoints. *arXiv preprint arXiv:2209.12892*, 2022. URL https://arxiv.org/abs/2209.12892.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.01952.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie Gen: A Cast of Media Foundation Models. *arXiv preprint arXiv:2410.13720*, 2024. URL https://arxiv.org/abs/2410.13720.

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS Challenge on Video Object Segmentation. *arXiv preprint arXiv:1704.00675*, 2017. URL https://arxiv.org/abs/1704.00675.

Yixuan Ren, Hanyu Wang, Bo He, Hao Chen, and Abhinav Shrivastava. NeRV-Diffusion: Diffuse Implicit Neural Representation for Video Synthesis. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1a17a06de88cf77f25cda0da91615a54-Abstract-Conference.html.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://arxiv.org/abs/2112.10752.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2202.00512.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers. In *International Conference on Machine Learning*, 2021. URL https://arxiv.org/abs/2102.11174.

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. URL https://arxiv.org/abs/2407.01425.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-A-Video: Text-to-Video Generation without Text-Video Data. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.14792.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2303.01469.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *arXiv preprint arXiv:1212.0402*, 2012. URL https://arxiv.org/abs/1212.0402.

Kaiyue Sun, Kaiyi Huang, Xian Liu, Yue Wu, Zihan Xu, Zhenguo Li, and Xihui Liu. T2V-CompBench: A Comprehensive Benchmark for Compositional Text-to-video Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8406–8416, 2025a. URL https://arxiv.org/abs/2407.14505.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (Learn at Test Time): RNNs with Expressive Hidden States. In *International Conference on Machine Learning*, 2025b. URL https://arxiv.org/abs/2407.04620.

Anni Tang, Tianyu He, Junliang Guo, Xinle Cheng, Li Song, and Jiang Bian. VidTok: A Versatile and Open-Source Video Tokenizer. *arXiv preprint arXiv:2412.13061*, 2024. URL https://arxiv.org/abs/2412.13061.

Team Wan. Wan: Open and Advanced Large-Scale Video Generative Models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

Rui Tian, Qi Dai, Jianmin Bao, Kai Qiu, Yifan Yang, Chong Luo, Zuxuan Wu, and Yu-Gang Jiang. REDUCIO! Generating 1K Video within 16 Seconds using Extremely Compressed Motion Latents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2411.13552.

Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing Motion and Content for Video Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. URL https://arxiv.org/abs/1707.04993.

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. URL https://arxiv.org/abs/1812.01717.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1711.00937.

Hanyu Wang, Saksham Suri, Yixuan Ren, Hao Chen, and Abhinav Shrivastava. LARP: Tokenizing Videos with a Learned Autoregressive Generative Prior. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.21264.

Kai Wang, Dongwen Tang, Boya Zeng, Yida Yin, Zhaopan Xu, Yukun Zhou, Zelin Zang, Trevor Darrell, Zhuang Liu, and Yang You. Neural Network Diffusion. *arXiv preprint arXiv:2402.13144*, 2024. URL https://arxiv.org/abs/2402.13144.

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. URL https://ece.uwaterloo.ca/~z70wang/research/ssim/.

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. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 68208–68224. PMLR, 2025. URL https://proceedings.mlr.press/v267/xi25c.html.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. SANA: Efficient High-Resolution Text-to-Image Synthesis with Linear Diffusion Transformers. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/34345e243156da67605d4b63d71c8d98-Abstract-Conference.html.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Xu_MSR-VTT_A_Large_CVPR_2016_paper.html.

Wilson Yan, Volodymyr Mnih, Aleksandra Faust, Matei Zaharia, Pieter Abbeel, and Hao Liu. ElasticTok: Adaptive Tokenization for Image and Video. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.08368.

Brandon Yang, Gabriel Bender, Quoc V. Le, and Jiquan Ngiam. CondConv: Conditionally Parameterized Convolutions for Efficient Inference. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1904.04971.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2408.06072.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved Distribution Matching Distillation for Fast Image Synthesis. In *Advances in Neural Information Processing Systems*, 2024a. URL https://arxiv.org/abs/2405.14867.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step Diffusion with Distribution Matching Distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b. URL https://arxiv.org/abs/2311.18828.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Frédo Durand, Eli Shechtman, and Xun Huang. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2412.07772.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. MAGVIT: Masked Generative Video Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023a. URL https://arxiv.org/abs/2212.05199.

Lijun Yu, José Lezama, Nitesh B. Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language Model Beats Diffusion – Tokenizer is Key to Visual Generation. In *International Conference on Learning Representations*, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/036912a83bdbb1fd792baf6532f102d8-Abstract-Conference.html.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An Image is Worth 32 Tokens for Reconstruction and Generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 128940–128966, 2024b. URL https://arxiv.org/abs/2406.07550.

Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating Videos with Dynamics-aware Implicit Generative Adversarial Networks. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2202.10571.

Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video Probabilistic Diffusion Models in Projected Latent Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023b. URL https://arxiv.org/abs/2302.07685.

Sihyun Yu, Weili Nie, De-An Huang, Boyi Li, Jinwoo Shin, and Anima Anandkumar. Efficient Video Diffusion Models via Content-Frame Motion-Latent Decomposition. In *International Conference on Learning Representations*, 2024c. URL https://arxiv.org/abs/2403.14148.

Zhihang Yuan, Rui Xie, Yuzhang Shang, Hanling Zhang, Siyuan Wang, Shengen Yan, Guohao Dai, and Yu Wang. DLFR-Gen: Diffusion-based Video Generation with Dynamic Latent Frame Rate. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16410–16419, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Yuan_DLFR-Gen_Diffusion-based_Video_Generation_with_Dynamic_Latent_Frame_Rate_ICCV_2025_paper.html.

Jintao Zhang, Jia Wei, Pengle Zhang, Jun Zhu, and Jianfei Chen. SageAttention: Accurate 8-Bit Attention for Plug-and-play Inference Acceleration. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/b286c344d38e10d2466c0514b78e2f36-Abstract-Conference.html.

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. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76397–76413. PMLR, 2025b. URL https://arxiv.org/abs/2502.18137.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame Context Packing and Drift Prevention in Next-Frame-Prediction Video Diffusion Models. In *Advances in Neural Information Processing Systems*, 2025c. URL https://arxiv.org/abs/2504.12626.

Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast Video Generation with Sliding Tile Attention. In *International Conference on Machine Learning*, 2025d. URL https://arxiv.org/abs/2502.04507.

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*, 2018. URL https://arxiv.org/abs/1801.03924.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-Time Video Generation with Pyramid Attention Broadcast. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2408.12588.

Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-Sora: Democratizing Efficient Video Production for All. *arXiv preprint arXiv:2412.20404*, 2024. URL https://arxiv.org/abs/2412.20404.

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. IRASim: A Fine-Grained World Model for Robot Manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9834–9844, 2025. URL https://arxiv.org/abs/2406.14540.

AI USE STATEMENT

Generative AI tools supported literature search and discovery, research ideation and mathematical exposition, drafting manuscript sections, and language editing. The authors retain responsibility for reviewing AI-assisted content and verifying the final claims, references, and results.

A ALGORITHM AND IMPLEMENTATION

This appendix specifies everything needed to run KineTra with a given backbone: the complete procedure (Algorithm 1), the executor and encoder (Appendix A.1), the backbone interfaces (Appendix A.2), conditioning and output assembly (Appendix A.3), and training and measurement settings (Appendix A.4). Algorithm 1 uses one rollout, $\mathrm{Rollout}$, for both constructing training weights and executing generated ones, so that the executor sees the same kind of input in both stages.

Algorithm 1 KineTra: representation learning, flow learning, and generation

Require: data $\mathcal{T}$; frozen VAE $(\mathcal{E}, \mathcal{D})$; initial parameters (ϕ, ψ, θ); segment length R; weight-token layout.
Require: iteration counts $I_{\mathrm{rep}}, I_{\mathrm{flow}}$; AdamW settings; Euler grid $1 = s_N > \cdots > s_0 = 0$ with $N = 50$.
Require: test condition $c^\star$ and segment count K; generation mask $\mathbf{m}$; observed coordinates $\mathbf{q}_{\mathrm{obs}}$, zero where $\mathbf{m} = 1$.
Ensure: trained (ϕ, ψ, θ) and a generated RGB video $\widehat{X}$.

function $\mathrm{Rollout}(X, c; \phi, \psi)$ *// encode a video along its own predicted boundaries*
  $Z \leftarrow \mathcal{E}(\mathrm{Pad}(X))$; $K \leftarrow (T-1)/R$; $B_0 \leftarrow \mathrm{Init}(X, c)$; $Z_0 \leftarrow B_0$; $\widehat{Z}_0 \leftarrow B_0$
  for $k = 1, \ldots, K$ **do**
    $O_k \leftarrow E_\phi(B_{k-1}, Z_{(k-1)R+1:kR})$; $w_k \leftarrow \mathrm{vec}(O_k)$
    $B_k \leftarrow F(B_{k-1}, w_k)$; $\widehat{Z}_{kR} \leftarrow B_k$ *// shared endpoint, computed once*
    for $j = 1, \ldots, R-1$ **do** $\widehat{Z}_{(k-1)R+j} \leftarrow H(B_{k-1}, w_k, j/R)$ **end for**
  end for
  return $\big(\mathrm{vec}(B_0, O_{1:K}),\ \widehat{Z},\ Z\big)$
end function
Stage 1: learn the encoder and executor through predicted boundaries
for $i = 1, \ldots, I_{\mathrm{rep}}$ **do**
  sample a batch $\mathcal{B} \subset \mathcal{T}$; $(\mathbf{q}_b, \widehat{Z}_b, Z_b) \leftarrow \mathrm{Rollout}(X_b, c_b; \phi, \psi)$ for $b \in \mathcal{B}$
  $\mathcal{L}_{\mathrm{rep}} \leftarrow |\mathcal{B}|^{-1} \sum_{b \in \mathcal{B}} \|\widehat{Z}_b - Z_b\|_F^2 / (T_b S C)$
  $(\phi, \psi) \leftarrow \mathrm{AdamW}\big((\phi, \psi), \nabla_{\phi,\psi} \mathcal{L}_{\mathrm{rep}}\big)$
end for; keep the checkpoint with the lowest validation $\mathcal{L}_{\mathrm{rep}}$
Stage 2: freeze (ϕ, ψ) **and learn to generate the weights**
for $i = 1, \ldots, I_{\mathrm{flow}}$ **do**
  sample $\mathcal{B} \subset \mathcal{T}$; $\mathbf{q}_b \leftarrow \mathrm{stopgrad}\big(\mathrm{Rollout}(X_b, c_b; \phi, \psi)_1\big)$; $s_b \sim \mathcal{U}[0,1]$; $\boldsymbol{\epsilon}_b \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $\mathbf{q}_{b,s} \leftarrow \mathbf{q}_b + s_b\, \mathbf{m} \odot (\boldsymbol{\epsilon}_b - \mathbf{q}_b)$
  $\mathcal{L}_{\mathrm{flow}} \leftarrow |\mathcal{B}|^{-1} \sum_{b \in \mathcal{B}} \big\| \mathbf{m} \odot \big(v_\theta(\mathbf{q}_{b,s}, s_b, c_b) - (\boldsymbol{\epsilon}_b - \mathbf{q}_b)\big) \big\|_2^2 / \|\mathbf{m}\|_1$
  $\theta \leftarrow \mathrm{AdamW}(\theta, \nabla_\theta \mathcal{L}_{\mathrm{flow}})$
end for
Stage 3: sample the weights, then execute them
$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; $\mathbf{q}_{s_N} \leftarrow \mathbf{q}_{\mathrm{obs}} + \mathbf{m} \odot \boldsymbol{\epsilon}$
for $i = N, \ldots, 1$ **do** $\mathbf{q}_{s_{i-1}} \leftarrow \mathbf{q}_{s_i} + (s_{i-1} - s_i)\, \mathbf{m} \odot v_\theta(\mathbf{q}_{s_i}, s_i, c^\star)$ **end for**
$(B_0, O_{1:K}) \leftarrow \mathrm{unvec}(\mathbf{q}_{s_0})$; $\widehat{Z}_0 \leftarrow B_0$
for $k = 1, \ldots, K$ **do**
  $w_k \leftarrow \mathrm{vec}(O_k)$; $B_k \leftarrow F(B_{k-1}, w_k)$; $\widehat{Z}_{kR} \leftarrow B_k$
  for $j = 1, \ldots, R-1$ **do** $\widehat{Z}_{(k-1)R+j} \leftarrow H(B_{k-1}, w_k, j/R)$ **end for**
end for
return $\widehat{X} \leftarrow \mathrm{Valid}\big(\mathcal{D}(\widehat{Z}_{0:KR})\big)$

The maps F and H are those of Equation (2). vec concatenates the first frame and the ordered weight tokens without changing their values, and unvec restores their shapes. $\mathrm{Init}(X, c)$ returns Z_0 for text and class conditioning, the posterior-mode latent of the isolated first image for image conditioning, and the native initial latent for action conditioning. Pad and Valid are defined in Appendix A.3; iteration counts, optimizer settings, and fixed permutations are configuration inputs, and the algorithm fixes the forward paths, losses, gradient paths, and masking.

A.1 Executor and Slot Encoder

Executor. The executor has $L = 8$ layers of width $d = 512$ with $G = 16$ channel groups of width $d_g = 32$. Weight tokens $2(\ell G + g) - 1$ and $2(\ell G + g)$ hold the two halves of $M_{k,\ell,g}$ for $\ell = 0, \dots, L-1$ and $g = 1, \dots, G$, so $n_w = 2LG = 256$ and $D_w = LGd_g^2 = n_w d_w = 131{,}072$. The input is $V^0 = BW_{\text{in}} + \pi_{\text{pos}}(p) + \pi_{\text{time}}(\tau)$, where W_{in} embeds the carried frame and π_{pos} and π_{time} embed the spatial coordinates p and the local time τ; an output projection maps V^L to the residual $U_\psi(B, w, \tau) \in \mathbb{R}^{S \times C}$. All embeddings and projections are shared across segments and local times. In Equation (1), each attention head projects $\widetilde{V}^\ell$ with learned matrices $W^{\text{q}}, W^{\text{k}} \in \mathbb{R}^{d \times d_a}$ and $W^{\text{v}} \in \mathbb{R}^{d \times d_v}$ and, with $\text{sp}(x) = \log(1 + e^x)$ applied elementwise and rows x_i of $\widetilde{V}^\ell$, computes

$$\text{LinAttn}(\widetilde{V}^\ell)_i = \frac{\text{sp}(x_i W^{\text{q}}) \sum_{i'=1}^{S} \text{sp}(x_{i'} W^{\text{k}})^\top x_{i'} W^{\text{v}}}{\text{sp}(x_i W^{\text{q}}) \sum_{i'=1}^{S} \text{sp}(x_{i'} W^{\text{k}})^\top + \varepsilon_{\text{attn}}}, \qquad \varepsilon_{\text{attn}} > 0, \tag{7}$$

which costs $\mathcal{O}(S)$ per layer. Head outputs are concatenated and projected to width d. Layer normalization uses $\varepsilon_{\text{LN}} > 0$, and $\text{FFN}_\ell(Y) = \text{SwiGLU}(YW_{1,\ell})W_{2,\ell}$ with hidden width $4d$. The permutation Π_ℓ is fixed and moves channels between groups from one layer to the next. The endpoint $F(B, w)$ is evaluated once per segment; interior times can be batched, while the K endpoints are computed in sequence.

Slot encoder. E_ϕ has six transformer blocks of width 512 with eight heads. Its n_w learned queries attend to the tokens of the incoming predicted boundary B_{k-1} and of the target frames $Z_{(k-1)R+1:kR}$, and each query returns the weight token of one fixed slot.

A.2 Backbone Interfaces

Table 4: **Backbone interfaces for 961 frames at** 512×768**.** Text tokens are excluded. Patch size is temporal $\times$ height $\times$ width on VAE latents. SANA and LTX use spatial/temporal reduction $32/8$; CogVideoX and Wan use $8/4$. Architectures and checkpoints follow Chen et al. (2026a); HaCohen et al. (2024); Yang et al. (2025); Huang et al. (2025); Team Wan (2025).

Pretrained body	Transformer / latent			Dense patch	Video tokens	
	Layers	Width	C		Dense	KineTra
SANA-Video 2.0 (Chen et al., 2026a) / 5B	32	2560	128	$1 \times 1 \times 1$	46,464	8,064
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev	48	4096	128	$1 \times 1 \times 1$	46,464	8,064
CogVideoX1.5 (Yang et al., 2025) / 5B	42	3072	16	$2 \times 2 \times 2$	185,856	21,504
Self Forcing (Huang et al., 2025) / Wan2.1-T2V / 1.3B	30	1536	16	$1 \times 2 \times 2$	370,176	21,504

Table 4 lists, for each backbone, the dense video tokens and the KineTra tokens for a 961-frame video. SANA uses the 2.0 5B 720p model (Chen et al., 2026a) with its attention residuals and its layout of three linear-attention layers per softmax layer. LTX uses the 13B 0.9.8-dev model (HaCohen et al., 2024) with self- and cross-attention. CogVideoX1.5-5B (Yang et al., 2025) keeps its joint text–video updates and normalization and is converted to the flow objective. The causal model uses Self Forcing's exponential-moving-average parameters (Huang et al., 2025), the Wan2.1 VAE and text encoder (Team Wan, 2025), and a block-causal mask over the full history. In KineTra, video rotary rotations are set to the identity, and frame and weight tokens receive fixed positions. All body parameters are trained in Stage 2.

A.3 Conditioning, Output Assembly, and Padding

Image conditioning. The Stage 1 target is $(B_0, Z_1, \dots, Z_{T-1})$, with B_0 the posterior-mode encoding of the isolated first image and the other targets from the posterior-mode encoding of the full video. Encoding the image alone keeps future frames out of the condition, and using the same B_0 as the first target keeps the first reconstruction consistent with the condition. During sampling, B_0 stays clean and only the weights are generated ($\mathbf{m}$ selects the weight coordinates). Image and UCF class models are trained separately for each family and seed.

Action conditioning. Action prediction keeps IRASim's Video-Ada trajectory embedding and modulation. RT-1 and Bridge use 256×320 inputs and Language Table 288×512, with the SDXL spatial VAE (Podell et al., 2024). The 15 actions keep their original scaling: the first six RT-1/Bridge components are multiplied by 20 and the gripper component is unchanged, and both Language Table

components are multiplied by 20. Here $R = 5$, $K = 3$, the executor has depth 32, and the widths are 640, 640, and 1152. Native patching gives $n_w = S$, so both representations have $4S$ transformer tokens with different spatial and temporal layouts.

Output assembly and padding. The output is B_0 followed by the R frames of each segment. Pad repeats the final RGB frame and, where needed, pads the latent sequence to the native temporal patch size; padded positions enter the training loss and the measured cost. Valid returns the valid decoded frames, to which each benchmark applies its own resizing and cropping, identically for paired methods.

A.4 Data, Optimization, Controls, and Measurement

Data and optimization. Open-domain training mixes two million Panda-70M short windows (Chen et al., 2024b) with long windows from MiraData 330k (Ju et al., 2024) in an 80/20 ratio. We hold out 10,000 source videos of each pool for validation, keep duplicate sources in one split, and separate training and validation from all evaluation sources and prompts. Both stages use AdamW with batch size 16, learning rates 10^{-4} (Stage 1) and 10^{-5} (Stage 2), and cosine decay. Stage 1 keeps the checkpoint with the lowest validation latent MSE, and generation uses the final Stage 2 model. KineTra spends one quarter of the common training budget on Stage 1 and three quarters on Stage 2, and Dense spends the whole budget on flow training; the budget includes video encoding, both stages, validation, and communication.

Component controls. Each control in Table 3 changes one part and keeps KineTra's update counts and data order. *Linear interior* removes the term $\tau(1-\tau)U_\psi$ in Equation (2). *Initial state only* feeds B_0 to every segment. *Teacher boundaries* feeds true VAE boundaries in Stage 1 and when constructing training weights, and predicted boundaries at generation. *Tied segment code* uses global queries that produce one O for all segments. All four use the same reconstruction objective.

Timing and statistics. Timing uses A100-SXM 80GB GPUs with BF16 parameters and activations. Complete-video time runs from conditioning to the transfer and saving of the valid output, with participating GPUs synchronized, and includes transformer sampling, execution, decoding, and communication; decoder compilation and warm-up precede timing. Memory is the largest allocated peak over phases and devices. For a learned configuration, training-seed intervals are $\bar{x} \pm t_{0.975,2}\,\hat{\sigma}/\sqrt{3}$ with sample standard deviation $\hat{\sigma}$, and paired intervals use the three within-seed differences. Data intervals use 2,000 cluster-bootstrap resamples with percentile endpoints; each resample keeps paired methods and all ratings, views, or windows of a cluster together, and FVD and FID are recomputed from the resampled feature statistics.

B Analysis

This appendix proves the two results of Section 3.5, extends them to interior frames, generated matrices, and the decoder, derives the cost ratio, and compares the bounds with measured errors.

B.1 Code Size for Independent Segments

Theorem 1 concerns the local dimension of the family of videos that KineTra can produce around a reference. The first frame and the shared parameters are fixed, so the coordinates describe only segment-specific variation.

Setup. Let u_k range over an open neighborhood $\mathcal{N}_k$ of zero in a Euclidean space, let $w_k : \mathcal{N}_1 \times \cdots \times \mathcal{N}_k \to \mathbb{R}^{D_w}$ be continuously differentiable, and let $B_k(u_{1:k}) = F(B_{k-1}(u_{1:k-1}), w_k(u_{1:k}))$. The frames returned by a segment, excluding its entrance, are

$$\mathcal{H}(B, w) = \mathrm{vec}\big(H(B, w, 1/R), \ldots, H(B, w, 1)\big), \qquad \Phi_k(u) = \mathcal{H}\big(B_{k-1}(u_{<k}), w_k(u_{\le k})\big). \tag{8}$$

Write $B_k^\circ = B_k(0)$, $w_k^\circ = w_k(0)$, and $\Phi = (\Phi_1, \ldots, \Phi_K)$. The coordinate u_k can change segment k without changing earlier frames, so $\mathrm{D}_{u_j}\Phi_i(0) = 0$ for $j > i$. The diagonal blocks are

$$A_k = \mathrm{D}_w \mathcal{H}(B_{k-1}^\circ, w_k^\circ)\,\mathrm{D}_{u_k} w_k(0), \qquad r_k = \operatorname{rank} A_k, \tag{9}$$

because B_{k-1} does not depend on u_k; derivatives through earlier boundaries appear below the diagonal.

Lemma 1 (Rank of a block lower triangular matrix)**.** *If J is block lower triangular with possibly rectangular diagonal blocks A_k, then* $\operatorname{rank} J \geq \sum_k \operatorname{rank} A_k$.

Proof. For each k with $r_k > 0$, choose r_k rows and r_k columns of A_k that form an invertible minor. Keep the union of these rows and columns of J in segment order. The resulting square matrix is block lower triangular with the chosen minors on its diagonal, so its determinant is the product of their nonzero determinants. J therefore has a nonsingular minor of order $\sum_k r_k$. □

Proof of Theorem 1. By causality, $\mathrm{D}\Phi(0)$ is block lower triangular with diagonal blocks A_k, so Lemma 1 gives $\operatorname{rank} \mathrm{D}\Phi(0) \geq \sum_k r_k$. If a continuously differentiable encoder $\mathcal{C}$ with values in $\mathbb{R}^n$ and a continuously differentiable decoder $\mathcal{R}$ satisfy $\mathcal{R}(\mathcal{C}(u)) = \Phi(u)$ near zero, the chain rule gives $\mathrm{D}\Phi(0) = \mathrm{D}\mathcal{R}(\mathcal{C}(0))\,\mathrm{D}\mathcal{C}(0)$, whose rank is at most n. This proves Equation (5). The argument covers any fixed global code, whether it stores content or weights. □

Exact reproduction is stronger than what a learned tokenizer achieves. The next result states which first-order variations a smaller smooth code must miss.

Proposition 1 (First-order approximation error)**.** *Let $J = \mathrm{D}\Phi(0)$, and let $\widetilde{\Phi} = \widetilde{\mathcal{R}} \circ \widetilde{\mathcal{C}}$ be continuously differentiable through an n-dimensional code with $\widetilde{\Phi}(0) = \Phi(0)$. If ξ is bounded and $\mathbb{E}[\xi\xi^\top] = \mathbf{I}$, then, with singular values in nonincreasing order,*

$$\liminf_{h \downarrow 0} \; h^{-2}\, \mathbb{E}\big\|\Phi(h\xi) - \widetilde{\Phi}(h\xi)\big\|_2^2 \;\geq\; \textstyle\sum_{i>n} \sigma_i(J)^2. \tag{10}$$

Proof. Let $\widetilde{J} = \mathrm{D}\widetilde{\Phi}(0)$, whose rank is at most n. Continuous differentiability and bounded ξ give $\Phi(h\xi) - \widetilde{\Phi}(h\xi) = h(J - \widetilde{J})\xi + o(h)$ uniformly on the support of ξ, so the left side of Equation (10) equals $\|J - \widetilde{J}\|_F^2$. For any matrix A of rank at most n with column-space projector P_A, $\|J - A\|_F^2 \geq \|(\mathbf{I} - P_A)J\|_F^2 = \|J\|_F^2 - \operatorname{tr}(P_A J J^\top)$. In an eigenbasis of $JJ^\top$ the diagonal entries of P_A lie in $[0,1]$ and sum to at most n, so $\operatorname{tr}(P_A JJ^\top) \leq \sum_{i \leq n} \sigma_i(J)^2$. Taking $A = \widetilde{J}$ proves Equation (10); independent Rademacher signs are one admissible ξ. □

For KineTra, $r_k \leq D_w$, and inactive directions, saturation of $\tanh$, and redundant transformations can lower it. Per-segment weights let the available coordinates grow with the number of independently changing segments, which is the property tested by the tied-code control in Section 4.2. Figure 8 illustrates the same effect on source pixels: a rank-three temporal approximation fitted to a short window leaves less residual than one fitted to a long window, because the short window contains fewer independent changes.

B.2 Boundary Error Recursion

Setup. Fix a reference weight sequence $w_{1:K}$ with rollout $\bar{B}_k = F(\bar{B}_{k-1}, w_k)$, and a perturbed sequence $\hat{w}_{1:K}$, for example sampled weights. The computed boundary is $\hat{B}_k = F(\hat{B}_{k-1}, \hat{w}_k) + \rho_k$, where F is the exact map and $\|\rho_k\| \leq \nu_k$ is numerical error. Let $\Delta B_{k-1} = \hat{B}_{k-1} - \bar{B}_{k-1}$, $\Delta w_k = \hat{w}_k - w_k$, $e_k = \|\Delta B_k\|$, and $\omega_k = \|\Delta w_k\|$, with Euclidean norms of vectorized tensors. With the path points $B^\lambda = \bar{B}_{k-1} + \lambda \Delta B_{k-1}$ and $w^\lambda = w_k + \lambda \Delta w_k$, the path gains for nonzero displacements are

$$\alpha_k = \int_0^1 \frac{\|\mathrm{D}_B F(B^\lambda, \hat{w}_k)\, \Delta B_{k-1}\|}{e_{k-1}}\, d\lambda, \qquad \beta_k = \int_0^1 \frac{\|\mathrm{D}_w F(\bar{B}_{k-1}, w^\lambda)\, \Delta w_k\|}{\omega_k}\, d\lambda, \tag{11}$$

and a gain is zero when its displacement is zero. Continuous differentiability along these segments, ensured by the positive stabilizers in normalization and attention, suffices below.

Proof of Theorem 2. Adding and subtracting $F(\bar{B}_{k-1}, \hat{w}_k)$ gives

$$\hat{B}_k - \bar{B}_k = \big[F(\hat{B}_{k-1}, \hat{w}_k) - F(\bar{B}_{k-1}, \hat{w}_k)\big] + \big[F(\bar{B}_{k-1}, \hat{w}_k) - F(\bar{B}_{k-1}, w_k)\big] + \rho_k. \tag{12}$$

The fundamental theorem of calculus writes the two brackets as integrals of the derivatives along the segments in Equation (11), and the triangle inequality gives $e_k \leq \alpha_k e_{k-1} + \beta_k \omega_k + \nu_k$. Once

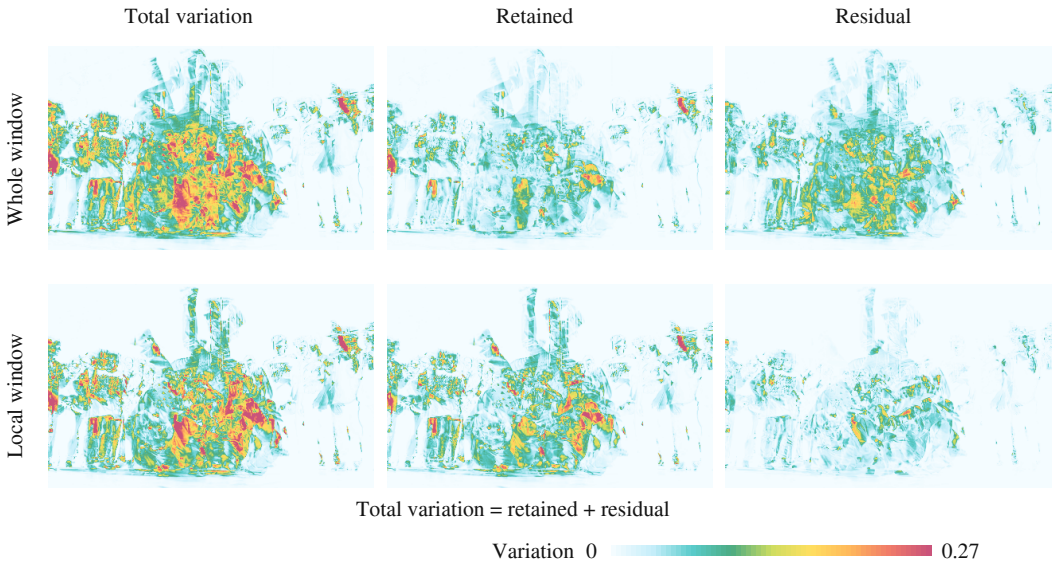

Figure 8: **A fixed rank budget fits a short window better than a long one.** Temporal PCA on DAVIS *breakdance*, shown at 410×256. The whole window has 11 sampled frames spanning source frames 0–83; the local window has six of them, spanning 25–66. Each window is centered per pixel and fitted independently with rank three. Columns show the total, retained, and residual energy (temporal mean of summed squared RGB components), with total = retained + residual at every pixel, on one 0–0.27 scale.

B_k is stored, all later executor states are compositions of F applied to B_k and the later weights, so a change of a returned frame that leaves the stored B_k unchanged does not enter them. □

Accumulated bound and segment length. All factors are nonnegative, so with $\mu_i = \beta_i\omega_i + \nu_i$ and empty products equal to one, induction gives

$$e_k \le \Big(\prod_{j=1}^{k} \alpha_j\Big) e_0 + \sum_{i=1}^{k} \mu_i \prod_{j=i+1}^{k} \alpha_j. \tag{13}$$

At a differentiable reference, a boundary error ΔB_k changes a later boundary $B_{k'}$ to first order by $\mathrm{D}_B F_{k'} \cdots \mathrm{D}_B F_{k+1}\, \Delta B_k$; this path creates the blocks below the diagonal of $\mathrm{D}\Phi$ in Appendix B.1. With uniform bounds $\alpha_k \le \bar\alpha$ and $\mu_k \le \bar\mu$, both depending on R,

$$e_K \le \bar\alpha^K e_0 + \bar\mu\, \Gamma_K(\bar\alpha), \qquad \Gamma_K(x) = \begin{cases} (1-x^K)/(1-x), & x \ne 1, \\ K, & x = 1. \end{cases} \tag{14}$$

For $\bar\alpha < 1$ the new-error term stays below $\bar\mu/(1-\bar\alpha)$; for $\bar\alpha = 1$ it grows linearly in K; and for $\bar\alpha > 1$ it can grow exponentially. For a fixed horizon $T - 1 = KR$, the initial-error factor is $\exp[(T-1)\log(\bar\alpha)/R]$ when $\bar\alpha > 0$, so comparing segment lengths requires $\log(\bar\alpha)/R$, $\bar\mu\,\Gamma_K(\bar\alpha)$, and the reconstruction residuals together. The scalar map $F(x, w) = \alpha x + w$ with $\alpha \ge 0$, aligned perturbations, constant $\omega_k = \omega$, and no numerical error attains $e_K = \alpha^K e_0 + \omega\,\Gamma_K(\alpha)$. It also shows that fewer boundaries do not always reduce error: with $\alpha = 1/2$, $\omega = 0$, and $e_0 = 1$, halving the number of boundaries raises the final error from 2^{-K} to $2^{-K/2}$. The R sweep in Section 4.2 is therefore a measured trade-off.

Interior frames. For $\tau_j = j/R$, let $\bar Z_{k,j} = H(\bar B_{k-1}, w_k, \tau_j)$ and $\hat Z_{k,j} = H(\hat B_{k-1}, \hat w_k, \tau_j) + \rho_{k,j}$, and define $\alpha_{k,j}, \beta_{k,j}$ by replacing F with $H(\cdot,\cdot,\tau_j)$ in Equation (11). With target latent $Z^\star_{k,j}$ and reconstruction residual $\zeta_{k,j} = \|\bar Z_{k,j} - Z^\star_{k,j}\|$, the same decomposition gives

$$\big\|\hat Z_{k,j} - Z^\star_{k,j}\big\| \le \zeta_{k,j} + \alpha_{k,j} e_{k-1} + \beta_{k,j}\omega_k + \nu_{k,j}. \tag{15}$$

The reconstruction residual $\zeta_{k,j}$ is separate from the error of sampled weights and from the propagated boundary error. The raw latent MSE in Section 4.3 measures this executed trajectory before any returned frame is replaced.

Generated matrices. The map $w \mapsto M(w) = \gamma_\ell \tanh(w)$ satisfies $\|M(w) - M(\widetilde{w})\|_F \le |\gamma_\ell| \, \|w - \widetilde{w}\|_2$ on each slot. For the features V of an active group, $VM - V'M' = (V - V')M + V'(M - M')$, so

$$\|VM - V'M'\|_F \le \|V - V'\|_F \, \|M\|_2 + \|V'\|_2 \, \|M - M'\|_F. \tag{16}$$

Applying this inequality layer by layer, together with the shared projections, attention, normalization, and residual connections, gives a telescoping bound with products of layer gains; the path gains in Equation (11) contain all of these interactions.

Decoder. Let $\mathcal{D}$ be the exact decoder, $\mathcal{D}_{\mathrm{run}}(Z) = \mathcal{D}(Z) + \eta(Z)$ its numerical realization, and $\Delta Z = \hat{Z} - Z^\star$. With $\chi = \int_0^1 \|\mathrm{D}\mathcal{D}(Z^\star + \lambda \Delta Z)\Delta Z\| / \|\Delta Z\| \, d\lambda$ for $\Delta Z \neq 0$ and $\chi = 0$ otherwise,

$$\left\|\mathcal{D}_{\mathrm{run}}(\hat{Z}) - X\right\| \le \chi \, \|\Delta Z\| + \|\mathcal{D}(Z^\star) - X\| + \|\eta(\hat{Z})\|. \tag{17}$$

The middle term is the frozen VAE's own reconstruction error. Frame selection does not increase the stacked Euclidean norm, while resizing multiplies it by its operator norm. A temporal decoder can spread an edit of one latent frame to several output frames even when no executor feedback exists, which is why the returned-output replacement in Section 4.3 changes pixel scores without changing latent MSE.

B.3 Cost Accounting

Counting token-linear work before native patching,

$$\mathcal{W}_{\mathrm{dense}} = N\kappa_H S(1 + KR), \qquad \mathcal{W}_{\mathrm{KineTra}} = N\kappa_H(S + Kn_w) + KRS\kappa_E. \tag{18}$$

Their difference is $K\left[N\kappa_H(RS - n_w) - RS\kappa_E\right]$, which for positive N, κ_H, R, S is positive exactly when the limit in Equation (4) is below one; dividing both costs by $KRSN\kappa_H$ gives that limit. Full self-attention adds a term proportional to the squared token count, $S^2(1 + KR)^2$ for Dense and $(S + Kn_w)^2$ for KineTra, whose ratio tends to $(n_w/RS)^2$. For 961-frame LTX the token ratio is 5.8 and its square 33; the measured $11.2\times$ speedup of the transformer phase lies between the two (Section 4.1). Complete-video time further includes the native token layout and the VAE, which decodes every output frame in both representations. In action prediction, native patching gives equal transformer token counts, so only the executor term changes (Section 4.4).

B.4 Measured Errors and the Bounds

Figure 9 evaluates the bounds on CogVideoX Physics-IQ at $R = 4$ and width 512, with five segments and 20 valid future positions, over 66 scenes, three views, and three training seeds. Boundary error $e_k = \|\hat{B}_k - \bar{B}_k\|_2$ uses the encoded rollout as reference, and raw-query error uses the target latent and therefore includes $\zeta_{k,j}$. Latent RMS divides by $\sqrt{SC}$, and pixel RMS by the square root of the valid RGB count before resizing. Scene curves average views and seeds, and query curves average the 594 paths at each position. In every panel the bound stays above the measured error, and the recursive bound Equation (13), which uses the measured gain of each segment, is about three times tighter than the uniform bound Equation (14).

C How Generated Weights Change the Output

Appendix B bounds the output through ranks and gains. This appendix evaluates those quantities with the tensor dimensions of the CogVideoX executor ($R = 4$, width 512): how many weight directions change the output (the local ranks r_k of Theorem 1), how the recurrent boundary couples segments, and how large the generated matrices and their effects are (Equation (16)).

C.1 Local and Complete Sensitivity

Operators. The analytical operators below have the executor's tensor dimensions. A local operator maps one segment's weight change to its valid output latents with the entering boundary fixed; composing it with the encoder derivative restricts it to the directions the encoder uses (encoder directions), while free directions are arbitrary weight changes. A complete operator also propagates the change through later boundaries and stacks all valid future outputs, with input and output

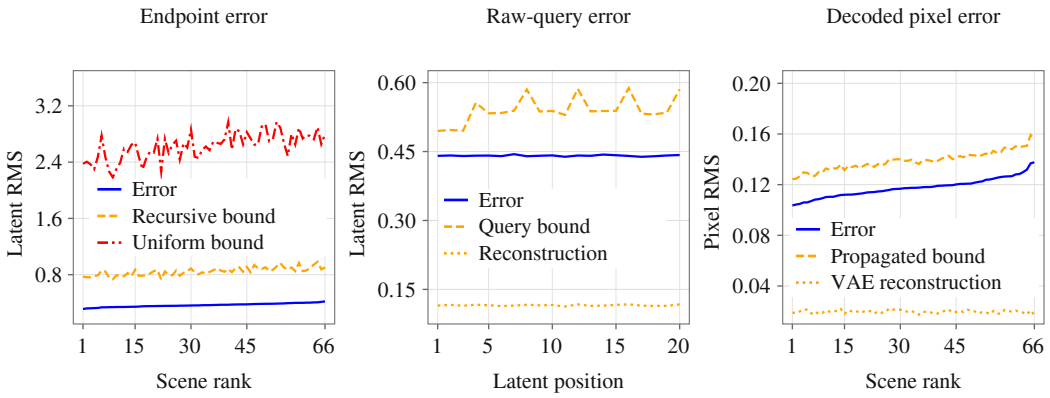

Figure 9: **Measured errors along execution and their bounds.** CogVideoX, Physics-IQ, $R = 4$. Left: boundary error with the recursive bound Equation (13) and the uniform bound Equation (14). Middle: raw interior error, its bound Equation (15), and the reconstruction residual. Right: decoded pixel error, the propagated bound Equation (17), and the frozen VAE's reconstruction error. All errors are RMS; the left and right panels sort scenes independently.

dimensions D_{in} and D_{out}. The boundary coupling is $\Lambda_{ij} = \beta_j \prod_{j'=j+1}^{i} \alpha_{j'}$ for $i \geq j$, with local weight gain β_j and boundary gain $\alpha_{j'}$ as in Theorem 2. We estimate 256 singular values with 16 oversampling directions, two re-orthogonalized power iterations, and 256 independent residual probes, and bound ranks with the threshold $\max(D_{\text{in}}, D_{\text{out}})\, \epsilon_{\text{FP32}}\, \sigma_{\max}$, including the uncertainty of $\sigma_{\max}$.

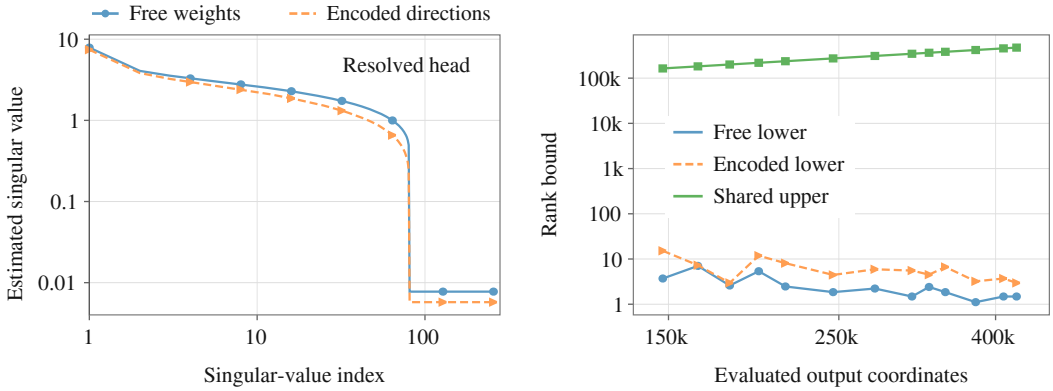

Figure 10: **Complete-output spectra and rank bounds.** Analytical operators with CogVideoX dimensions, $R = 4$, width 512. Left: all 256 estimated singular values, averaged over 198 Physics-IQ views and three seeds at encoded references, for free weight directions and encoder directions. Right: rank bounds for 90 DAVIS operators per type (30 sources, three seeds), grouped by their 13 valid output dimensions; upper bounds range over 147,456–425,984 coordinates.

Spectra and ranks. Each local spectrum has 16 leading directions followed by a repeated lower level (Figure 12a). The complete spectrum over the five segments of a Physics-IQ video keeps a resolved head of about 80 directions before the same drop (Figure 10, left), five times the local count: each segment adds its own directions, as the block lower triangular structure behind Theorem 1 states. Encoder directions follow the same spectrum with slightly smaller values, so the encoder uses the directions that change the output. The rank intervals (Figure 10, right) bound the exact ranks of Theorem 1 from both sides at FP32 precision. Figure 11 shows the same structure on a reduced example: propagation through later boundaries changes the spatial response pattern, and restricting to encoder directions changes which pattern is tested.

C.2 Generated Matrices and Their Effect

Figure 12(a) normalizes first-segment spectra per operator over 594 CogVideoX Physics-IQ paths (198 views, three seeds). Panels (b) and (c) cover all layers, groups, segments, views, and seeds of

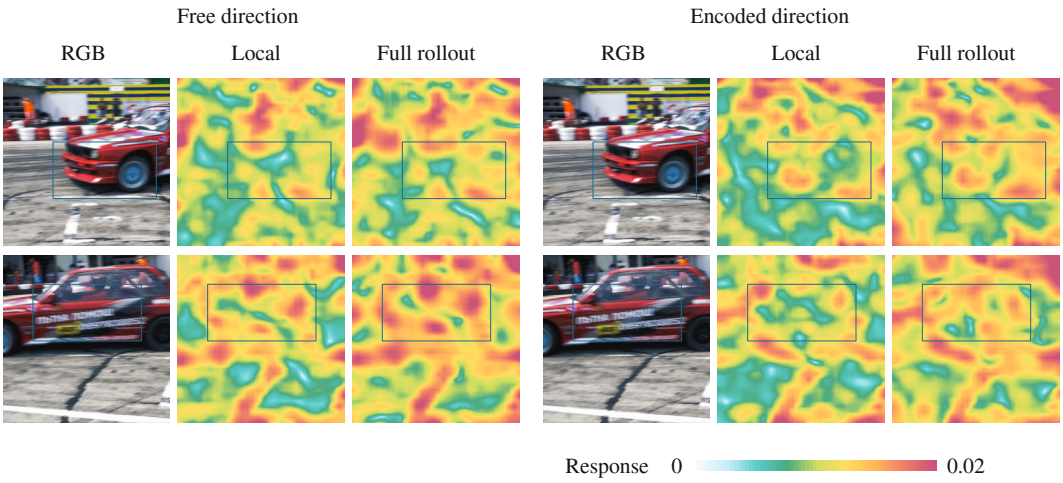

Figure 11: **Local and full-rollout responses.** A reduced analytical executor on DAVIS *drift-straight*, frames 29 and 34: four segments, $32 \times 32 \times 3$ state, 128 weights per segment. Unit directions are a free direction in the 512 weights of all four segments and a normalized direction in a 32-dimensional encoder space. Local responses fix the entering boundary; full responses propagate upstream derivatives. Maps show the channel-RMS Jacobian–vector response on a square-root scale (0–0.02), verified by centered differences with step 10^{-5}.

SANA, LTX, and CogVideoX on Physics-IQ at $R = 4$ and width 512: 836,352 generated 32×32 matrices (128 per segment over $3 + 3 + 5$ segments and 594 paths) and 3,345,408 queries. Panel (b) compares the raw maximum $\|w\|_\infty$ with the mapped maximum $\|M(w)\|_\infty/|\gamma_\ell|$ after BF16 rounding, averaging equal raw maxima; horizontal lines give the averages for encoded matrices.

For shared group features V with n_V rows, let $\Sigma = V^\top V/n_V$ and $\Delta M = M_{\text{gen}} - M_{\text{ref}}$. Then $\|V\Delta M\|_F^2/n_V = \operatorname{tr}(\Delta M^\top \Sigma \Delta M) \le \lambda_{\max}(\Sigma)\,\|\Delta M\|_F^2$. Panel (c) plots the measured response and this bound against the raw distance $\|w_{\text{gen}} - w_{\text{ref}}\|_2$; the bound exceeds the response by a factor of 3.65 on average, so matrix-level differences overstate the resulting feature change.

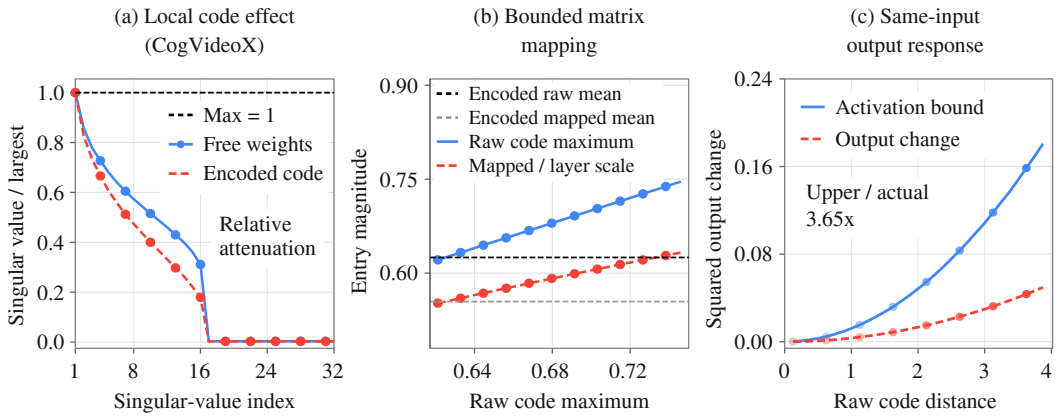

Figure 12: **From weight directions to output change.** (a) First 32 local singular values at encoded references. (b) Raw and mapped matrix amplitudes. (c) Same-input response and its activation-based bound.

D Generation Quality: Protocols and Complete Results

This appendix supports Section 4.1. Each subsection states the protocol of one benchmark family and then reports component scores for its listed configurations, so that the aggregate changes in the main text can be traced to individual scores. Appendix D.1 covers VBench, Appendix D.2 composition, motion, and preference, and Appendix D.3 Physics-IQ.

D.1 VBench and the Long-Video Timeline

Protocol. VBench uses all 946 published prompts with their dimension mapping and one video per prompt and training seed, that is, 946 videos per trained model and 2,838 per configuration. Short and long videos have 129 and 961 frames at 512×768 and 16 fps. Quality is the official normalized mean of seven appearance and motion dimensions, with dynamic degree weighted by 0.5; Semantic averages the nine normalized condition-following dimensions.

Timeline. Figure 5 compares the default LTX model with Dense on VBench-Long (Huang et al., 2026). Subject consistency, background consistency, dynamic degree, and text–video consistency use 72, 86, 72, and 93 prompts, with one video per prompt and seed. Subject and background scores measure consistency within a clip, dynamic degree is the percentage of clips classified as dynamic, and text–video consistency is the benchmark's video–text similarity. The first three panels use 31 two-second clip starts and the last seven ten-second starts; each point averages the videos and seeds at that start, and the last incomplete interval borrows adjacent frames. The within-clip differences in Figure 5 are $+1.48$ and $+0.60$ points for subject and background consistency; the complete scores in Table 5, which add cross-clip consistency, differ by $+1.49$ and $+0.64$. For the complete scores, the paired prompt-bootstrap 95% intervals of the change are $[0.87, 2.11]$ (subject), $[0.35, 0.94]$ (background), $[-2.10, 2.65]$ (dynamic degree, paired change $+0.25$), and $[-1.13, 1.08]$ (text–video consistency, -0.06). Consistency thus improves while the amount of motion stays the same.

Table 5: **Complete VBench-Long component means.** The same 961-frame protocol is used for all bodies. Higher is better for every displayed official score. Bold marks the better of Dense and KineTra for each metric and body. Aggregate scores and complete-pipeline costs are in Table 2. Model sources are in Table 4.

Metric	SANA 2.0 / 5B		LTX 0.9.8 / 13B		CogVideoX1.5 / 5B		Self Forcing / 1.3B	
	Dense	KineTra	Dense	KineTra	Dense	KineTra	Dense	KineTra
Quality components								
Subject consistency	93.41	**93.42**	91.78	**93.27**	**92.25**	92.18	91.10	**91.47**
Background consistency	**96.21**	96.05	95.43	**96.07**	**95.63**	95.62	**94.81**	94.69
Temporal flickering	**98.36**	98.25	**98.12**	97.98	98.17	**98.28**	**97.73**	97.67
Motion smoothness	**97.66**	97.53	97.18	**97.57**	**97.31**	97.23	**97.06**	96.78
Dynamic degree	68.37	**68.60**	60.37	**60.63**	**63.63**	63.37	**67.27**	67.24
Aesthetic quality	**62.71**	62.64	**61.10**	60.96	**61.57**	61.34	**59.90**	59.83
Imaging quality	69.04	**69.28**	**67.37**	67.32	67.86	**67.89**	**65.29**	65.16
Semantic components								
Object class	**93.45**	93.23	92.24	**92.36**	**92.33**	92.32	**91.57**	91.24
Multiple objects	66.20	**66.40**	**62.48**	61.85	60.91	**61.77**	59.76	**59.99**
Human action	94.18	**94.22**	**92.81**	92.79	**92.70**	92.39	92.10	**92.24**
Color	87.40	**87.43**	**86.30**	86.16	**86.38**	86.27	**84.68**	84.65
Spatial relationship	**62.12**	62.08	**57.78**	57.56	**57.92**	57.64	55.13	**55.67**
Scene	59.46	**59.68**	56.37	**57.45**	57.38	**58.05**	55.88	**55.93**
Appearance style	**26.04**	25.94	25.43	**25.45**	25.46	**25.47**	24.88	**24.98**
Temporal style	27.04	**27.10**	**26.05**	25.88	26.23	**26.26**	**25.74**	25.69
Text–video consistency	28.12	**28.24**	**27.22**	27.16	**27.47**	27.42	**26.66**	26.65

All components. Table 5 lists the 16 VBench-Long components for the four default Dense/KineTra pairs. The two largest differences are LTX's subject consistency ($+1.49$) and scene score ($+1.08$); every other component changes by less than one point, and the scene score rises in all four families.

D.2 Composition, Motion, and Human Preference

Protocols. T2V-CompBench (Sun et al., 2025a) reports seven category scores over 1,400 prompts, 200 per category, without an overall score. Consistent attributes, actions, and interactions use six frames and three evaluator repeats; dynamic attributes use the endpoints and six internal frames; spatial relations and numeracy use 16 frames; and motion binding samples at 8 fps. VMBench (Ling et al., 2025) evaluates all 1,050 prompts in the short format and reports commonsense adherence (CAS), motion smoothness (MSS), object integrity from detected keypoints (OIS), perceptible subject-motion amplitude (PAS), and temporal coherence (TCS), each multiplied by 100, plus their equal-weight average. MovieGen Video Bench (Polyak et al., 2024) uses all 1,003 prompts with 256-frame videos at 24 fps; a model that computes 257 frames returns 256 valid ones. Each prompt contributes one output per method, left and right positions are randomized and anonymous, and each pair receives three judgments on each of seven axes, with ties allowed. We average the three

Paraglider: launch to flight



Motorcycle: approach to landing



Figure 13: **Long motion trajectories.** Top: a paraglider from launch into flight. Bottom: a motorcycle from approach to landing. Frames run from left to right and show changes in pose, viewpoint, and scene layout over the sequence.

win/tie/loss indicators within a prompt and then average prompts equally; net preference is the win percentage minus the loss percentage.

Table 6: **Complete T2V-CompBench comparison.** Each category uses 200 prompts and its official evaluator (Sun et al., 2025a); all scores are in 10^{-2} units. Backbone sources are in Table 4; native methods follow Atanov et al. (2026); Yuan et al. (2025); Mahapatra et al. (2025). Bold marks the best value within each matched body or the ProMAG pair.

Method	Attribute binding		Relations		Actions / interactions / count		
	Consist. ↑	Dynamic ↑	Spatial ↑	Motion ↑	Action ↑	Object ↑	Numeracy ↑
SANA-Video 2.0 (Chen et al., 2026a) / 5B							
Dense	**74.66**	12.78	63.52	**41.75**	**68.84**	66.11	**45.27**
KineTra	74.21	**13.00**	**63.58**	41.54	68.39	**66.85**	45.03
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev							
Dense	70.20	8.74	**59.48**	36.05	**63.48**	**62.30**	42.33
KineTra	**70.28**	**8.99**	58.99	**36.45**	62.79	62.28	**42.54**
CogVideoX1.5 (Yang et al., 2025) / 5B							
Dense	**71.91**	7.45	**58.79**	**34.35**	**61.31**	**62.82**	44.08
KineTra	71.76	**7.78**	58.62	34.12	60.43	62.65	**44.47**
Self Forcing (Huang et al., 2025) / Wan2.1-T2V / 1.3B							
Dense	67.96	**8.40**	**56.64**	34.92	**61.46**	59.70	**40.91**
KineTra	**68.69**	7.98	56.10	**35.27**	61.44	**59.86**	40.62
External native generators; formats as in Table 1							
VideoFlexTok (Atanov et al., 2026)	65.60	6.06	54.48	30.41	57.87	57.18	39.42
DLFR-Gen (Yuan et al., 2025)	71.18	9.78	59.87	39.05	63.48	63.22	43.05
ProMAG (Mahapatra et al., 2025) 4×	**71.55**	**10.22**	**61.19**	**39.58**	**64.94**	**64.09**	**43.63**
ProMAG (Mahapatra et al., 2025) 16×	69.32	9.02	58.58	37.52	62.13	61.47	41.87

Composition. Table 6 reports all seven categories for the four matched bodies and four native external generators. Dynamic attribute binding, which scores attributes that change during a video, rises in the three non-causal families (12.78 to 13.00, 8.74 to 8.99, and 7.45 to 7.78) and falls in the causal family (8.40 to 7.98). The other categories change by less than one point.

Motion. Table 7 resolves the VMBench average for the listed Dense, KineTra, DiCache, and native-generator configurations. For SANA, motion smoothness rises from 77.98 to 78.92 while temporal coherence falls from 94.73 to 94.04, so the average moves by only $+0.04$. For LTX and CogVideoX the five components each change by less than 0.5 points. Figure 14 shows a generated sequence with fast changes of articulated pose, where the interior of each segment carries most of the motion.

Human preference. Table 8 reports mean net preference on each axis, and Table 9 the overall-quality outcomes. Each axis has 3,009 judgments per pair and seed (1,003 prompts, three judgments each). Mean overall win/tie/loss percentages are 35.08/28.97/35.95 for SANA, 36.88/26.87/36.25 for LTX, and 34.73/30.25/35.02 for CogVideoX, so wins and losses differ by less than one point in every family. For overall net preference, training-seed 95% intervals are $[-12.95, 11.22]$, $[-7.75, 9.01]$, and $[-3.62, 3.05]$, and prompt-bootstrap intervals are $[-2.60, 0.88]$, $[-1.17, 2.49]$, and $[-1.91, 1.34]$; all include zero. Raters also disagree often: with n_{ia} ratings of outcome $a \in \{\text{win}, \text{tie}, \text{loss}\}$ for prompt i, observed agreement $P_o = N_p^{-1} \sum_{i,a} n_{ia}(n_{ia} - 1)/6$ and chance agreement $P_e = \sum_a \left(\sum_i n_{ia}/3N_p\right)^2$ over N_p prompts give a chance-adjusted agreement $(P_o - P_e)/(1 - P_e)$ between

Table 7: **VMBench motion components for the listed configurations.** Higher is better for all five components and the official average (Ling et al., 2025). Model and external-method sources are in Table 1; metric definitions are in Appendix D.2. Bold marks the best displayed value within each matched body or the ProMAG pair.

Method	VMBench components					Average
	CAS ↑	MSS ↑	OIS ↑	PAS ↑	TCS ↑	
SANA-Video 2.0 (Chen et al., 2026a) / 5B						
Dense	**69.04**	77.98	**80.73**	34.75	**94.73**	71.45
KineTra	68.91	**78.92**	80.48	**35.10**	94.04	**71.49**
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev						
Dense	65.66	**77.22**	78.91	**29.40**	**93.90**	**69.02**
DiCache $\delta = 0.1$ (Bu et al., 2026)	65.06	76.82	**79.14**	29.20	93.60	68.76
DiCache $\delta = 0.2$ (Bu et al., 2026)	65.25	76.39	79.00	29.13	93.00	68.55
KineTra	**65.72**	76.81	79.13	29.19	93.74	68.92
CogVideoX1.5 (Yang et al., 2025) / 5B						
Dense	64.53	**74.30**	**78.72**	**31.43**	**92.90**	**68.37**
KineTra	**64.83**	74.12	78.50	31.39	92.89	68.34
External native generators; formats as in Table 1						
VideoFlexTok (Atanov et al., 2026)	60.77	79.15	78.63	18.05	93.47	66.01
DLFR-Gen (Yuan et al., 2025)	66.56	77.68	80.11	32.31	93.64	70.06
ProMAG (Mahapatra et al., 2025) 4×	**67.25**	**77.39**	**81.06**	**33.38**	**93.56**	**70.53**
ProMAG (Mahapatra et al., 2025) 16×	64.69	75.84	79.37	31.01	92.84	68.75

Wind-up, sweep and spin



Inversion, rotation and recovery



Figure 14: **Fast articulated motion.** A breakdance sequence through wind-up, sweep, spin, inversion, rotation, and recovery; frames run from left to right along the first row and then the second.

Table 8: **Seven-axis human preference.** Each row is KineTra versus its matched Dense model under the MovieGen protocol (Polyak et al., 2024). Values are mean net preference in percentage points; each axis is judged separately. Bold marks the largest mean on each axis; training- and prompt-level intervals are given in Appendix D.2.

Matched pair	Overall ↑	Frame ↑	Natural ↑	Complete ↑	Text ↑	Realness ↑	Aesth. ↑
SANA 2.0 (Chen et al., 2026a) / 5B	-0.86	-0.83	+0.02	**+1.25**	+0.11	-2.39	-0.43
LTX 0.9.8 (HaCohen et al., 2024) / 13B	**+0.63**	**+0.69**	+0.21	+0.00	**+0.70**	**+1.61**	+0.21
CogVideoX1.5 (Yang et al., 2025) / 5B	-0.29	+0.43	**+0.42**	+0.19	-0.78	+0.68	**+0.51**

-0.028 and 0.033 across families, seeds, and axes. The study therefore detects no preference between KineTra and Dense, and individual judgments agree at chance level.

The bootstrap for human judgments resamples prompts and keeps the three ratings of a prompt together, so that the interval reflects prompt-level variation. Training-seed intervals use the three independently trained pairs and are wider, since they also contain the variation between training runs. The two kinds of interval answer different questions: whether the observed models differ on these prompts, and whether retraining would reproduce the difference.

Table 9: **Overall-quality judgment outcomes.** Mean win/tie/loss percentages for KineTra against Dense. Each pair uses 1,003 prompts and three ratings per prompt per seed; values are means over seeds.

Matched pair	Win ↑	Tie	Loss ↓
SANA 2.0 (Chen et al., 2026a) / 5B	35.08	28.97	35.95
LTX 0.9.8 (HaCohen et al., 2024) / 13B	**36.88**	26.87	36.25
CogVideoX1.5 (Yang et al., 2025) / 5B	34.73	30.25	**35.02**

D.3 First-Image Prediction on Physics-IQ

Protocol. Physics-IQ Original (Motamed et al., 2026) contains 66 scenes filmed from three views, 198 inputs in total. Each condition is the switch image with its description. We generate 80 future frames over five seconds at 16 fps, decode at 576×1024, and score at 540×960. The 150-frame reference is aligned to the prediction by the official endpoint-preserving interpolation at indices $149j/79$, $j = 0, \ldots, 79$, and the valid output contains the input and the 80 future frames. With the $8\times$ temporal VAE, default padding computes 97 frames; CogVideoX's $4\times$ temporal VAE gives 20 valid future latent positions. Intervals group views and reference takes by scene.

Table 10: **First-image prediction: quality and generation cost.** All 198 Physics-IQ Original inputs (Motamed et al., 2026) use the protocol in Appendix D.3. Four raw components are in 10^{-2} units; Original retains its official scale. The complete pipeline includes decoding. Backbone sources are in Table 4; caching follows Bu et al. (2026). Bold is within each matched body.

Method	Physics-IQ Original					Complete pipeline	
	S-IoU ↑	ST-IoU ↑	W-S-IoU ↑	MSE ↓	Original ↑	Sec. ↓	GiB ↓
SANA-Video 2.0 (Chen et al., 2026a) / 5B							
Dense	**24.85**	**13.25**	**5.70**	**1.34**	**23.35**	102.9	26.51
KineTra	24.78	13.21	5.59	1.36	23.22	**36.7**	**19.80**
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev							
Dense	23.04	**11.99**	4.92	1.47	**21.06**	235.0	51.80
DiCache $\delta = 0.1$ (Bu et al., 2026)	22.97	11.88	4.93	1.48	20.95	163.2	52.55
DiCache $\delta = 0.2$ (Bu et al., 2026)	22.65	11.85	4.73	1.49	20.64	126.7	52.47
KineTra	**23.16**	11.81	**4.98**	**1.45**	**21.06**	**71.4**	**39.97**
CogVideoX1.5 (Yang et al., 2025) / 5B							
Dense	**24.16**	12.70	**5.42**	1.40	**22.43**	600.7	46.90
KineTra	24.08	**12.74**	5.39	**1.39**	22.41	**153.5**	**26.88**

Results. Table 10 decomposes the Original score into spatial IoU (where motion occurs), spatiotemporal IoU (where and when), weighted spatial IoU, and pixel MSE, together with time and memory; the LTX block adds matched caching. KineTra is 2.8–3.9× faster, and every component changes by at most 0.18 in the displayed units. With caching, LTX Dense takes 126.7–163.2 seconds and loses up to 0.42 Original points, whereas KineTra takes 71.4 seconds with no loss. Figure 15 shows four kinds of physical process in the benchmark.

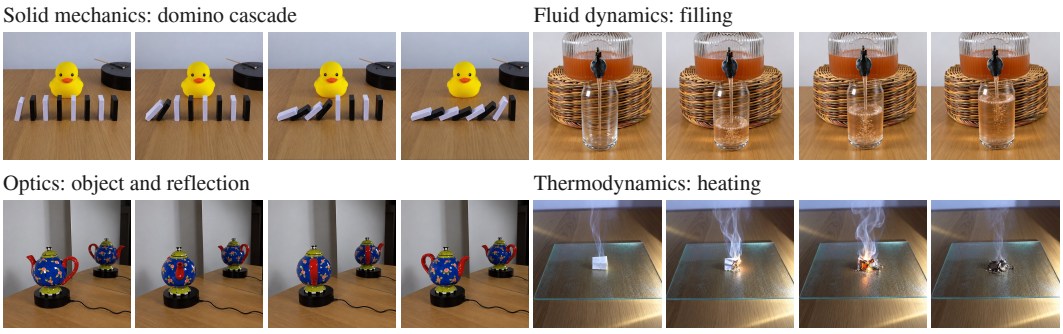

Figure 15: **Physical processes in first-image prediction.** A domino cascade, fluid filling, an object with its reflection, and heating; each four-frame group runs from left to right. Table 10 scores the complete predicted trajectories.

E Representation: Protocols and Complete Results

This appendix supports Section 4.2. It measures how much information the weights keep (Appendix E.1), how well they can be generated (Appendix E.2), and how segment length, width, and boundary training shape the error of a segment (Appendix E.3).

E.1 Reconstruction

Protocol. UCF101 (Soomro et al., 2012) uses all 3,783 validation sources with one fixed native-rate 16-frame window per source at 128^2; Kinetics-600 (Carreira et al., 2018) uses the complete 29,791-source CVDF validation set with the same windows; MSR-VTT (Xu et al., 2016) uses all 2,990 test

sources at 128^2; and DAVIS 2017 (Pont-Tuset et al., 2017) uses all 30 validation sources at 256^2 with their complete valid durations. PSNR is computed from frame MSE on RGB values in $[0, 1]$, averaged within a source and then over sources. SSIM (Wang et al., 2004) uses the RGB Gaussian-window implementation (window 11, $\sigma = 1.5$, population covariance, data range one), and LPIPS (Zhang et al., 2018) uses version 0.1 with the VGG backbone on inputs in $[-1, 1]$. Reconstruction FVD (Unterthiner et al., 2018) compares I3D features (400-dimensional, before softmax) of original and reconstructed nonoverlapping 16-frame windows, excluding padding. Weights are produced by the encoder, so these scores isolate the representation from generation.

Table 11: **Reconstruction across four complete evaluation source sets.** PSNR is in dB; SSIM and LPIPS retain their raw scales. Protocols follow Appendix E.1. Bold marks the best value within each frozen-VAE/KineTra pair or the ProMAG sweep. Fixed deterministic VAEs have a single value; learned or stochastic configurations show their mean. Native systems with narrower interfaces are separated in Table 13 and Table 12.

Representation	UCF101			Kinetics-600			MSR-VTT			DAVIS		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
SANA-Video 2.0 (Chen et al., 2026a) / 5B												
Frozen VAE	**35.72**	**0.967**	**0.022**	**35.06**	**0.960**	**0.027**	**33.68**	**0.953**	**0.034**	**34.42**	**0.962**	**0.030**
KineTra	35.45	0.965	0.023	34.83	0.958	0.029	33.42	0.951	0.035	34.06	0.961	0.032
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev												
Frozen VAE	**34.91**	**0.960**	**0.029**	**34.23**	**0.955**	**0.034**	**32.54**	**0.943**	**0.043**	**33.21**	**0.952**	**0.038**
KineTra	34.74	0.959	**0.029**	33.94	0.952	0.036	32.23	0.941	0.046	32.81	0.948	0.040
CogVideoX1.5 (Yang et al., 2025) / 5B												
Frozen VAE	**36.48**	**0.972**	**0.018**	**35.82**	**0.966**	**0.024**	**34.42**	**0.959**	**0.030**	**35.18**	**0.968**	**0.025**
KineTra	36.14	0.970	0.020	35.43	0.964	0.025	34.02	0.957	0.031	34.78	0.965	0.028
ProMAG / 16-channel VAE / temporal compression sweep (Mahapatra et al., 2025)												
ProMAG (Mahapatra et al., 2025) 4×	**36.07**	**0.969**	**0.021**	**35.54**	**0.965**	**0.025**	**34.02**	**0.957**	**0.031**	**34.81**	**0.967**	**0.027**
ProMAG (Mahapatra et al., 2025) 8×	34.68	0.959	0.033	34.16	0.953	0.038	32.61	0.944	0.044	33.26	0.955	0.039
ProMAG (Mahapatra et al., 2025) 16×	32.43	0.937	0.055	31.96	0.931	0.062	30.21	0.920	0.072	31.39	0.934	0.066

Results. Table 11 shows that KineTra stays within 0.17–0.40 dB PSNR, 0.001–0.004 SSIM, and 0.000–0.003 LPIPS of its frozen VAE across all four datasets and three bodies. ProMAG, which compresses time inside the tokenizer, loses 3.4–3.8 dB between its 4× and 16× settings. Table 13 and Table 12 report native tokenizers at their own interfaces: NeRV, LARP, and CoordTok at 128^2, VideoFlexTok's K600 tokenizer at 128^2 and its Panda tokenizer at 256^2 on DAVIS. NeRV, CoordTok 3072, ProMAG, and the VideoFlexTok generator are trained in three replicas; LARP, CoordTok 1280, and the VideoFlexTok tokenizers use released parameters. NeRV's native training set contains all 13,320 UCF videos, so its UCF reconstruction includes training sources.

For VideoFlexTok the number of tokens sets the trade-off inside one interface: on DAVIS, going from 32 to 256 tokens raises PSNR from 29.93 to 34.56 dB (Table 12), and on the 128^2 sets, going from 32 to 128 tokens raises it by 3.9–4.0 dB (Table 13). Each native system is evaluated at the resolution, source set, and training exposure it was built for.

Table 12: **VideoFlexTok Panda reconstruction.** The d18–d28 tokenizer (Atanov et al., 2026) uses its native 256×256 interface on all 30 DAVIS sources. Values are means over stochastic evaluations; PSNR is in dB.

Representation	DAVIS		
	PSNR ↑	SSIM ↑	LPIPS ↓
32 tokens	29.93	0.919	0.108
256 tokens	**34.56**	**0.961**	**0.031**

Table 13: **Native** 128×128 **reconstruction interfaces.** The three metrics and source-set protocols match Table 11. †: NeRV native training includes all UCF videos, so this UCF result includes training sources. Bold marks the better token count within each system.

Representation	UCF101			Kinetics-600			MSR-VTT		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRV-Diffusion-L / 55M representation (Ren et al., 2026)									
NeRV-L† (Ren et al., 2026)	32.87	0.951	0.041	32.28	0.945	0.047	30.59	0.929	0.057
LARP-L-Long / 173M tokenizer (Wang et al., 2025)									
LARP-L-Long (Wang et al., 2025)	33.29	0.954	0.037	32.71	0.949	0.042	30.98	0.934	0.053
CoordTok / coordinate-based tokenizer (Jang et al., 2025)									
1,280 tokens	30.17	0.919	0.069	29.63	0.907	0.078	29.02	0.903	0.082
3,072 tokens	**31.43**	**0.935**	**0.055**	**30.74**	**0.925**	**0.064**	**30.20**	**0.921**	**0.067**
VideoFlexTok K600 / d18–d18 tokenizer (Atanov et al., 2026)									
32 tokens	29.89	0.902	0.095	30.30	0.918	0.081	28.46	0.887	0.119
128 tokens	**33.84**	**0.953**	**0.039**	**34.28**	**0.961**	**0.032**	**32.31**	**0.946**	**0.049**

Examples. Figure 16 inspects fine geometry on a moving subject, the helmet outline and the front wheel, for the frozen VAE, KineTra, and the linear-interior control. At a single frame the three reconstructions differ little, as expected from the 0.36–0.40 dB DAVIS gap in Table 11; the larger loss of the linear-interior control in Table 3 comes from the frames between boundaries, where it must interpolate. Figure 17 shows the two demands that Figure 2 identifies, content that becomes visible and fast local change: in both cases the carried frame lacks the new content, so the generated weights must supply it. Figure 18 shows a generated sample in which foreground motion exposes the street behind a car.

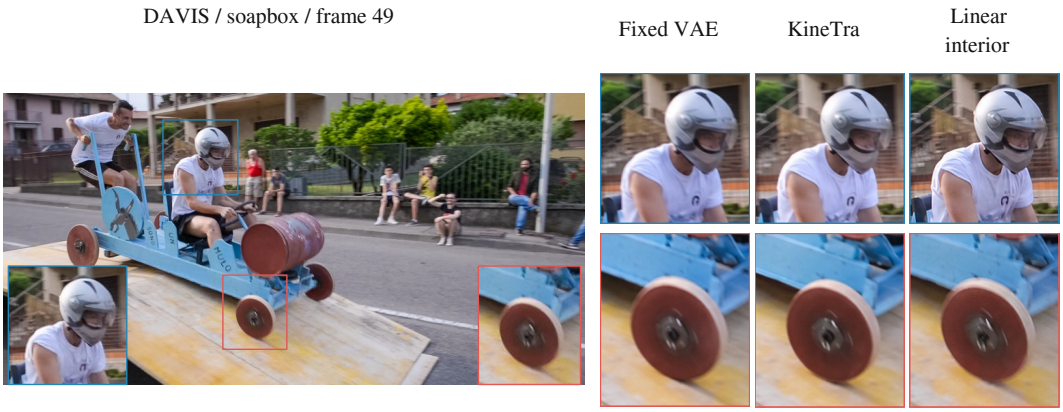

Figure 16: **Fine geometry on a moving subject.** DAVIS *soapbox*, frame 49. The full view gives the scene, and two marked regions isolate the helmet outline and the front wheel. Crops compare the frozen VAE, KineTra, and the linear-interior control of Table 3 with the same crop coordinates and display scale in each region.

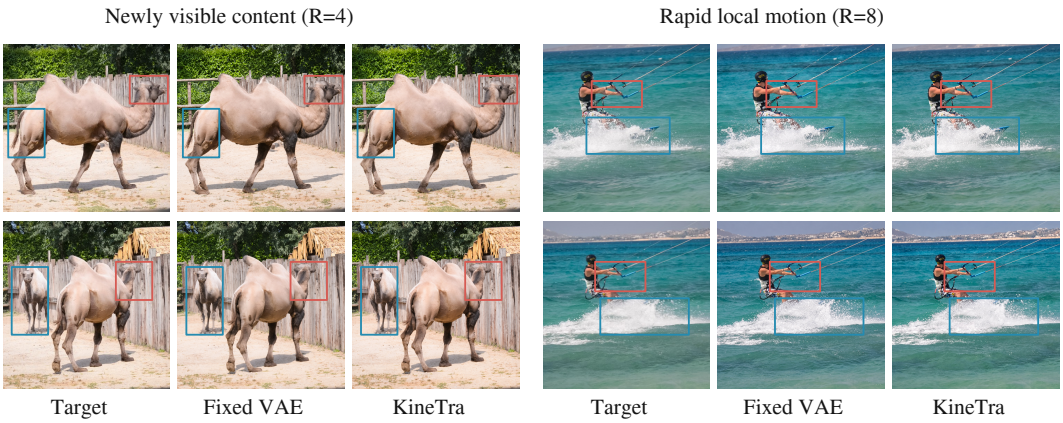

Figure 17: **Newly visible content and fast local change.** DAVIS reconstructions by the frozen VAE and KineTra. Left: *camel*, frames 0 and 53, $R = 4$; a second animal becomes visible as the first turns. Right: *kite-surf*, frames 0 and 29, $R = 8$; the pose and splash change quickly. Boxes mark corresponding regions. All methods share each 480×480 source crop; the camel crop shifts by 68 pixels between times to keep both animals in view.

E.2 Class-Conditioned Generation

Protocol. UCF class generation compares 10,000 generated videos with 10,000 training windows. For our class models, one group per class provides 526 validation videos from the 9,537 training sources, which leaves 9,011 for training and keeps the 3,783 test sources separate.

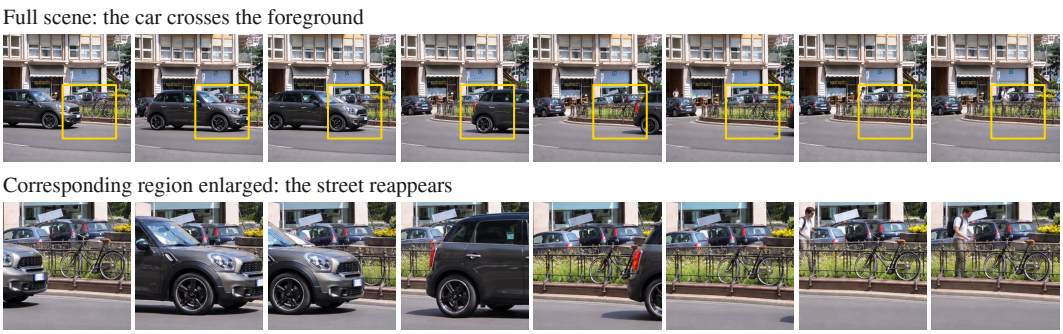

Figure 18: **Foreground motion exposes hidden content.** A car crosses the foreground while the street behind it becomes visible. The lower row enlarges the regions marked in the upper row; columns show the same sampled times. The exposed street is the new content that the generated weights must supply.

Results. Table 14 tests whether generated weights are as easy to model as dense latents. KineTra's gFVD is within 1.0 of Dense on SANA and CogVideoX and 2.0 lower on LTX (paired change -2.04, interval $[-4.35, 0.27]$). LARP uses its autoregressive sampler with guidance 1.25; its published gFVD 57 follows the same 10,000-sample protocol and differs from our 56.1 mean. At this short length KineTra removes little transformer work, so its time ranges from a 0.6% increase on SANA to an 18% reduction on CogVideoX, with slightly higher memory.

Table 14: **UCF class generation.** All rows use the same 10,000-sample metric protocol; native training and samplers differ. Sources: NeRV (Ren et al., 2026), LARP (Wang et al., 2025), and DiCache (Bu et al., 2026). Bold is within each matched body.

Method	gFVD ↓	Sec. ↓	GiB ↓
SANA-Video 2.0 (Chen et al., 2026a) / 5B			
Dense	**55.1**	**8.519**	**16.74**
KineTra	55.2	8.573	17.57
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev			
Dense	61.9	12.037	**31.58**
DiCache $\delta = 0.1$ (Bu et al., 2026)	62.0	9.812	31.95
DiCache $\delta = 0.2$ (Bu et al., 2026)	65.5	**8.338**	31.93
KineTra	**59.9**	10.559	32.47
CogVideoX1.5 (Yang et al., 2025) / 5B			
Dense	**53.2**	17.598	**18.98**
KineTra	54.2	**14.438**	19.74
Native class-conditioned generators			
NeRV-L (Ren et al., 2026)	81.3	18.161	6.71
LARP-L-Long (Wang et al., 2025)	56.1	19.503	7.55



Figure 19: **Class-conditioned samples.** Eight UCF101 classes from left to right (biking, horse riding, salsa spin, dog walking, surfing, skydiving, rafting, skateboarding), two samples per class.

E.3 Segment Length, Width, and Boundary Training

Figure 6 varies segment length, width, and code sharing with the VAE, output format, and solver fixed within each family; each setting uses 72 subject-consistency prompts per seed. Width changes two things at once: a wider executor has more generated coefficients per segment, which raises capacity, and more computation per position, which raises κ_E in Equation (4). Tying the weights keeps the executor and removes the per-segment capacity. Each curve therefore traces a quality–time trade-off within one family.

Segment length. Figure 7 isolates the load that a longer segment places on one weight set. The same 33-frame sequence is fitted with segments of two, four, and eight frames, each with exact endpoints

and the same 768-coefficient interior. As segments lengthen, the error between the exact endpoints grows and spreads over more of each segment, because the same interior budget must describe more change; this is the per-segment side of the R trade-off in Section 4.2.

Boundary training. Figure 20 illustrates the role of the feedback gain α in Theorem 2 with a controlled recurrence. Both rows start from the same state and receive the same local mismatch at every step; they differ only in a prescribed gain, 0.98 or 0.50. With the high gain, the mismatch accumulates over time into large errors across the moving cow; with the low gain it stays small. Training on predicted boundaries exposes the executor to its own mismatch during Stage 1; the trained comparison is the teacher-boundary control in Table 3, which loses 2.08 dB PSNR on SANA.

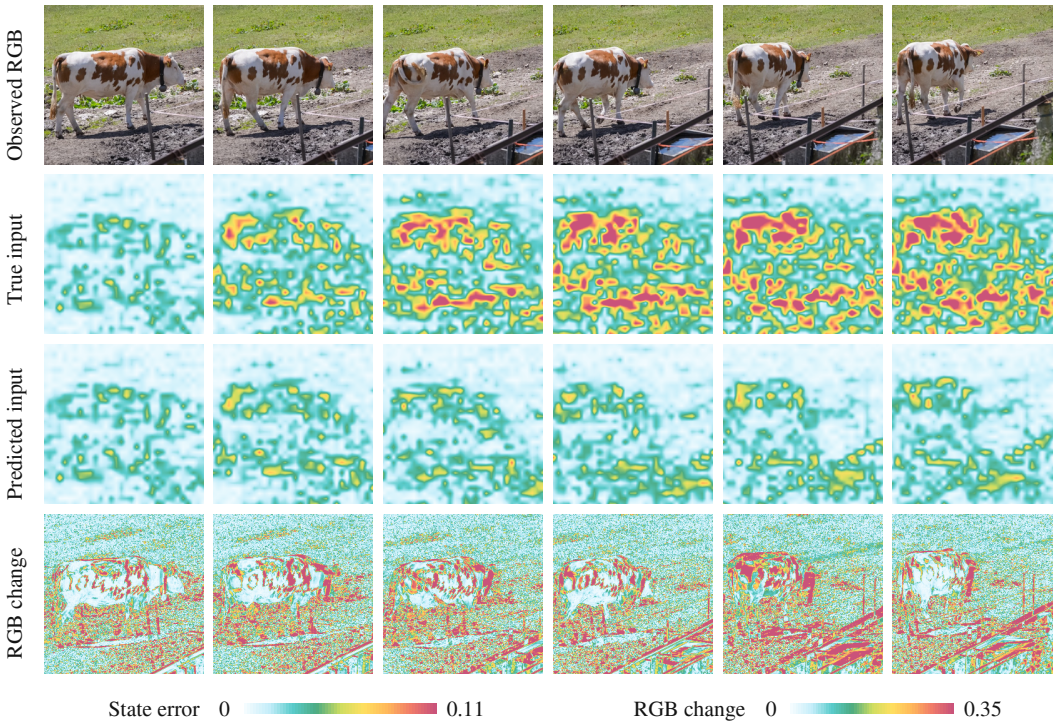

Figure 20: **Boundary mismatch accumulates with high feedback gain.** DAVIS *cows*, frames 10, 31, 52, 72, 93, and 103, identical 256×256 views. Below the observed RGB, two rows show RMS state error against the same reference for a controlled recurrence with feedback gain 0.98 (true input) and 0.50 (predicted input), driven by identical local mismatch; both use a 0–0.11 scale. The last row shows RGB change between sampled frames (0–0.35) as motion context.

F Boundary Interventions

This appendix supports Section 4.3. Theorem 2 separates two tensors at each internal boundary: the frame fed to the next segment, which enters all later executor states, and the frame returned to the VAE, which enters only decoding. The interventions below change one tensor at a time with everything else fixed.

Results. With fixed generated weights, we replay the rollout while replacing, at every internal boundary, either the frame fed to the next segment (state) or the frame returned to the VAE (output) by its reference value (Table 15). Replacing the state lowers raw latent MSE on Physics-IQ from 0.247 to 0.204 for LTX and raises the Physics-IQ Original score from 21.06 to 25.57; SANA and CogVideoX gain 5.1 and 5.8 points, and DAVIS latent MSE falls by 28–35% on all three bodies. Replacing only the output leaves every later executor state, and hence latent MSE, unchanged, yet raises Original by 0.3 to 1.0 points through decoding, since the VAE decodes neighboring frames jointly. Replacing both gives the highest Original score on every body, and the two effects add (Table 16). Accuracy is thus decided at the carried boundary, whose error enters every later segment, scaled by the later path gains (Appendix B.4).

Design. Let $Y(I_{\text{st}}, I_{\text{out}})$ be the Physics-IQ Original score with state-replacement indicator $I_{\text{st}} \in \{0, 1\}$ and output-replacement indicator $I_{\text{out}} \in \{0, 1\}$. Both replacements use the same reference boundary at every internal boundary that is followed by a valid output. All four conditions keep the same sampled weights, initial condition, checkpoint, and valid frames, and later segments are replayed with their original weights. Holding the weights fixed is what isolates the two paths: resampling them after a replacement would change the later segments through their weights as well as through the boundary, and the effect of the boundary could no longer be read off. Replacing the fed frame and the returned frame by the same reference tensor also makes the two interventions equal in size, so their effects can be compared directly. Raw latent MSE sums squared error over valid latent coordinates, including the common first frame, and divides by their number; it is computed before the returned frame is replaced, so output-only replacement and no replacement share the same executed path.

Table 15: **Fixed-code boundary interventions.** DAVIS latent MSE is in 10^{-2} units; Physics latent MSE is in 10^{-1} units. Original retains its official scale. Reference boundaries replace only the specified tensors.

Replacement	DAVIS	Physics-IQ	
	MSE ↓	MSE ↓	Original ↑
SANA-Video 2.0 (Chen et al., 2026a) / 5B			
Neither	1.68	2.19	23.22
Output only	1.68	2.19	24.12
State only	**1.13**	**1.71**	28.30
Both	**1.13**	**1.71**	**29.11**
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev			
Neither	2.24	2.47	21.06
Output only	2.24	2.47	22.10
State only	**1.61**	**2.04**	25.57
Both	**1.61**	**2.04**	**26.50**
CogVideoX1.5 (Yang et al., 2025) / 5B			
Neither	1.36	1.88	22.41
Output only	1.36	1.88	22.72
State only	**0.88**	**1.32**	28.17
Both	**0.88**	**1.32**	**28.72**

Conditional effects. The state effects are $Y(1,0)-Y(0,0)$ and $Y(1,1)-Y(0,1)$, the output effects are $Y(0,1)-Y(0,0)$ and $Y(1,1)-Y(1,0)$, and the interaction is the difference between the two state effects. Under either setting of the other factor, state replacement raises Original by 4.4 to 6.0 points and output replacement by 0.3 to 1.0 points (Table 16). The scene-bootstrap 95% intervals of the interaction, $[-1.03, 0.86]$ for SANA, $[-1.00, 0.72]$ for LTX, and $[-0.71, 1.17]$ for CogVideoX, all include zero, so the two effects add.

Table 16: **Conditional replacement effects.** Mean Physics-IQ Original contrasts in score units. Paired values are differenced before rounding. Bold marks the larger of the two state effects and of the two output effects within each body.

Contrast	SANA 5B	LTX 13B	Cog 5B
$\Delta_{\text{state}\mid I_{\text{out}}=0}$	**+5.07**	**+4.52**	+5.77
$\Delta_{\text{output}\mid I_{\text{st}}=0}$	**+0.90**	**+1.04**	+0.32
$\Delta_{\text{state}\mid I_{\text{out}}=1}$	+4.99	+4.40	**+5.99**
$\Delta_{\text{output}\mid I_{\text{st}}=1}$	+0.81	+0.93	**+0.54**
$\Delta_{\text{interaction}}$	-0.09	-0.12	+0.23

Why output replacement changes pixels. Figure 21 isolates the decoder path with an analytical decoder. One returned frame is edited while all stored boundaries stay fixed. Decoding each frame separately confines the change to the edited frame, whereas a temporal decoder spreads it to the neighboring sampled frame. This is how output replacement can change the decoded video, and with it the Original score, without changing any executor state.

Per-source effects. Figure 22 shows the interventions for every source on CogVideoX ($R = 4$, width 512): the 30 DAVIS validation videos with encoded weights and the 66 Physics-IQ scenes, three views each, with generated weights, over three training seeds. Pixel MSE is $\|\widehat{X} - X\|_2^2 / n_{\text{pix}}$; for each source or scene we average over seeds and views and divide by the mean without replacement, so ratios above one mean larger pixel error after replacement. Both panels of a dataset use the same order, ascending in latent error without replacement. State replacement lowers latent MSE throughout the ranking, and the gap widens for the sources with the largest error, where more error has been carried forward. The pixel-MSE ratios of both replacements scatter around one, so pixel MSE alone does not separate the two paths, while latent MSE identifies the fed state as the tensor that carries the recurrent error.

G Cost and Acceleration

This appendix supports Section 4.4. It reports the regime in which KineTra removes no transformer tokens (Appendix G.1), where the time goes in each phase (Appendix G.2), and how KineTra combines with other accelerations (Appendix G.3). Timing follows Appendix A.4.

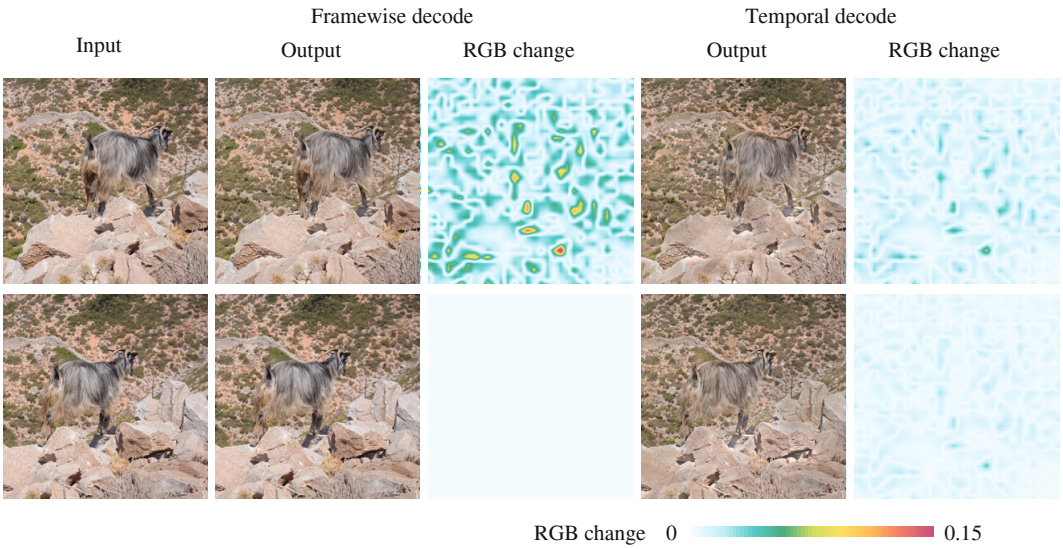

Figure 21: **A temporal decoder spreads an edit of one returned frame.** DAVIS *goat*, frames 44 and 53, one 480×480 crop shown at 256×256. Only the returned frame at 44 is perturbed (state RMS 0.035); stored boundaries are fixed. The perturbation is decoded frame by frame or with the five-tap temporal kernel $[0.1, 0.2, 0.4, 0.2, 0.1]$. RGB-change maps share one 0–0.15 scale.

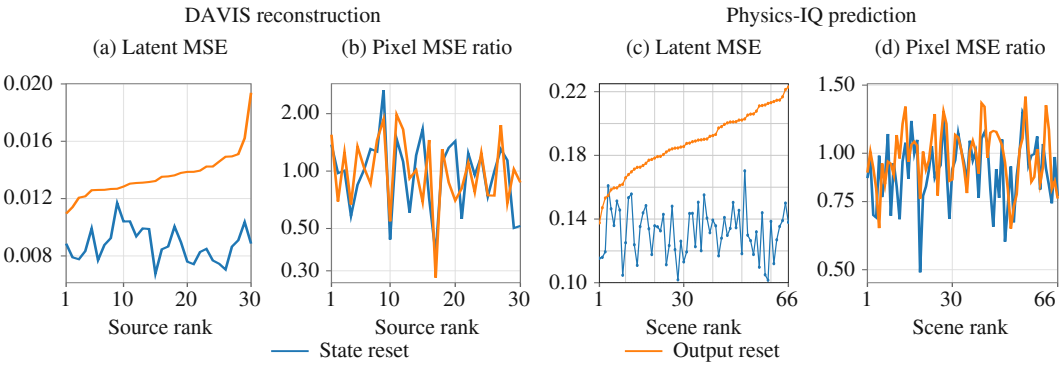

Figure 22: **State replacement lowers latent error most where error is largest.** CogVideoX, $R = 4$, width 512; 30 DAVIS sources (a, b) and 66 Physics-IQ scenes with three views (c, d), three training seeds. (a, c) Raw latent MSE before output replacement. (b, d) Pixel-MSE ratio to no replacement, computed from source- or scene-mean errors. Both panels of a dataset share the ascending order of latent error without replacement.

G.1 Action-Conditioned Prediction with Equal Token Counts

Protocol. IRASim's complete short tests (Zhu et al., 2025) contain 4,799 RT-1, 2,946 Bridge, and 5,243 Language Table windows. Each supplies one initial frame and 15 native actions, and the 15 future frames are scored. PSNR and SSIM follow the original grayscale path, latent MSE uses the scaled native VAE space, and FID (Heusel et al., 2017) and FVD (Unterthiner et al., 2018) use the full output sets; FVD uses I3D features of the 15-frame predictions with temporal mirroring disabled. Episode clustering keeps windows of one trajectory together.

Results. Native patching gives both representations $4S$ transformer tokens (Appendix A.3), so Equation (4) predicts no saving and an added executor cost. Table 17 confirms both parts: quality is nearly unchanged in all three domains, while KineTra takes 22.9–35.3% more time and 46–115% more memory. Language Table, which has the largest frames and the widest executor (1152), shows the largest overhead.

Table 17: **Short action-conditioned prediction across three domains.** Both representations retain IRASim Video-Ada and its native action conditioning (Zhu et al., 2025). Each test supplies an initial image and 15 actions. Quality uses the native protocols in Appendix G.1; costs are complete-video means. Bold is within each dataset and metric. With equal token counts, the executor adds cost.

Metric	RT-1		Bridge		Language Table	
	Dense	KineTra	Dense	KineTra	Dense	KineTra
PSNR (dB) ↑	25.58	**25.60**	**24.94**	24.89	**24.04**	23.98
SSIM ↑	**0.827**	**0.827**	**0.831**	0.830	**0.863**	**0.863**
Latent MSE ↓	**0.2120**	0.2143	**0.1976**	0.1982	**0.1983**	0.2000
FID ↓	4.88	**4.50**	**9.97**	10.43	**6.78**	7.29
FVD ↓	**30.51**	31.79	**23.99**	25.53	**74.73**	76.24
Seconds ↓	**6.173**	7.587	**6.191**	7.682	**10.818**	14.642
GiB ↓	**5.62**	8.48	**5.80**	8.47	**6.87**	14.78

G.2 Where the Time Goes

Table 18 splits the complete time of 961-frame LTX videos into phases. At 50 steps, the transformer takes 2936.8 of 3060.9 seconds for Dense (96%) and 262.5 of 394.6 seconds for KineTra (67%); the executor adds 8.1 seconds, and the VAE takes about 113 seconds in both. At four steps, KineTra spends 22.4 seconds in the transformer and 113.4 in the VAE, 73% of 155.4 seconds. Each reduction of transformer time therefore raises the share of decoding.

Table 18: **Complete-video latency breakdown.** Mean seconds for 961 frames, 512×768, 16 fps, on one A100-SXM 80GB. Other includes conditioning and transfer/saving. Lower is better. Rounded phase means can differ from the rounded total.

Config.	Body	Exec.	VAE	Other	Total
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev; 50 steps					
Dense / 50	2936.8	**0.0**	112.8	**11.4**	3060.9
KineTra / 50	**262.5**	8.1	**112.6**	**11.4**	**394.6**
LTX-Video (HaCohen et al., 2024) / 13B / 0.9.8-dev; 4 steps					
Dense / 4	238.2	**0.0**	113.8	**11.5**	363.5
KineTra / 4	**22.4**	8.1	**113.4**	**11.5**	**155.4**

G.3 Combination with Other Accelerations

Settings. Table 19 applies each acceleration independently to both 961-frame LTX models. DiCache (Bu et al., 2026) uses thresholds 0.1 and 0.2. SpargeAttention (Zhang et al., 2025b) uses the released sparse and quantized operator with top-k fraction 0.5 on compatible self-attention calls; cross-attention, feed-forward layers, and unsupported masked calls keep their original operators. Our four-step LTX adaptation of distribution-matching distillation (Yin et al., 2024b;a) initializes the student and the fake-score network from the corresponding LTX model and uses the teacher of the same representation, with 1,000 student updates, five fake-score updates per student update, batch size 16, and teacher guidance one.

External workpoints. For Table 1 and Table 2, D2Cache (Liu et al., 2026) is applied to LTX; SenCache (Haghighi & Alahi, 2026) to LTX and flow-adapted CogVideoX1.5; and Sparse-vDiT (Chen et al., 2025b), reimplemented from its paper, to CogVideoX1.5's spatial video grid. C/F denote conservative/faster presets. D2Cache enables second-order correction with accumulated relative-error thresholds 0.10/0.20, resets each sample, and computes full steps until two valid histories exist. SenCache uses $(n, \epsilon) = (2, 0.25)/(3, 0.50)$ on LTX and $(2, 0.30)/(3, 0.60)$ on CogVideoX, with $\epsilon = 0.01$ for the first 20% of steps and the original $m < n$ refresh rule. Sparse-vDiT uses $(\lambda, \epsilon) = (0.5, 0.5)/(0.5, 1.0)$, its three sparse patterns, head skipping, and offline search on the target A100 grid. These rows reuse the three Dense checkpoints, 50 Euler steps, guidance one, and paired prompts and initial noise, at inference batch size one.

Calibration and evaluation. Calibration uses the first eight length-eligible MiraData validation sources sorted by source ID, each source's first valid window and caption, and generation seed zero, independently for each length, representation, and training replica; error maps, sensitivities, and sparse configurations are recalibrated, with search ties resolved by lower cost then lexicographic configuration order. Evaluation follows Appendix D.1–D.2: VBench uses 2,838 videos per configuration and short VMBench 3,150; long Dynamic averages the same 31 windows for each of 72 prompts per replica. Complete-pipeline timing includes cache decisions, sparse operators, decoding, and communication; model loading, calibration, compilation, and warm-up precede timing. In Table 19, D2Cache and SenCache act on the generator transformer of each representation; the executor and

boundary recurrence retain their original computation. Its four D2Cache/SenCache Dense rows reuse the corresponding results in Table 2.

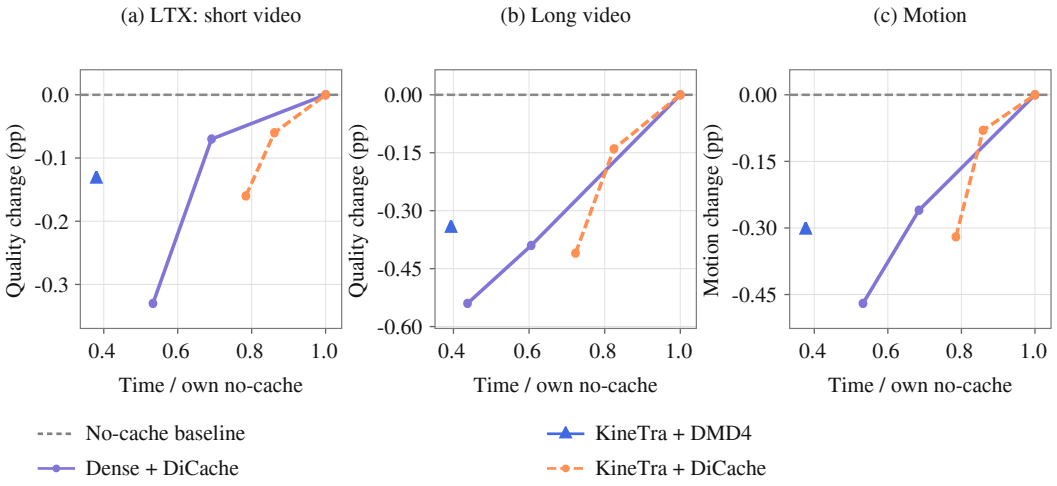

Figure 23: **Caching adds a smaller increment on top of KineTra.** LTX short-video Quality, long-video Quality, and VMBench motion, mean over three training seeds. Each curve is relative to its own uncached model: time ratio on the horizontal axis and score change in points on the vertical axis. DiCache thresholds are 0.1 and 0.2; the separate four-step marker is relative to 50-step KineTra.

Table 19: **Representation and acceleration are complementary.** LTX 0.9.8-dev at 961 frames, 512×768, 16 fps, one A100-SXM 80GB per video. Each intervention is independent. DiCache, D2Cache, and SenCache settings (C: conservative; F: faster) are in Appendix G.3; sparse attention follows Zhang et al. (2025b) and distillation adapts Yin et al. (2024a). Bold marks the better representation for each setting and metric.

Acceleration setting	LTX-Video 13B / Dense				LTX-Video 13B / KineTra			
	Quality ↑	Sem. ↑	Sec. ↓	GiB ↓	Quality ↑	Sem. ↑	Sec. ↓	GiB ↓
Default / 50 steps	81.38	**77.28**	3060.9	67.13	**81.92**	77.27	**394.6**	**43.52**
DiCache $\delta = 0.1$ (Bu et al., 2026)	80.99	**77.18**	1853.7	70.30	**81.78**	**77.18**	**325.3**	**44.40**
DiCache $\delta = 0.2$ (Bu et al., 2026)	80.84	76.87	1338.4	70.28	**81.51**	**77.01**	**285.2**	**44.49**
D2Cache C (Liu et al., 2026)	81.22	**77.19**	1678.2	71.32	**81.80**	**77.19**	**310.6**	**44.91**
D2Cache F (Liu et al., 2026)	80.91	76.87	1196.4	71.41	**81.48**	**77.00**	**269.4**	**45.04**
SenCache C (Haghighi & Alahi, 2026)	81.32	**77.24**	1746.8	68.27	**81.88**	77.22	**316.2**	**43.92**
SenCache F (Haghighi & Alahi, 2026)	80.99	76.99	1421.7	68.35	**81.61**	**77.07**	**283.8**	**44.00**
SpargeAttention (Zhang et al., 2025b) / 0.5	81.26	**77.16**	1560.7	62.81	**81.79**	**77.16**	**347.1**	**42.97**
Distillation (Yin et al., 2024a) / 4 steps	81.06	**76.96**	363.5	67.11	**81.58**	76.95	**155.4**	**43.60**
Frozen VAE compilation	81.37	**77.28**	3029.9	67.11	**81.92**	77.27	**363.5**	**43.41**

Results. KineTra is faster than Dense under every acceleration and keeps higher Quality in every row. Each transformer-side acceleration still helps KineTra, but by a smaller factor than it helps Dense, because it acts on the transformer phase and KineTra has already shortened that phase: caching speeds up Dense by 1.65–$2.56\times$ and KineTra by 1.21–$1.46\times$, sparse attention by $1.96\times$ and $1.14\times$. Under D2Cache or SenCache, KineTra is 4.44–$5.52\times$ faster than Dense with the same cache preset. D2Cache F reduces KineTra from 394.6 to 269.4 seconds while Quality changes from 81.92 to 81.48; the presets therefore retain a quality–time trade-off. Compiling the frozen VAE reduces KineTra's decoding from 112.6 to 79.4 seconds and its total from 394.6 to 363.5 seconds, while it changes Dense by 1%, since decoding is a small part of Dense's time.

Figure 23 compares the incremental effect of DiCache. The short and long VBench tasks each have 946 videos per seed and VMBench 1,050, all at 512×768 and 16 fps with three seeds; short and motion videos have 129 frames and long videos 961. On each task, DiCache removes a smaller fraction of KineTra's time than of Dense's (long-video time ratios 0.72–0.82 versus 0.44–0.61) and changes its scores by less in absolute terms. On KineTra, four-step distillation reaches a lower time ratio than caching, 0.39 on long videos, at a Quality change of -0.34.