StudentProbe: What Survives Distillation in Training-Free Video Diffusion Acceleration

Runguo Li
University of Illinois Urbana-Champaign
runguol2@illinois.edu

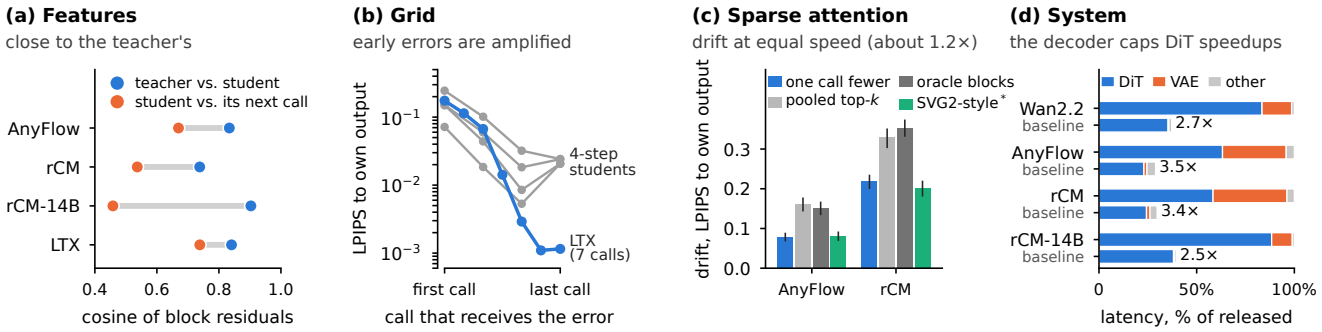

Figure 1. StudentProbe at a glance. (a) Teacher run on a student's states: block residuals are closer to the teacher's than to the student's next call (33 held-out prompts). (b) A Gaussian state error of 10% moves the output most at early calls. (c) Sparse attention over position blocks vs one call fewer; *density 0.23, ideal kernel. (d) Latency as released and with the baseline; the non-DiT part bounds any DiT accelerator. GPUs: H100 NVL, 4090/A100, 4090, PRO6000 BW.

Abstract

Video generation is increasingly served by distilled students—few-step and autoregressive models that produce a clip in a handful of network calls—yet the training-free accelerators deployed on them, from feature caches to sparse attention, were designed and calibrated on 30–50-step teachers. Whether these accelerators survive distillation has so far been assumed rather than measured. We introduce StudentProbe, to our knowledge the first protocol that measures each student against its own teacher under controlled conditions: at matched noise level, on the states the student actually visits and on one GPU type, with predictions registered before held-out tests on a new family, scale, prompt set and GPU. Recent work concludes that distillation leaves little redundancy or sparsity to exploit; StudentProbe shows the opposite at the level of the network: students keep their teachers' attention concentration and features. What changes is the sampling grid, whose few wide steps remove step-to-step redundancy and amplify early errors, and the system, where the decoder and the attention kernel now cap the gain. This single picture explains why cache signals calibrated on teachers mis-rank the steps a student can skip, and why block-sparse attention over position blocks, even with oracle selection or compensation, drifts further than running one step fewer at equal latency. StudentProbe replaces teacher-side calibration with a few minutes of measurement on the student that predicts the speedup and the lowest-drift skip positions; guided by it, released components set a reference point of 2.5–3.5× end to end on four students, at a measured cost in motion (all four) and aesthetics (three) on untouched prompts. We will release StudentProbe and its full registration record.

1. Introduction

Video diffusion models are increasingly deployed as distilled *students*: few-step bidirectional models that need 4–8 network evaluations instead of 50 [6, 47], and autoregressive (AR) models that stream a video chunk by chunk [8, 41]. Many of the training-free accelerators paired with them were designed, tuned and evaluated on 30–50-step teachers. These include step caches that reuse features across denoising steps [15, 18, 23], sparse attention [32, 44] and post-

Accelerator (designed on teachers)	On students	Evidence	Decided by
Step cache with a teacher-calibrated signal (FBCache, TeaCache)	mis-ranks skips	tuned on the student to skip one call, they never outperform the measured skip and are worse in drift or quality for 4 of 5 student–cache pairs; FBCache's signal orders 6 of 15 LTX pairs correctly (Sec. 5.1)	grid: amplification, sampler
One skipped call at a measured position	transfers	1.16–1.29×; 8 tuning prompts find the lowest-drift position (22/23 pairs); confirmed on held-out prompts and at 14B (Sec. 5.4)	measurement
Training-free block-sparse attention (pooled top-k over position blocks)	loses to one call fewer	drifts further than dropping a step at equal latency ($+0.08$ to $+0.18$ LPIPS; density 0.1–0.5, first call dense), even when an oracle selects the blocks with the most attention mass; lower aesthetics on three students but higher imaging quality on AnyFlow; $\leq 1.05\times$ on LTX; a semantic-aware port matches it in drift under an ideal kernel (Sec. 5.2)	selector, kernel shape, GPU
Dynamic INT8 W8A8 (torchao), compiled	varies	DiT $\leq 1.04\times$ at 1.3B, $1.13\times$ at 14B, $1.23\times$ at 13B, teacher and student alike (Sec. 4.3)	model size, not distillation
More steps on the first AR chunk	transfers	one extra call more than halves the drift of a reduced schedule (Sec. 5.3)	context anchoring
Any DiT accelerator	capped	the VAE decoder takes 26–35% of a 1.3B Wan student's latency, 9.5% at 14B, 39–47% on a B200 (Sec. 4.3)	latency split
Two calls + lightweight decoder + block-wise compilation	baseline	2.52–3.45× on four students; drift within 14% of two calls alone; on untouched prompts, smoothness and flickering improve; aesthetics (three of four) and motion (all four) fall (Sec. 6)	grid and system

Table 1. What StudentProbe finds transfers from teacher to student, and what decides it (end-to-end speedups on one A800; last row, the StudentProbe-guided baseline, on a PRO6000 BW).

training quantization [10]. Whether they retain their speed–quality trade-offs after distillation is mostly asserted rather than measured. Several works state that step caching degrades on few-step models [16, 28, 50], because they "have no inter-step redundancy left" [42], or that video attention is not reliably sparse [2], and both statements motivate *trained* replacements. Student and teacher differ in weights, grid and visited states, and reported speedups mix GPUs and bases, so such statements cannot say what distillation removed.

StudentProbe, the protocol we introduce, separates these factors on three DiT families, Wan2.1 [29], LTX-Video [7] and FLUX.2-klein, each with its teacher and its released students: two bidirectional, two AR, one image and one held-out video student. We compare student and teacher at matched noise level and on the latent states the student visits, run an any-step student on a fine grid with its weights fixed, and time end to end on one GPU type per comparison. After studying the Wan family we pre-registered predictions for a held-out family, a held-out scale (rCM on Wan2.1-14B with its teacher), held-out prompts and a second GPU, and we report every outcome. *Held out* marks results on such new data, and *registered* marks checks on models already studied.

At the level of the network these statements do not hold: at matched noise level the students' attention is as concentrated as their teachers', and on the same latent states their block residuals are closer to the teacher's than to their own at the next call (Fig. 1a). On AnyFlow the low similarity between consecutive calls follows from the grid, since the same weights on a 50-step grid are as similar between calls as the teacher. The few wide steps amplify errors made early (Fig. 1b), and with 4–7 DiT calls the VAE decoder and the attention kernel's speed at each model's shape bound the end-to-end gain (Fig. 1d). Cache signals calibrated on teachers therefore mis-rank skip positions, and block-sparse attention over position blocks drifts further than dropping a step at equal latency, even with oracle block selection or compensation for the unselected keys (Fig. 1c, Sec. 5.2). Table 1 lists what transfers and what decides it.

Our contributions are:

- **Protocol.** StudentProbe, to our knowledge the first controlled teacher–student comparison of training-free accelerators, separating weights, sampling grid, visited states and system cost on six distilled students.
- **Findings.** Students keep their teachers' attention concentration and block features, yet teacher-calibrated cache signals and position-block sparsity do not transfer (Tab. 1).

- **Diagnosis.** The few wide steps amplify early errors, and the decoder and the kernel's speed cap the gain.
- **Probe.** A latency-split probe and an 8-prompt skip sweep predict the speedups and find the lowest-drift skip position on every student within minutes, confirmed on a held-out family, scale, prompt set and GPU.
- **Reference point.** A baseline assembled from released components by these measurements reaches 2.5–3.5× end to end on four students, with its cost in motion and aesthetics measured on untouched prompts.

2. Related Work

Few-step and autoregressive students. Video generators are now deployed as distilled students. Distillation by distribution matching [39, 40], by continuous-time consistency, alone or with a DMD regulariser [20, 47], and by on-policy flow maps [6] brings Wan2.1 [29] to 1–4 network evaluations. Autoregressive (AR) students [8, 36, 41] generate chunk by chunk from a clean-context key-value (KV) cache and target streaming latency. Vendors also release official step-distilled checkpoints such as LTX-Video [7].

Training-free acceleration, designed on many-step teachers. Three families dominate, each tuned on 30–50-step samplers: step caching and forecasting [3, 15, 17, 18, 23, 26, 48], sparse attention [11, 14, 32, 37, 44, 46] (trainable [43] or compensating for unselected keys [19]), and post-training quantization [4, 10]. Step caches decide when to skip from a low-cost signal calibrated on the teacher's trajectory. TeaCache [15] uses the change of a timestep-modulated input, First-Block Cache (FBCache) [1] the change of the first block's residual, and MagCache [23] residual magnitude ratios. A calibrated schedule is tied to the sampler, step count, resolution and precision it was calibrated with [31], and the best recipe differs across models and hardware [12].

Acceleration of few-step and AR students. Several recent works assert that step-wise reuse stops working after distillation [16, 28, 38, 42, 50]. For attention, the real-time AR method MonarchRT finds that video attention "is not reliably sparse" [2]. The responses are mostly trained: learnable caches [50], block-sparse attention distilled with steps [5], and a trained stack of step distillation, sparse-linear attention and W8A8 [45]. For AR students, chunk-aware sparsity allocation [21] and trainable persistent-block sparsity [33] report that attention concentrates on a persistent subset of the context, and chunk-wise caches [22] and, concurrently, cache anchors [34] treat chunks differently. Training-free work on students includes frame-interleaved sparsity [28], solver-aware cache targets [38], model routing [25], progressive resolution [30], head-specific KV

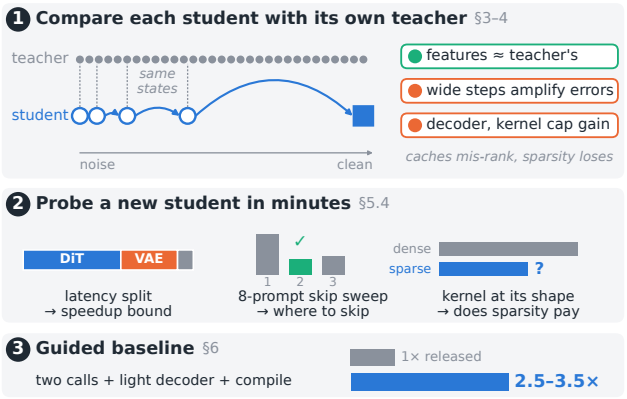

Figure 2. StudentProbe (schematic): (1) a controlled teacher–student comparison on the same states, (2) a minutes-long probe of a new student, and (3) the baseline they guide. Pairs and sampling settings are in the supplement.

caches [27], fewer steps for later AR chunks [24] and, concurrently, compensated sparse attention [13]. These works evaluate a method on students. StudentProbe instead compares each student with its own teacher to find out why a teacher-designed accelerator transfers or fails, and we test frame-interleaved sparsity, a semantic-aware selector [37] and the AR schedule of [24] against the grid.

Measuring acceleration. Speedups often mix bases (FLOPs vs latency, DiT vs end to end, GPUs), caches are scored by fidelity to uncached outputs [17], and once DiT time is small the VAE decoder is the bottleneck [49].

3. The StudentProbe Protocol

Teacher–student pairs. We study three families, each with a teacher and the students released for it (Fig. 2; settings in the supplement). Wan2.1-1.3B [29] comes with two bidirectional students, AnyFlow [6] and rCM [47] (an unofficial public reproduction, at both scales), and two AR students, AnyFlow-FAR (frame-autoregressive) and Self-Forcing [8]. LTX-Video 0.9.7 (13B) [7] has official *dev* and *distilled* checkpoints, and FLUX.2-klein (4B) gives an image pair, to test whether the findings are specific to video. Every model runs with its official sampling settings (supplement). Skip and quality runs use AnyFlow, rCM, klein and LTX; the AR students enter the probes, AR schedules and latency split. LTX-Video is the held-out family: its predictions were registered before any LTX student run (Sec. 7). rCM on Wan2.1-14B is a held-out scale registered the same way.

Prompts and seeds. Development uses 33 VBench [9] prompts stratified over its dimensions (seed 0). The held-out set is 33 disjoint VBench prompts (seeds 0 and 1), and

images use 64 DrawBench prompts. Eight separate tuning prompts per modality serve only probes and calibration.

Accelerators. All accelerators are training-free and plug into the released pipeline. On the step axis we use residual reuse of the whole block stack at a chosen call and first-order Taylor forecasting [18]. On teachers we also use FBCache, TeaCache [15] and MagCache [23] with their published calibrations. On the spatial axis we use block-sparse self-attention with pooled-score selection [35, 44]. It keeps a fraction d of 128×128 blocks, runs on a FlexAttention kernel, and keeps the first w steps dense. System-level options are W8A8 dynamic INT8 (torchao), `torch.compile` of every block, and VAE precision.

Speed. Timings use 1–2 warm-up and 3–5 timed runs, and a run is discarded if another process appears on the GPU. Each comparison stays on one GPU type (an A800 unless noted; the supplement lists the GPU behind every table and figure). We report *latency*, the end-to-end time of one generation, split into text encoder, DiT, VAE and other, and peak memory. A latency-split probe adds CUDA-event timing of self- and cross-attention, and the sparse kernel's speedup at each model's attention shape, to predict an accelerator's effect before it is run (Sec. 4.3).

Quality. The *drift* of an accelerated run (for a skip, its fidelity cost) is its LPIPS to the unaccelerated output of the same model, prompt and seed. A *quality change* is the change of a named reference-free score: CLIP text alignment, LAION aesthetics, DINOv2 subject consistency, RAFT motion and VBench's imaging quality, motion smoothness and temporal flickering. Differences are paired by prompt and seed, with 95% bootstrap confidence intervals that resample prompts, the two seeds of a prompt together (66 videos are 33 prompts with two seeds). A score shows *no significant decline* when its CI does not lie below 0, and is *non-inferior* when the CI's lower end lies above a negative margin fixed in advance (Sec. 6).

Compile reference. Compilation can change numerical results relative to eager execution, and the trajectory amplifies the difference. We report the LPIPS between compiled and eager outputs, the *compile reference*: AnyFlow 0.027, LTX dev 0.025, LTX distilled 0.075, Wan 0.099, rCM 0.145. Rankings always use the raw values.

Pre-registration. Predictions, thresholds and decision rules were recorded before the runs they concern, and the supplement lists every registration and the data it had seen. Section 7 reports every outcome.

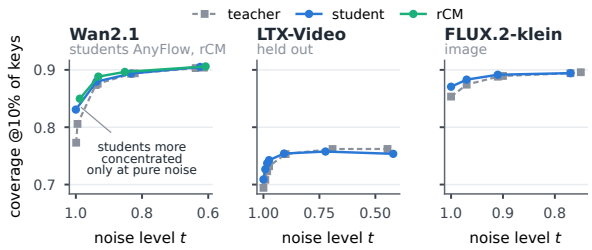

Figure 3. Oracle attention coverage at 10% of the keys vs noise level. Teacher and students coincide at matched noise level in all three families, including the held-out pair (pre-registered).

Interval (t)	Wan teacher		AnyFlow		rCM	FAR, chunks 1–7	14B	
	one step	same interval	4 steps	50 steps			teacher	rCM
$1000 \to 940$	0.99	0.51	0.48	0.96	0.42	0.76–0.90	0.59	0.31
$940 \to 833$	0.99	0.81	0.75	0.99	0.57	0.88–0.93	0.75	0.46
$833 \to 620$	0.995	0.86	0.80	0.99	0.63	0.88–0.92	0.70	0.55

Table 2. Block-residual cosine between consecutive calls. Over the same noise interval, the teacher's features change about as much as a student's do in one step, and AnyFlow on a 50-step grid (same weights) is as redundant as its teacher. 14B: teacher over the same interval (one step: 0.98). AnyFlow at 50 steps: its calls grouped by t into these intervals (its grid differs from the teacher's).

4. What Distillation Changes

StudentProbe asks three questions about a student and its teacher: do they attend and compute differently (Sec. 4.1), how does the few-step grid propagate errors (Sec. 4.2), and where does the time go (Sec. 4.3)?

4.1. Features at matched noise level

To compare attention, we sample 256 queries in every fourth layer and compute their exact softmax. The *oracle coverage* at density p, the attention mass held by the top p fraction of keys, bounds what any selector can keep at that density. At matched noise level, teacher and student coincide in all three families, within 0.018 including the held-out scale and family (Fig. 3); only at pure noise are the students more concentrated. Because coverage is a fraction of the keys, AR chunks with a growing context are compared at equal key count (supplement). Distillation thus leaves intact the concentration that sparse attention exploits. At the oracle level, student attention stays as sparse as the teacher's, in line with the persistent concentration that AR sparse methods build on [21, 33] and against MonarchRT's premise that video attention "is not reliably sparse" [2].

Step caches rely on features changing little between calls, so we next ask whether students lost this redundancy. We take the cosine between a block's residual (output minus input) at two consecutive calls, averaged over blocks and 4 tuning prompts, and measure the teacher (conditional branch) both between its consecutive steps and over the noise interval of one student step. Step to step the teacher looks highly

redundant (0.99, Tab. 2), but over a student-sized interval its features change about as much as AnyFlow's do in one step. On AnyFlow, redundancy therefore follows the step size rather than the weights: run on a 50-step grid with the same weights, AnyFlow is as redundant as its teacher (0.96–0.99; registered). On rCM and klein the teacher's summed change over the student's intervals, $\sum_k (1 - \cos_k^{\text{teacher}})$, is only 55–59% of the student's (AnyFlow: 84%), so on separate trajectories part of their gap remains. Step placement matters as well: LTX puts its smallest steps where features change fastest, where consecutive calls reach 0.69–0.86.

Separate trajectories mix the network with the states it visits. To isolate the network, we run the teacher on the states a student visits while it samples (AnyFlow, rCM at both scales and LTX; 8 tuning prompts and 33 held-out prompts). Over the same two states the teacher's residuals change at least as much as the student's (ratio 102–106%), so on these pairs the remaining gap goes with the states visited rather than with the weights. For rCM, these runs also include its sampler's re-noising. On a single state, a student's residuals are even closer to its teacher's than to its own one call later (paired difference $+0.10$ to $+0.45$, every CI above 0; Fig. 1a). Distillation thus changes the path through latent space more than what the network computes along it.

4.2. The grid amplifies early errors

Late intervals of a few-step grid are the widest, yet on the LTX student late skips cost little fidelity. A registered diagnostic tested three explanations. The decoder is not the reason: with a plain decode the last skip still costs only 0.045 of the first. Nor is the local error: the error that reuse injects at call k, $|\Delta\sigma_k| \, \|v_k - v_{k-1}\| / \|x_k\|$, grows 29× from call 1 to call 6. What decides is how the rest of the trajectory propagates an error: a Gaussian state error of 10% of the state's norm moves the output by LPIPS 0.114 at call 1 but by 0.003 or less from call 4 on (Fig. 1b). The remaining wide steps amplify early errors and damp late ones, and on LTX this orders every skip position. The order holds at every error size from 3% to 30% of the state's norm, and the actual skip costs (Fig. 4) follow it. A registered repeat on AnyFlow, rCM, rCM-14B and Wan2.2 finds the same direction with a smaller first-to-last ratio (3.5–10×): there amplification makes the first call the costliest but does not order the rest. For accelerators, drift therefore measures how far the trajectory moved rather than whether the video got worse, and block similarity, which ignores this amplification, cannot rank skip positions.

4.3. The system caps the gain

With far fewer DiT calls, components that were negligible for the teacher bound what an accelerator can deliver (Fig. 1d; per-model split in the supplement). The VAE decoder takes 3.0% of end-to-end time in the 50-step Wan teacher but 26–35% in its 1.3B students, so even an infinitely fast DiT would give them at most 2.6–3.4×. The share falls with model size (9.5% for rCM at 14B) and latent compression (6.8% on LTX), and rises on faster GPUs: on a B200, rCM's DiT runs 5.6× faster than on the A800 but its decoder only 3.1×, so the decoder takes 39–47% of a 1.3B student's latency. Kernel gains depend on the attention shape and the GPU, and are the same for teacher and student. Block-sparse attention is 3.1× faster than dense attention at the Wan shapes on the A800, but no faster on a B200, and no faster, or slower, for the short attention of LTX, klein and Self-Forcing's chunks; on LTX and klein, self-attention takes only 13–16% of the time anyway. Compiled W8A8 INT8 gains at most 1.04× at 1.3B and up to 1.24× at 13B.

	AnyFlow			rCM			LTX (held out)		
Method	call	LPIPS	Δaes.	call	LPIPS	Δaes.	call	LPIPS	Δaes.
StudentProbe	2	.074	−.015	2	.284	−.054	6	.009	+.**055**
FBCache	2†	.069	−.**058**	3	.300	−.**555**	3	.117	+.029
TeaCache	1	.206	−.060	2	.284	−.054	–	–	–

Table 3. StudentProbe's measured skip against FBCache and TeaCache tuned on each student to skip one call per video (66 held-out videos, A800): the call skipped, LPIPS to the unaccelerated output and the change in aesthetic score (bold: 95% CI excludes 0). †In 57 of 66 videos. TeaCache uses its Wan2.1 coefficients. FBCache also skips the costliest call on rCM-14B and HunyuanVideo-1.5 (supplement).

5. Calibrating on the Student

Section 4 predicts which teacher-designed accelerators should transfer; we now test step skipping, sparse attention and AR schedules on the students and turn the measurements into StudentProbe's calibration (Sec. 5.4).

5.1. Where a student can skip

We measure two costs of skipping call k, i.e. reusing the previous call's block-stack residual there. The *fidelity cost* is the LPIPS to the unaccelerated output of the same prompt and seed; caches are designed to minimize it. The *quality effect* is the paired change of reference-free scores, mainly aesthetics and DINO subject consistency. We relate the fidelity cost to four low-cost signals (Fig. 4; every value in the supplement): S1, $1-$ the block-residual cosine; S2, the first block's relative residual change, i.e. FBCache's signal; S3, $1-$ the output's cosine to the previous call's; and S4, the output change caused by Gaussian noise of norm $0.1\|v_k\|$ added at call k. A signal *ranks* the positions if it orders at least 8 of the 9 position pairs of the three 4-step students and 13 of the 15 LTX pairs as the fidelity cost does (registered thresholds).

No signal ranks every student (supplement). FBCache's signal, meant to be small where reuse is safe, is largest at

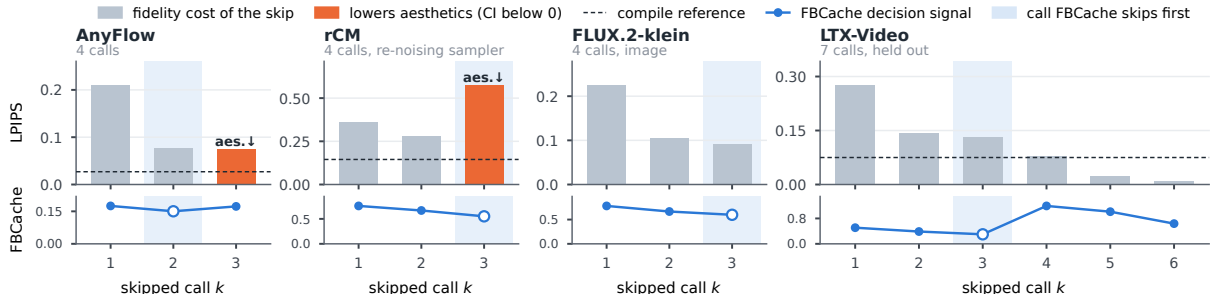

Figure 4. Top: fidelity cost of skipping one network call (LPIPS to the student's unaccelerated output); orange bars significantly lower the aesthetic score; dashed: compile reference (Sec. 3). Bottom: FBCache's decision signal at that call; shaded: the call its threshold skips first. Both axes start at zero. The cache signal tracks the fidelity cost on klein but inverts it on the held-out LTX student and on rCM.

LTX calls 4–5, where skipping costs least (Fig. 4), and TeaCache's depends only on the timestep, so on a fixed sampler it is a fixed schedule. S3 and S4 rank LTX, but no signal ranks the three 4-step students; only S4, which adds noise at call k, orders rCM, whose costliest skip comes from its sampler's re-noising (below). Tuned on the student to skip one call per video, FBCache and TeaCache therefore never outperform the measured skip on held-out prompts (Tab. 3): FBCache skips LTX call 3 (13× the cost of call 6) and rCM's last call, and TeaCache skips AnyFlow's first. On LTX and AnyFlow these picks are early calls, whose errors the remaining steps amplify most (Sec. 4.2), and neither a signal of local change nor the timestep sees this amplification.

Low drift and preserved quality are different targets, and the two costs of a skip disagree in both directions. On LTX, early skips yield a different video (high drift) without a measurable quality change (Fig. 5), and no skip lowers aesthetics or DINO significantly. On AnyFlow, the last skip has the lowest drift but is the only one that lowers aesthetics (-0.10; -0.15 held out), while call 2 drifts about as little without a significant decline; these verdicts replicate on a B200 and in VBench. Choosing by drift alone would thus pick the wrong call on AnyFlow and overstate the cost of early skips on LTX. rCM adds a third case, caused by its sampler, which draws fresh noise before every call: a velocity reused at the last call carries the previous call's noise into the output, scaled by the last noise level $t \approx 0.61$ (supplement). Its last skip therefore severely degrades the output at both scales (aesthetic -2.00), whereas stopping before that call, which removes the reused velocity and nothing else, does not lower aesthetics. On rCM, saving the last call means stopping early rather than reusing.

5.2. Sparse attention

Students keep their teachers' attention concentration (Sec. 4.1), so sparse attention should transfer; the fair comparison is the student's own alternative, one call fewer, at equal latency. As in base-model sparse methods, the first call stays dense, where sparsity costs most (AnyFlow: LPIPS 0.430, against 0.023 at the last call), consistent with the amplification of Sec. 4.2. At density 0.2, block-sparse attention with pooled top-k selection matches one call fewer in measured speed (1.18–1.22× vs 1.18×) but drifts 1.6–2× as far on AnyFlow and rCM on held-out prompts, and further at 14B; no density from 0.1 to 0.5 is both as fast and as close (supplement). Nor would a faster kernel close it: under an ideal kernel, every sparse setting as close as one call fewer saves at most 0.82 of that call's time. On LTX (short attention), sparsity gives only 1.05×, while reusing the last call gives 1.16× at a quarter of the drift.

Lost attention mass does not explain this gap. At density 0.2, an oracle that keeps the blocks holding the most mass retains 0.85 of it (pooled scoring: 0.78), yet on AnyFlow and rCM it still drifts further than one call fewer, and an oracle over single tokens (0.93) drifts the most. Both are selection effects: the same code with every key kept stays within 0.02 LPIPS of the base (registered). Keeping more mass therefore does not guarantee a closer output. Our reading is that with 4 calls each call carries a large share of the denoising, so a schedule the student was trained to support perturbs the output less than an approximation inside every later call. In quality at equal latency, pooled selection also scores lower aesthetics (-0.07 to -0.10) and flickering than one call fewer on all three students, though higher imaging quality on AnyFlow. A semantic-aware selector, our port of SVG2's permutation [37], matches one call fewer in drift at density 0.23 (not at 0.19) under an ideal kernel and free clustering, with lower DINO consistency, so semantic grouping is a promising direction if a real kernel keeps its speed. Compensating for the unselected keys does not close the gap: our port of Rectified SpaAttn [19] drifts as far as pooled selection with 12% more attention, further at its budget, and on rCM further than one call fewer at equal latency.

5.3. Autoregressive students

AR students raise a different question: every later chunk attends to the first through the KV cache, so first-chunk errors should propagate and later chunks should tolerate fewer

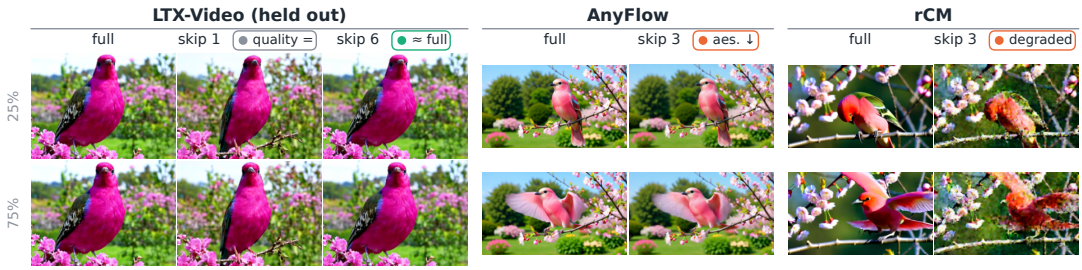

Figure 5. Frames at 25% and 75% of the clip, for the prompt whose LTX skip-1 fidelity cost is closest to the median (not selected by appearance). LTX skip 1 has a high fidelity cost (a slightly different video) but no significant quality change. rCM skip 3 severely degrades the output, through its sampler's re-noising (Sec. 5.1).

steps. On Self-Forcing (7 chunks, 4 calls per chunk), keeping 4 steps in the first chunk and 3 in the rest (22 calls) more than halves the drift of uniform 3 steps (0.131 vs 0.312 LPIPS), and even 4 and 2 (16 calls) stays at 0.155; both replicate on held-out prompts. LPIPS to the 4-step output favors schedules whose first chunk matches the reference, so we also compared at equal budget on 33 new prompts (registered): spending the two extra steps on the first chunk rather than the last raises aesthetics by 0.19 $[0.10, 0.28]$, and with 16 instead of 28 calls the anchored schedule shows no significant aesthetic decline against the 4-step output (-0.014 $[-0.057, +0.030]$); DINO does not separate the two. The anchoring in the schedules of [24] thus improves the aesthetic score of the whole video but not subject consistency.

5.4. StudentProbe on a new student

Since no low-cost signal ranks the positions on every student, we measure the cost of every skip position directly, which takes few runs with only 4–7 calls. On 8 tuning prompts, disjoint from the evaluation prompts, a registered check orders 22 of 23 position pairs as the 33-prompt development set does and finds the same lowest-drift position on all four students, for 0.6–6.9 GPU-minutes per student. Because drift and quality can disagree (Sec. 5.1) and aesthetic changes of 0.1 are within the sweep's noise, we choose positions by drift on a few prompts, confirm quality on 30 or more, and move to the next-lowest drift if quality drops; on AnyFlow this replaces call 3 by call 2. With this check, calibration takes under half an hour of generation per student on one A800, and a per-video choice of position would gain only 0–8% in drift (supplement). We apply the protocol unchanged at the held-out 14B scale, where the sweep again picks call 2, and to two third-party LightX2V students, where the quality check corrects the pick on Wan2.2 A14B and the sweep ties on HunyuanVideo-1.5.

For speed, the latency-split probe applies Amdahl's law, $1/((1-f)+f/s)$, with share f and speedup s measured on the student beforehand. It predicted end-to-end speedups within 5% on the held-out LTX pair (registered bound: 15%), at 14B, and on an H100 and a B200, and overstated only the sparse slowdown at LTX's short attention (supplement). Because the parts barely interact, timing each once predicts a combination (Sec. 6).

Student	Speedup	Pred.	LPIPS	Δaes.	ΔDINO	Δmotion	GB
Wan2.2 A14B	2.69×	2.67×	.414	−.04	−.002	**−16**%	+9
AnyFlow	3.45×	3.46×	.256	**−.10**	**−.007**	**−22**%	+13
rCM	3.36×	3.33×	.288	+.02	.000	**−9**%	+13
rCM-14B (held out)	2.52×	2.50×	.260	−.06	**−.004**	**−8**%	+13

Table 4. Two calls + lightweight decoder + block-wise compilation on a PRO6000 BW, 66 held-out videos, against each student's base. Pred.: from separately timed parts. Bold: 95% CI excludes 0. Motion relative to the base's level. No VBench score has a CI below 0.

6. A Baseline Guided by StudentProbe

Sections 4 and 5 point to two places where a student's time can be saved: fewer calls on a grid the student supports, and the decoder, whose share grows as DiT time falls. We combine the two with block-wise compilation and choose each part with StudentProbe's measurements on the student.

For the grid, AnyFlow and the rCM students use their own two-step setting, and on Wan2.2 the 8-prompt sweep picks 1000+500 over the other two-call grids, which lower aesthetics. For the decoder, the latency split predicts where a lightweight replacement (`lightx2v/Autoencoders`, as released) pays off: on an H100 it gives 1.50–1.63× at 1.3B, where the decoder takes a third of the time, and 1.11–1.18× at 14B, where the share is smaller, at LPIPS below 0.04 and 13 GB more peak memory (supplement). Together with compilation, these parts give 2.5–3.5× end to end (Tab. 4). The measured speeds fall within 1.0% of the prediction from separately timed parts, and drift stays within 14% of the two-call grid's, so the decoder and compilation add little drift beyond the grid.

On the held-out prompts no VBench score declines significantly, while motion falls on all four students and aesthetics on AnyFlow. To test the combination on prompts that played no part in choosing it, we also ran it on a previ-

ously unused set of 33 prompts, against non-inferiority margins fixed in advance (registered: 0.2 SD of each released model's per-video score; for motion, 10% of its mean). There, smoothness and flickering improve on all four students, DINO is non-inferior on three and imaging quality on two, and aesthetics are non-inferior on rCM. Aesthetics fall by 0.08–0.18 on the other three, and motion by 7–18% on all four (CIs in the supplement). Built from released parts and short measurements, the combination gives future accelerators for students a reference point at 2.5–3.5×.

At equal latency, the two-call grid also drifts no more than two finer-grained alternatives (supplement). Our reimplementation of FIS-DiT [28] on Wan2.2 A14B shows no significant decline in aesthetics, DINO or VBench's scores, as its authors report, but with the decoder and compilation on both, at 2.63× vs 2.69×, it drifts further than the grid and lowers motion more; a planner that allocates frame sparsity and reuse across calls drifts at least as much as the grid on every student.

7. Pre-registered Predictions and Outcomes

After studying Wan and klein, we committed predictions, thresholds and decision rules before any held-out run and later diagnostic, and Tab. 5 lists every outcome. Later diagnostics were each registered before their own runs, and the H rows, which met new data (held-out family, scale, prompts, GPU and third-party students), are the confirmatory part.

The central measurements held on new data: coverage at matched noise level on the held-out family and scale (P4, D10), the same-state comparison on held-out prompts (D23), the headline differences (P6), the skip, cache and sparse results at 14B (D9) and the latency-split probe (P5, D11). Twenty-eight lettered criteria failed; Sections 4–6 report the measured result in each case, and the supplement groups them.

	Prediction (LTX unless noted)	Outcome
H	P1 Redundancy set by interval; small steps redundant	first held, second **failed**
H	P3 Early skips cost little, late ones much	untested; opposite order
H	P4 Coverage, student vs teacher: $\lvert\Delta\rvert \leq 0.03$	**held** (≤ 0.018)
H	P5 Amdahl 15%; sparse $<1.1\times$; INT8 $<1.15\times$	**held**; **held**; **failed**
H	P6 Headline differences replicate held out	**held** (4/4)
H	D5 AnyFlow: skip 3 lowers aesthetics, skip 2 not	**held**
H	D6 LTX: ranking reproduces, no skip lowers quality	**held** (15/15)
H	D7 Tuned caches vs protocol (LTX/rCM/AnyFlow)	**held/failed/failed**
H	D9 rCM-14B: system, skips, caches, sparse	**held** (9/9)
H	D10 14B pair: coverage; interval ratio	**held** (2/2)
H	D11 B200: probe; VAE; kernel; INT8	1 **failed**, 3 **held**
H	D12 Viewer study (registered)	not yet run
H	D15 B200 replicates quality; VBench agrees	**held** (4/4)
H	D16 Third-party Wan2.2, HunyuanVideo-1.5	**held** 8/12
H	D23 Same states on 33 held-out prompts	**held** 2/2
H	D25 Baseline on untouched prompts: non-inferiority	**held** 2/4
H	D26 AR anchoring at equal budget, new prompts	**held** 1/3
S	D3 A low-cost signal ranks skips on every student	**failed**
S	D4 8 prompts reproduce the ranking	**held** (22/23)
S	D13 Coverage at equal key count	**failed** (2/2)
S	D14 Densities 0.1–0.5 against one call fewer	**held** (4/4)
S	D17 FIS vs two calls; decoder; stacks	**held** 6/8
S	D18 Allocating compute across calls	**held** 4/6
S	D19 Two calls + decoder + compile (14B: H)	**held** 4/5
S	D20 Oracle and semantic-aware selectors	**held** 1/4
S	D21 Same states; error gain; coverage seeds	**held** 5/5
S	D24 AnyFlow on 50 and 16 steps: redundancy	**held** 2/2
S	D27 Parallel full-VAE decoding is faster	stopped (slower)
S	D28 Sensitivity-weighted caches rank skips	**failed**
S	D29 LightVAE better than the tiny decoder	**held** 2/4 (speed)
S	D30 Compensated sparsity vs pooled; vs grid	**failed**; **held** (rCM)
S	D31 Same attention code, every key kept	**held** 3/3
E	D2 Late skips cost little: decoder / local error / gain	no / no / **supported**
E	D8 Why D7 failed on rCM: plain reuse drops more	**held**
E	D22 rCM's last skip: stopping vs reuse	**held** 3/3

Table 5. Pre-registered predictions and outcomes. Grouped by kind; P3 was conditional on P1 and was not tested once P1 failed. H: new data (family, scale, prompts, GPU or third-party students). S: registered on models already studied. E: registered diagnostic after a surprise. The full record is in the supplement.

8. Limitations and Conclusion

Limitations. Timings span four GPU types (supplement), and kernel and INT8 gains are specific to the GPU. Sparse attention is timed with one kernel and one block size, and INT8 with one configuration (dynamic W8A8, torchao); window-based [11, 46] and AR persistent-block sparsity [33] lie outside this study. FIS and the SVG2-style selector are our ports, the latter without its kernel. rCM's checkpoints are an unofficial reproduction run with the official sampler, and its last-skip result follows from that sampler. AR students are tested on 5-s videos, and each student at one resolution (480p; LTX 512×704). Quality rests on 33–66 prompts per model, reference-free scores and frames, and CIs are not adjusted for multiplicity. Registrations are versioned records included in the supplementary material, and later held-out tests reuse the first held-out prompt set. The study covers three DiT families, two more in part, and training-free accelerators.

Conclusion. Distilled video students keep their teachers' attention concentration and block features. What changes is the sampling grid, whose few wide steps remove step-to-step redundancy and amplify early errors, and the system, where the decoder and the kernel's speed bound the gain. A skip measured on the student therefore transfers, while teacher-calibrated cache signals and position-block sparsity do not. For a new student, StudentProbe's three measurements decide what to accelerate: its latency split, which bounds any DiT accelerator; an 8-prompt skip sweep with a 30-prompt quality check; and the attention kernel's speed at its shape. An accelerator for a student is best compared with one call fewer, not with the teacher's redundancy. In practice this gives three rules: measure skip positions on the student instead of reusing teacher thresholds, judge any accelerator against one call fewer at equal latency, and replace the decoder first once it takes a quarter or more of the latency,

where a released lightweight decoder alone gives 1.5–1.6× at 1.3B. Next steps are a viewer study and a semantic-aware selector timed with its own kernel.

LLM Usage Statement

This manuscript was drafted by the authors. We used GPT-6 for (i) language editing of the manuscript text and (ii) assistance in implementing and debugging the measurement scripts. It was not used to generate figures, experimental results, related-work content, or research ideas. The extent of use was limited to sentence-level phrasing and code refactoring; all scientific content, system design, experiments, and analysis were developed by the authors, who verified all modified code and text.

References

[1] ParaAttention. Public code repository, 2025. 3

[2] Krish Agarwal, Zhuoming Chen, Cheng Luo, Yongqi Chen, Haizhong Zheng, Xun Huang, Atri Rudra, and Beidi Chen. MonarchRT: Efficient Attention for Real-Time Video Generation. *arXiv preprint arXiv:2602.12271*, 2026. 2, 3, 4

[3] Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. DiCache: Let Diffusion Model Determine Its Own Cache. *arXiv preprint arXiv:2508.17356*, 2025. 3

[4] Weilun Feng, Chuanguang Yang, Haotong Qin, Xiangqi Li, Yu Wang, Zhulin An, Libo Huang, Boyu Diao, Zixiang Zhao, Yongjun Xu, and Michele Magno. Q-VDiT: Towards Accurate Quantization and Distillation of Video-Generation Diffusion Transformers. *arXiv preprint arXiv:2505.22167*, 2025. 3

[5] Youping Gu, Xiaolong Li, Yuhao Hu, Minqi Chen, and Bohan Zhuang. BLADE: Block-Sparse Attention Meets Step Distillation for Efficient Video Generation. *arXiv preprint arXiv:2508.10774*, 2025. 3

[6] Yuchao Gu, Guian Fang, Yuxin Jiang, Weijia Mao, Song Han, Han Cai, and Mike Zheng Shou. AnyFlow: Any-Step Video Diffusion Model with On-Policy Flow Map Distillation. *arXiv preprint arXiv:2605.13724*, 2026. 1, 3

[7] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, et al. LTX-Video: Realtime Video Latent Diffusion. *arXiv preprint arXiv:2501.00103*, 2024. 2, 3

[8] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self Forcing: Bridging the Train-Test Gap in Autoregressive Video Diffusion. *arXiv preprint arXiv:2506.08009*, 2025. 1, 3

[9] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, et al. VBench: Comprehensive Benchmark Suite for Video Generative Models. *arXiv preprint arXiv:2311.17982*, 2023. 3

[10] Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. SVDQuant: Absorbing Outliers by Low-Rank Components for 4-Bit Diffusion Models. *arXiv preprint arXiv:2411.05007*, 2024. 2, 3

[11] 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: $O(n \log n)$ Sparse Attention with Energy Decay for Long Video Generation. *arXiv preprint arXiv:2506.19852*, 2025. 3, 8

[12] Yitong Li, Junsong Chen, Haopeng Li, Haozhe Liu, Jincheng Yu, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Sol Video Inference Engine: Agent-Native Full-Stack Acceleration Framework for Efficient Video Generation. *arXiv preprint arXiv:2606.23743*, 2026. 3

[13] Yanjin Li, Bingbing Qiu, Qiang Huang, Rui Qian, Yanfeng Li, Jian Wang, Xiaoying Tang, and Mingda Zhang. DecoupAttn: Decoupled Sparse Attention for Training-Free Acceleration of Video Diffusion. *arXiv preprint arXiv:2611.05917*, 2026. 3

[14] Enhuai Liu, Yunke Wang, Yutong Wang, Changming Sun, and Chang Xu. LoSA: Near-Lossless Sparse Attention for Training-Free Video Diffusion Acceleration. *arXiv preprint arXiv:2608.12032*, 2026. 3

[15] 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. *arXiv preprint arXiv:2411.19108*, 2024. 1, 3, 4

[16] Hao Liu, Ye Huang, Chenghuan Huang, Zhenyi Zheng, Jiangsu Du, Ziyang Ma, Jing Lyu, and Yutong Lu. Beyond Few-Step Inference: Accelerating Video Diffusion Transformer Model Serving with Inter-Request Caching Reuse. *arXiv preprint arXiv:2604.04451*, 2026. 2, 3

[17] Joseph Liu, Joshua Geddes, Ziyu Guo, Haomiao Jiang, and Mahesh Kumar Nandwana. SmoothCache: A Universal Inference Acceleration Technique for Diffusion Transformers. *arXiv preprint arXiv:2411.10510*, 2024. 3

[18] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From Reusing to Forecasting: Accelerating Diffusion Models with TaylorSeers. *arXiv preprint arXiv:2503.06923*, 2025. 1, 3, 4

[19] Xuewen Liu, Zhikai Li, Jing Zhang, Mengjuan Chen, and Qingyi Gu. Rectified SpaAttn: Revisiting Attention Sparsity for Efficient Video Generation. *arXiv preprint arXiv:2511.19835*, 2025. 3, 6

[20] Cheng Lu and Yang Song. Simplifying, Stabilizing and Scaling Continuous-Time Consistency Models. *arXiv preprint arXiv:2410.11081*, 2024. 3

[21] Chengtao Lv, Yumeng Shi, Yushi Huang, Ruihao Gong, Shen Ren, and Wenya Wang. Light Forcing: Accelerating Autoregressive Video Diffusion via Sparse Attention. *arXiv preprint arXiv:2602.04789*, 2026. 3, 4

[22] Yuexiao Ma, Xuzhe Zheng, Jing Xu, Xiwei Xu, Feng Ling, Xiawu Zheng, Huafeng Kuang, Huixia Li, Xing Wang, Xuefeng Xiao, Fei Chao, and Rongrong Ji. Flow caching for autoregressive video generation. *arXiv preprint arXiv:2602.10825*, 2026. 3

[23] Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast Video Generation with Magnitude-Aware Cache. *arXiv preprint arXiv:2506.09045*, 2025. 1, 3, 4

[24] Youssef Mansour, Enis Simsar, Fadime Sener, Markos Georgopoulos, Albert Pumarola, Ali Thabet, and Edgar Schoenfeld. UnStep: Training-Free Acceleration of Causal Video Diffusion with Fewer Steps Than Distillation. *arXiv preprint arXiv:2609.32518*, 2026. 3, 7

[25] Mustafa Munir, Huy Vu, Shreyas Misra, Rohit Jena, Sajad Norouzi, Ali Taghibakhshi, Anis Ahmad, Anjul Patney, Pavlo Molchanov, and Nima Tajbakhsh. Accelerating Video Diffusion via Training-Free Trajectory Routing. *arXiv preprint arXiv:2609.30096*, 2026. 3

[26] 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. 3

[27] Jinliang Shen, Lianghao Su, Zheming Li, Kang He, ZiLiang Lai, Yanbing Jiang, and Chengru Song. HeadCast: Casting Attention Heads for Efficient Autoregressive Video Generation. *arXiv preprint arXiv:2607.20125*, 2026. 3

[28] Jian Tang, Jiawei Fan, Qingbin Liu, Zheng Wei, Jiang Bian, and Zang Li. FIS-DiT: Breaking the Few-Step Video Inference Barrier via Training-Free Frame Interleaved Sparsity. *arXiv preprint arXiv:2605.11869*, 2026. 2, 3, 8

[29] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, et al. Wan: Open and Advanced Large-Scale Video Generative Models. *arXiv preprint arXiv:2503.20314*, 2025. 2, 3

[30] Tingyan Wen, Chenqian Yan, Xurui Peng, Xiazhang Fang, Shuai Wang, Xueqian Wang, and Songwei Liu. SelfLift: Accelerating Few-Step Diffusion via Self-Recovering Resolution Transition. *arXiv preprint arXiv:2609.02036*, 2026. 3

[31] Zihao Wu, Bohan Zeng, and Yuanxing Zhang. Fine-Grained Caching for Diffusion Transformers with Few Calibration Conditions. *arXiv preprint arXiv:2512.05134*, 2025. 3

[32] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, Jianfei Chen, Ion Stoica, Kurt Keutzer, and Song Han. Sparse VideoGen: Accelerating Video Diffusion Transformers with Spatial-Temporal Sparsity. *arXiv preprint arXiv:2502.01776*, 2025. 1, 3

[33] Boxun Xu, Yuming Du, Zichang Liu, Siyu Yang, Ziyang Jiang, Siqi Yan, Rajasi Saha, Albert Pumarola, Wenchen Wang, and Peng Li. Sparse Forcing: Native Trainable Sparse Attention for Real-time Autoregressive Diffusion Video Generation. *arXiv preprint arXiv:2604.21221*, 2026. 3, 4, 8

[34] Kerry Xu, Param Misra, Zachary Stenglein, Athri Karunamurthy, Fanjiang Ye, and Yuke Wang. Frequency-Step Cache: Separating Synchronization and Repair for Efficient Autoregressive Video Diffusion. *arXiv preprint arXiv:2610.15922*, 2026. 3

[35] Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. XAttention: Block Sparse Attention with Antidiagonal Scoring. *arXiv preprint arXiv:2503.16428*, 2025. 4

[36] Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time Interactive Long Video Generation. *arXiv preprint arXiv:2509.22622*, 2025. 3

[37] Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Jianfei Chen, Song Han, Kurt Keutzer, and Ion Stoica. Sparse VideoGen2: Accelerate Video Generation with Sparse Attention via Semantic-Aware Permutation. *arXiv preprint arXiv:2505.18875*, 2025. 3, 6

[38] Shuo Yang, Lihao Fang, Yi Zhang, Haixiang Wang, Xincheng Ye, Shufan Chen, Jipeng Guo, and Youqing Wang. Rethinking What to Cache in Few-Step Diffusion Transformers: Solver-Aware Target Selection. *arXiv preprint arXiv:2610.03577*, 2026. 3

[39] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T. Freeman, and Taesung Park. One-step Diffusion with Distribution Matching Distillation. *arXiv preprint arXiv:2311.18828*, 2023. 3

[40] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved Distribution Matching Distillation for Fast Image Synthesis. *arXiv preprint arXiv:2405.14867*, 2024. 3

[41] Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From Slow Bidirectional to Fast Autoregressive Video Diffusion Models. *arXiv preprint arXiv:2412.07772*, 2024. 1, 3

[42] Yixiao Zeng, Jianlei Zheng, Chaoda Zheng, Shijia Chen, Mingdian Liu, Tongping Liu, Tengwei Luo, Yu Zhang, Boyang Wang, Linkun Xu, Siyuan Lu, Bo Tian, and Xianming Liu. X-Cache: Cross-Chunk Block Caching for Few-Step Autoregressive World Models Inference. *arXiv preprint arXiv:2604.20289*, 2026. 2, 3

[43] Jintao Zhang, Haoxu Wang, Kai Jiang, Shuo Yang, Kaiwen Zheng, Haocheng Xi, Ziteng Wang, Hongzhou Zhu, Min Zhao, Ion Stoica, Joseph E. Gonzalez, Jun Zhu, and Jianfei Chen. SLA: Beyond Sparsity in Diffusion Transformers via Fine-Tunable Sparse-Linear Attention. *arXiv preprint arXiv:2509.24006*, 2025. 3

[44] Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. SpargeAttention: Accurate and Training-free Sparse Attention Accelerating Any Model Inference. *arXiv preprint arXiv:2502.18137*, 2025. 1, 3, 4

[45] Jintao Zhang, Kaiwen Zheng, Kai Jiang, Haoxu Wang, Ion Stoica, Joseph E. Gonzalez, Jianfei Chen, and Jun Zhu. TurboDiffusion: Accelerating Video Diffusion Models by 100-200 Times. *arXiv preprint arXiv:2512.16093*, 2025. 3

[46] Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast Video Generation with Sliding Tile Attention. *arXiv preprint arXiv:2502.04507*, 2025. 3, 8

[47] Kaiwen Zheng, Yuji Wang, Qianli Ma, Huayu Chen, Jintao Zhang, Yogesh Balaji, Jianfei Chen, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Large Scale Diffusion Distillation via Score-Regularized Continuous-Time Consistency. *arXiv preprint arXiv:2510.08431*, 2025. 1, 3

[48] Xin Zhou, Dingkang Liang, Kaijin Chen, Tianrui Feng, Xiwu Chen, Hongkai Lin, Yikang Ding, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. Less is Enough: Training-Free Video Diffusion Acceleration via Runtime-Adaptive Caching. *arXiv preprint arXiv:2507.02860*, 2025. 3

[49] Lunjie Zhu, Yushi Huang, Xingtong Ge, Yufei Xue, Zhening Liu, Yumeng Zhang, Zehong Lin, and Jun Zhang. Flash-VAED: Plug-and-Play VAE Decoders for Efficient Video Generation. *arXiv preprint arXiv:2602.19161*, 2026. 3

[50] Chang Zou, Changlin Li, Yang Li, Patrol Li, Jianbing Wu, Xiao He, Songtao Liu, Zhao Zhong, Kailin Huang, and Linfeng Zhang. DisCa: Accelerating Video Diffusion Transformers with Distillation-Compatible Learnable Feature Caching. *arXiv preprint arXiv:2602.05449*, 2026. 2, 3

StudentProbe – Supplementary Material

Runguo Li
University of Illinois Urbana-Champaign
runguol2@illinois.edu

This supplement holds StudentProbe's registration record (Appendix A), the full results behind each section of the paper, the method attempts and released alternatives tested after the main results (Appendix S), and the details of StudentProbe's probes (Appendix U).

A. Registration record

Every prediction and diagnostic was recorded before the runs it concerns. Table 1 lists the registration and scoring records. The log was kept privately and is not third-party timestamped; we include it here in full, with anonymized record identifiers, so that its order can be checked.

Item	Registered	Scored
P1–P6, decision rules, held-out prompts	R1 (10-07)	–
P2 on klein	(above)	R2 (10-07)
P1, P4 on LTX	(above)	R3 (10-07)
Scope change (withdrawn P6 comparisons)	R4 (10-08)	–
P2 on LTX, P3, P6	(above)	R5 (10-08)
D2 (mechanism of late skips)	R6 (10-08)	R7 (10-08)
P5	(P1–P6)	R7 (10-08)
D3 (signals ranking skip cost)	R8 (10-08)	R9 (10-08)
D4 (8-prompt sweep)	R10 (10-08)	R11 (10-08)
D5 (held-out quality, AnyFlow)	R12 (10-08)	R13 (10-08)
D6 (held-out quality, LTX)	R14 (10-09)	R15 (10-09)
D7 (student-tuned caches vs StudentProbe's sweep)	R16 (10-12)	R17 (10-13)
D8 (why D7b failed)	R17 (10-13)	R18 (10-13)
D9 (held-out scale, rCM 14B)	R19 (10-13)	R20 (10-14)
D9h, D9i rerun on B200 (servers reclaimed)	R21 (10-29)	R22 (10-29)
D10 (14B teacher at matched noise level)	R23 (10-29)	R24 (10-29)
D11 (system axis on a B200)	R23 (10-29)	R25 (10-29)
D13 (concentration at equal key count)	R26 (10-29)	R27 (10-29)
D14 (sparse attention at densities 0.1–0.5)	R28 (10-29)	R29 (10-30) (B200)
H100 timing (D14a, D14c, D14d)	(above)	R30 (10-30)
D14d addendum (after a 4-prompt validation)	R31 (10-29)	(both above)
D15 (VBench dimensions, B200 replication)	R32 (10-30)	R33 (10-31)
D16 (Wan2.2, HunyuanVideo-1.5 third-party students)	R34 (10-31)	R35 (10-31)
plan details and scorer, before stage T	R36 (10-31)	(above)
D17 (frame sparsity, lightweight decoder, stacks, 720p)	R37 (10-31)	R38 (11-01)
reading of FIS-DiT's equations, rule before the second run	R39 (11-01)	(above)
D18 (allocating compute across calls)	R40 (11-01)	R38 (11-01)
planner keeps the last call full, before stage T	R41 (11-01)	(above)
D19 (two calls, lightweight decoder, compilation)	R38 (11-01)	R42 (11-02)
moved to one PRO6000 BW type, before any run	R43 (11-01)	(above)
D20 (oracle and semantic-aware selectors)	R44 (11-02)	R45 (11-02)
D21 (same states, error gain, coverage seeds)	R44 (11-02)	R45 (11-02)
large models move to A100s, token oracle on 12 prompts	R46 (11-02)	(above)
D22 (rCM's last skip: reuse vs stopping) and D23 (same states, held out)	R47 (11-03)	R48 (11-04)
all on H100 NVLs, before any rCM-1.3B run	R49 (11-03)	(above)
D24–D26 (fine grid, untouched prompts, equal-budget AR)	R50 (11-04)	R51 (11-05)
D27 (parallel decoding of the full VAE)	R52 (11-06)	stopped after stage V, R53 (11-06)
D28 (sensitivity-weighted caches)	R54 (11-06)	R55 (11-06)
TeaCache criterion amended, scorer, before stage T was read	R56 (11-06)	(above)
D29 (released decoders) and D30 (compensated sparse attention)	R57 (11-06)	R58 (11-06)
D30's R1-eq, after stage V and before stage H; its drop rate frozen	R59, R60 (11-06)	(above)
D31 (each attention path with every key kept)	R61 (11-07)	R62 (11-07)
RTX 3090s before any run; top-k skipped when every key is kept	R63, R64, R65	(above)
D12 (viewer study)	R66 (10-29)	not yet run
A6000 stimuli; released vs baseline groups; owner's test excluded	R67, R68, R69	(awaiting ethics review)

Table 1. Registration and scoring records (2026; dates as month–day). Record identifiers are anonymized; no repository identifiers are disclosed.

Prediction outside the final scope (P2). P2 predicted the sign of the paired difference between residual reuse and an equal-cost modification of the sampling grid. The prediction was derived from the student-specific redundancy gap. Scored as registered, it was correct in 3 of 4 non-tie cases (75%): 1 of 2 on klein, 2 of 2 on LTX, with 4 LTX ties. This is between the registered thresholds for a supported (80%) and an abandoned (60%) predictive claim. The paper does not use this result, since the comparison lies outside its scope.

Scope change. Before any held-out comparison was computed, six of the ten registered P6 comparisons (residual reuse vs equal-cost grid modifications) were withdrawn: they served only that question, which left the paper's scope, as P2's did. No held-out comparison was computed for the withdrawn conditions, and the thresholds of the remaining four were kept. Two generation runs for them had started: AnyFlow with reuse of call 1 had completed and reuse of call 2 had produced 7 of 66 videos; neither was scored.

Decision rules applied. P1 failed and P4 held, so the registered rule selected the "narrow" branch: state the failing finding with its counter-example and continue. P6 kept 4 of 4 signs, above the abandon threshold of 60%.

Failed predictions. Twenty-eight lettered predictions in Tab. 5 of the paper, each a registered pass/fail criterion, failed, in three groups. Some extrapolated beyond the models we had studied: INT8's gain at 13B (P5), very short attention on a B200 (D11), FBCache's skip treated as plain reuse (D7) and four predictions for the third-party students (D16). Some tested explanations the data did not support: interval width as the cause of lost redundancy (P1; LTX's small early steps were predicted to reach cos $\geq$ 0.90), a low-cost signal that ranks skips on every student (D3), coverage compared across key counts (D13), attention mass as the cause of the sparse gap (D20), anchoring as the source of AR subject consistency (D26), and compensation for the unselected keys as a remedy for it (D30a). The rest concerned our own proposals: two calls and the lightweight

decoder cost less quality than we predicted (D17; imaging quality was predicted to fall), the allocator did not improve on the grid (D18), the baseline's decoder and compilation had quality costs where we had predicted none (D19), its aesthetic costs on untouched prompts were larger than predicted (D25), weighting a cache's indicator by the student's measured sensitivity did not make it rank skips (D28a), and a released pruned decoder did not score better than the tiny decoder (D29a, D29b; Appendix S). D27 stopped after its validation run and D28b–d were not run. CIs are per comparison and not adjusted for multiplicity; each registered prediction fixes its comparison in advance.

Data history. Each registration states which results its author had seen (the "Seen before" field of the registration log). Table 2 gives the first and the later uses of each prompt set. A held-out set is strongest at its first use; later H rows reuse the 33 held-out prompts for new measurements, and the baseline's grid and decoder costs had been seen on them (D17) before D19 combined them.

Set	Prompts	First use	Later uses
Tuning	8	10-07, Wan, klein	sweeps, probes, stage V
Development	33	10-07, Wan, klein	exploratory and diagnostic runs
Held out	33	10-08, P6	most later H and S rows
User study	12 held out	D12	–
Untouched	33	11-05, D25	D26

Table 2. Prompt sets (VBench), with two seeds each for evaluation runs.

B. Quality effect of every skip position

Table 3 gives the fidelity cost of every skip position and how often each low-cost signal orders it, and Tab. 4 extends it to all reference-free scores.

On LTX, tying calls 5 and 6, whose fidelity costs lie below the compile reference, changes no verdict (S2: 5/14 pairs; D6: 14/14). Values are paired differences against each student's unaccelerated output over the development prompts (`gmsa/analysis/quality_table.py` regenerates them, and Tab. 3, from the raw metric files). rCM's sampler draws fresh noise before every call, so a reused residual carries the previous call's noise into the output; D22 shows that this is what degrades its last skip. An asterisk marks a 95% bootstrap CI (2000 resamples over prompts) that excludes 0. Only rCM's last-call skip severely degrades the output (Fig. 5 of the paper). AnyFlow's last-call skip is the only other skip that lowers aesthetics and subject consistency together.

Figure 1 shows what AnyFlow's quality effect means at frame level. The aesthetic drop of the last-call skip is a small

		skipped call k						concordant pairs			
Student		1	2	3	4	5	6	S1	S2	S3	S4
AnyFlow	LPIPS	.210	.078	.074				3/3	2/3	3/3	1/3
	Δaes.	−.04	−.01	**−.10**							
rCM	LPIPS	.359	.283	.575				1/3	1/3	1/3	3/3
	Δaes.	+.01	**+.08**	**−2.00**							
klein	LPIPS	.225	.106	.092				3/3	3/3	3/3	1/3
	Δaes.	−.04	+.01	+.05							
LTX	LPIPS	.276	.143	.132	.079	.024	.009	8/15	6/15	14/15	13/15
	Δaes.	+.02	+.01	+.01	−.02	**+.09**	**+.08**				

Table 3. Fidelity cost (LPIPS to the unaccelerated output; 95% CIs are ±0.003–0.04) and quality effect (paired Δ aesthetic score; bold: 95% CI excludes 0) of skipping one call, on the 33 development prompts, one seed. Right: position pairs each low-cost signal (S1–S4, defined in Sec. 5.1 of the paper) orders as the fidelity cost does. It was Tab. 3 of the paper before the comparison with student-tuned caches took its place; Fig. 4 of the paper plots it.

average shift (-0.10 on a 10-point scale) and is not visible in single frames. Per-prompt changes are noisy, and on some prompts skipping call 2 lowers the score more. This is why we recommend confirming quality on 30 or more prompts.

Student	k	Δaesthetic	ΔCLIP-T	ΔDINO cons.	Δmotion
AnyFlow	1	−0.035	+0.200	−0.001	−0.015
	2	−0.008	−0.084	+0.001	−0.072*
	3	−0.102*	+0.242	−0.007*	−0.032
rCM	1	+0.012	−0.097	−0.000	−0.686*
	2	+0.083*	+0.030	−0.004	−0.460*
	3	−2.000*	−0.481	−0.199*	+19.74*
klein	1	−0.039	+0.142	–	–
	2	+0.006	+0.284	–	–
	3	+0.048	+0.115	–	–
LTX	1	+0.022	+0.483	−0.001	−0.290
	2	+0.010	−0.011	−0.004	−0.102
	3	+0.010	+0.038	−0.001	−0.192
	4	−0.018	−0.059	+0.000	−0.284
	5	+0.090*	+0.097	+0.002*	−0.041
	6	+0.081*	−0.007	+0.002*	+0.044*

Table 4. Quality effect of skipping call k (paired Δ vs the unaccelerated output). LTX call 2's ΔDINO is borderline (95% CI $[-0.008, +0.000]$).

Held-out replication (D5). On the held-out prompts with two seeds (66 videos per position), AnyFlow's skip costs are 0.206, 0.074 and 0.073 for calls 1–3. The aesthetic changes are -0.060, -0.015 and -0.148*, and the DINO changes -0.006*, -0.000 and -0.010*. The registered predictions concerned the fidelity order and the aesthetic score, and all three held. The small DINO decrease at call 1 was not predicted and is absent on the development prompts (-0.001).

Held-out replication (D6). On the same held-out set, LTX's skip costs for calls 1–6 are 0.265, 0.123, 0.116, 0.074, 0.022 and 0.009, which orders all 15 position pairs as on the

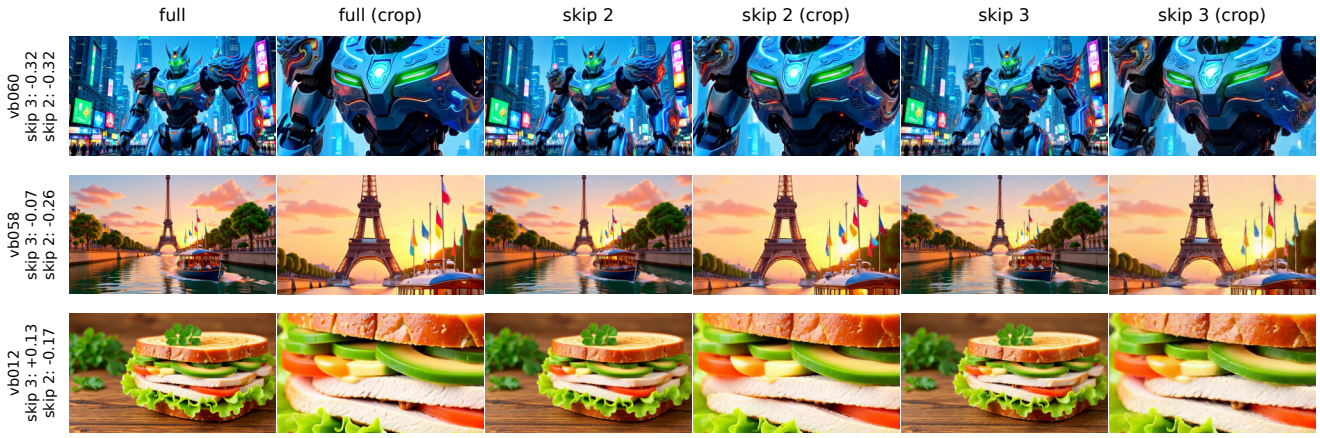

Figure 1. AnyFlow, middle frame and center crop: unaccelerated output, skip of call 2, skip of call 3. Rows are the development prompts at the 10th, 50th and 90th percentile of the skip-3 aesthetic change, chosen by rank, not by appearance; the labels give each prompt's aesthetic change for both skips.

development prompts. The aesthetic changes are -0.002, -0.015, $+0.028$, $+0.037$*, $+0.081$* and $+0.055$*, and no DINO change is negative with a CI excluding 0. All four registered predictions held. Three changes were not predicted: a slight aesthetic gain at call 4 (-0.018 on the development prompts), a CLIP-T gain at call 3 ($+0.28$*) and a slight motion decrease at call 4 (-0.029*).

Student-tuned caches vs StudentProbe's sweep (D7, D8). Table 5 lists the registered comparison on the held-out set (Tab. 3 of the paper shows its main columns). StudentProbe's choices (AnyFlow call 2, rCM call 2, LTX call 6) are the 8-prompt sweep's lowest-drift positions, moved to the next-lowest where the development quality check failed. TeaCache's schedule is prompt-independent and coincides with existing fixed-schedule runs (AnyFlow call 1, rCM call 2), which are reused, as registered. FBCache's first block costs $1/30$ (Wan) or $1/48$ (LTX) of a call per skip; on AnyFlow, 8 of the 66 videos skip no call at the tuned threshold. D7a held; D7b and D7c failed (Tab. 5). D8 shows that the held-out prompts are about as sensitive as the development prompts to plain reuse of rCM's last call (-1.74 vs -2.00), so D7b failed because FBCache's skip is a milder approximation at that call. At AnyFlow's call 2, in contrast, FBCache's skip lowers aesthetics slightly more than plain reuse.

C. LTX timing details

Table 6 lists every LTX timing. Each configuration ran on the same A800 with 2 warm-up and 5 timed runs. The standard deviation of end-to-end time is at most 0.4 s everywhere. The two sparse-attention runs measured 1.046× and 1.051× end to end, and unfused eager INT8 is 2.1–3.3× slower.

D. Held-out scale: rCM on Wan2.1-14B (D9)

rCM's 4-step Wan2.1-14B 480p student comes from the same reproduction as our 1.3B rCM. It runs with the 1.3B rCM's sampler (the official script uses the same mid times for both sizes), text encoder, bf16 VAE and output size; only the DiT changes (width 5120, 40 blocks, 40 heads vs 1536, 30, 12). Its predictions were registered before any run (Tab. 1), and one video on a tuning prompt was inspected for validity before the other jobs started. Timing ran on a second A800 80GB PCIe, on which the 1.3B rCM re-times at 11.32 s (11.35 s on the baseline GPU). Table 7 lists the timings and the skip and sparse-attention runs. The latency probe puts 9.5% of the time in the VAE, 46.4% in self-attention and 0.8% in cross-attention; the block-sparse kernel is 3.15× faster than dense attention at density 0.2 (5.06× at 0.1). Subtracting the probe's attention time, compiled INT8 speeds up the rest of the DiT by about 1.37× (an estimate: the probe's attention time is eager). The A800 servers were reclaimed before the last two predictions were scored, and every D9 output went with them. As registered before any further run (Tab. 1), D9h and D9i were run again on an NVIDIA B200 with the same code, model files, prompts, seeds and scorer; every paired comparison is within B200 runs, and no B200 timing is used. A validity check came first: on the B200, the 8-prompt sweep orders the calls as on the A800, for the 1.3B rCM (LPIPS 0.356, 0.267, 0.573 vs 0.404, 0.311, 0.586) and for rCM-14B (0.395, 0.212, 0.487 vs 0.419, 0.225, 0.501). Tuned by the D7 rule, FBCache skips call 3, the costliest, in all 8 tuning videos (D9h); its signal falls from call 1 to call 3 (1.142, 1.047, 0.943), as on the 1.3B rCM. On the held-out set, sparse attention drifts

Student	Method	Run	Skipped call (videos)	DiT calls	LPIPS→own	Δaes.	ΔDINO	ΔLPIPS vs prot.	Δaes. vs prot.
AnyFlow	StudentProbe	reuse	2 (66)	3.00	0.074	−0.015	−0.000	–	–
	FBCache	τ^*=0.221	2 (57), 1 (1), none (8)	3.15	0.069	−0.058*	−0.001	−0.005	−0.042*
	TeaCache	τ^*=5.69	1 (66)	3.00	0.206	−0.060	−0.006*	+0.131*	−0.045
rCM	StudentProbe, TeaCache	reuse; τ^*=0.341	2 (66)	3.00	0.284	−0.054	−0.003	–	–
	FBCache	τ^*=1.145	3 (66)	3.03	0.300	−0.555*	−0.071*	+0.016*	−0.501*
	plain reuse (D8)	reuse	3 (66)	3.00	0.560	−1.739*	−0.185*	+0.276*	−1.685*
LTX	StudentProbe	reuse	6 (66)	6.00	0.009	+0.055*	+0.001*	–	–
	FBCache	τ^*=0.181	3 (66)	6.02	0.117	+0.029	+0.000	+0.107*	−0.026

Table 5. Registered comparison on the held-out prompts (33 prompts × 2 seeds), thresholds tuned on the 8 tuning prompts to skip one call per video. Δ: paired difference vs the unaccelerated output (aes., DINO) or vs StudentProbe's run (prot.); *: 95% cluster-bootstrap CI excludes 0. `gmsa/analysis/d7_heldout.py` and `d8_heldout.py` regenerate the table.

Model	Configuration	e2e (s)	DiT (s)	peak mem. (GB)
distilled	as released	7.77 ± 0.08	6.78	39.6
	compiled	6.57	5.61	39.6
	compiled + INT8	5.51	4.55	27.5
	one call skipped (reuse)	6.70	5.76	39.7
	sparse d=0.2, call 0 dense	7.43	6.47	39.6
dev	as released	59.28 ± 0.07	58.25	39.6
	compiled	49.73	48.69	39.6
	compiled + INT8	40.16	39.14	27.5
	sparse d=0.2, 6 dense calls	56.38	55.35	39.6
	15 steps	29.78 ± 0.22	28.76	39.6

Table 6. LTX-Video 0.9.7 timings (single-scale pass, 512×704×121). The VAE takes 0.53 s in every configuration.

Configuration	e2e (s)	DiT (s)	peak (GB)	Amdahl	measured
as released	43.42 ± 0.14	39.00	42.0		
compiled	40.19	35.75	42.0		1.080×
compiled + INT8	36.03	31.61	28.6		1.205×
call 2 skipped (reuse)	33.68	29.26	42.6	1.290×	1.289×
one call fewer	33.69	29.27	42.0	1.290×	1.289×
sparse d=0.2, call 0 dense	33.57	29.13	42.0	1.311×	1.294×

	LPIPS→own		held-out Δ vs as released		
Skipped call / method	tuning	held-out	aesthetic	DINO	motion
call 1	.419	.341	+**.08**	+.001	−**0.13**
call 2	.225	.179	−.01	−**.004**	−**0.23**
call 3	.501	.453	−**1.42**	−**.257**	+**7.7**
sparse d=0.2, call 0 dense		.342	−.05	−**.023**	−**0.49**
sparse d=0.2, call 0 dense (B200)		.337	−**.095**	−**.024**	−**0.46**
one call fewer (B200)		.161	+.005	+.002	−**0.27**

Table 7. rCM on Wan2.1-14B. Top: timings (2 warm-up, 5 timed runs; VAE 4.11 s throughout), the latency probe's predicted and the measured end-to-end speedups over the as-released run. Bottom: fidelity cost on the 8 tuning prompts and on the held-out set (66 videos), and paired quality changes on the held-out set (bold: 95% CI excludes 0). Held-out LPIPS CIs are ±0.015–0.021. Rows marked B200 were generated and scored on a B200 after the A800 servers were reclaimed (D9h, D9i); every difference is paired within one GPU type.

further than one call fewer at equal speed (paired +0.176 [0.167, 0.185], D9i; speedups 1.294× and 1.289× on the A800). One call fewer shows no significant decline in aesthetics or DINO consistency, whereas sparse attention lowers both (Tab. 7). The compile reference of rCM-14B on the B200 is 0.130.

E. The 14B pair at matched noise level (D10)

The Wan2.1-T2V-14B teacher combines the official 14B DiT with the text encoder, VAE, tokenizer and scheduler of the 1.3B release, which are byte-identical in the two releases, and samples as released (UniPC, 50 steps, shift 5, CFG 5). Teacher and student ran on a B200 on the 4 tuning prompts. Each student call is matched to the teacher step nearest in t (steps 3, 13, 23 and 38). Table 8 lists coverage and cosines. At 14B the student is as concentrated as its teacher at every call, and the descriptive interval ratio is 55% (summed over calls 1–3, the teacher's cosine falls by 0.90 from one step to the student's interval and the student's lies a further 0.73 below), against 59% for the 1.3B rCM (56% for klein, 84% for AnyFlow) with the rounded values of Tab. 2 of the paper; the ratio does not quantify a causal contribution of interval width. Per prompt (descriptive), the coverage difference stays within 0.020 at every call, and the interval ratio ranges from 50% to 62% (49.6% for one prompt): at 14B the ratio is about one half, closer to the registered threshold of 50% for some prompts than its mean suggests.

Call	t student / teacher	cov@0.1 student / teacher	student cos	teacher same int.	teacher one step
0	988 / 987	.775 / .789	–	–	–
1	932 / 934	.808 / .817	.310	.593	.977
2	852 / 854	.818 / .827	.463	.754	.984
3	608 / 613	.830 / .834	.548	.704	.985

Table 8. rCM-14B vs Wan2.1-T2V-14B (B200, 4 tuning prompts). Oracle coverage at 10% of the keys at the matched call; block-residual cosine to the previous call for the student, and for the teacher over the same interval and over one step. `gmsa/analysis/d10_teacher14.py` regenerates the table.

F. Latency split as released

Table 9 gives the per-model latency split behind Sec. 4.3 of the paper.

Model	e2e (s)	VAE	self-attn	sparse kern.	INT8
Wan teacher (50)	192	3.0%	59%	3.10×	1.03×
AnyFlow (4)	12.8	31.1%	33%	3.08×	1.04×
rCM (4)	11.4	35.3%	39%	3.10×	1.04×
rCM-14B (4)	43.4	9.5%	46%	3.15×	1.13×
Self-Forcing (AR)	12.4	32.4%	30%	0.40×	0.95×
AnyFlow-FAR (AR)	15.3	26.3%	–	–	–
klein (4)	1.16	13.8%	13%	0.58×	–
LTX dev (30)	59.3	0.9%	15%	1.00×	1.24×
LTX distilled (7)	7.77	6.8%	14%	1.08×	1.23×

Table 9. Latency split as released, one A800 per measurement (PyTorch default numerics): e2e time and VAE share from timing runs; self-attention share and sparse kernel speedup (density 0.2, own shape) from the latency probe. INT8: DiT speedup of compiled W8A8 over compiled bf16.

G. The system axis on a B200 (D11)

D11 repeats the system measurements on one Slurm-exclusive NVIDIA B200 (180 GB) with the same code and PyTorch's default numerics; other jobs may use the node's other GPUs. The 1.3B rCM, timed first and last, took 2.79 s both times. Before registration we had seen only single smoke videos, no timing run. Table 10 lists the latency split, the kernel, INT8 and compile speedups, and the 12 registered Amdahl predictions. Every step prediction is within 0.4%, and the Wan sparse predictions within 3.0%. The LTX sparse predictions fail (D11a): at LTX's short self-attention (5,632 tokens) the isolated benchmark gives 2.9 ms per sparse call (0.35 ms dense), whereas inside the model a sparse call costs about 1.5 ms (the sparse run's extra DiT time over its sparse calls, plus the probe's dense time per call), so the benchmark overstates the slowdown; we did not isolate the cause of this gap. On this GPU, the teacher-to-AnyFlow speedup is 10.5× (15.0× on the A800), because the decoder speeds up less than the DiT.

H. Attention concentration at equal key count (D13)

Oracle coverage at $p\%$ is a fraction of the attended keys, so two settings with different key counts can differ in coverage because of the count alone. D13, registered before its runs, adds the attention mass of a fixed number of top keys, on the same sampled queries, layers and heads (B200, 4 tuning prompts, seed 0). With more keys, coverage at a fixed fraction rises if the mass stays on a fixed set of keys, and the mass of a fixed number of keys falls if it spreads over the new keys; a real difference in concentration should show in both.

Both registered comparisons fail (Tab. 11). From the first AR chunk to the last, coverage rises by 0.16 (AnyFlow-FAR) and 0.10 (Self-Forcing), but the mass of the top 256 keys falls by 0.09 and 0.05; between AnyFlow and LTX, coverage differs by 0.14 and the top-256 mass by -0.01. These differences therefore come from the key count, and the paper compares concentration only at equal key count. Within each teacher–student pair the key count is equal, and the invariance replicates on the B200: AnyFlow vs the Wan teacher within 0.006 at $t < 1000$, LTX distilled vs dev within 0.019, and Self-Forcing's last chunk, which attends to as many keys as the teacher, within 0.014 at $t < 1000$. The B200 reproduces the A800 coverage values of the paper within 0.002.

I. Sparse attention at several densities (D14)

D14 asks whether any density, with the first one or two of the four calls dense, lets the block-sparse kernel of the paper match one call fewer. It was registered before any of its runs; the addendum D14d was registered after a 4-prompt validation run and before the full run, and its record lists the validation values it had seen. Quality runs use the B200 (33 dev prompts, seed 0, LPIPS to each model's own base on the same GPU). The B200 cannot time sparse attention (kernel 0.76–0.87× dense at densities 0.1–0.3), so the speed runs use one dedicated H100, which passed the registered gate: kernel at density 0.2 2.08× dense at the Wan shape ($\geq 1.5\times$; 2.93×, 1.65×, 1.15× at 0.1, 0.3, 0.5). On the H100, self-attention takes 21.5% (AnyFlow) and 26.9% (rCM) of the base latency, the end-to-end standard deviation is at most 0.008 s, and the 1.3B rCM, timed first and last, took 4.88 s both times. With 256 key blocks, the realized densities are 0.102, 0.203, 0.301 and 0.500. As a validity check, the B200 reproduces the A800 values of the paper within 0.001 on AnyFlow and 0.028 lower on rCM (both settings; registered tolerance 0.03).

Table 12 gives every value. D14a: every setting at least 0.95× as fast as one call fewer (AnyFlow: density 0.1 with one dense call; rCM: 0.1 and 0.2 with one, 0.1 with two) drifts further. D14b: AnyFlow at 0.2 with two dense calls drifts less than one call fewer. D14c: all 16 Amdahl predictions are within 0.8% of the measurements. D14d: a setting with the first w calls dense and realized density d saves at most $(4-w)\,a\,(1-d)$ calls with an ideal kernel (time proportional to the kept blocks, no selection cost), where a is self-attention's share of the DiT time (B200: 0.288, 0.417; H100: 0.364, 0.512 for AnyFlow, rCM). The eight settings that drift no more than one call fewer save at most 0.82 calls (rCM, 0.2 with two dense calls, H100), against one for one call fewer; they would need $a \geq 0.63$. Even an infinitely fast kernel with the first call dense gives at most 1.19× (AnyFlow) and 1.25× (rCM) on the H100, against 1.43× and 1.36× measured for two calls fewer. Sparse settings change the mean aesthetic score by at most 0.11 and CLIP-T by at most 0.27 from the base.

Model	e2e (s)	VAE	self-attn	sparse kern.	INT8	compile	step run: pred. / meas.	sparse run: pred. / meas.
Wan teacher (50)	35.30	4.4%	37.4%	0.97×	0.968×	1.362×	1.893 / 1.896 (25 steps)	0.989 / 1.020
AnyFlow (4)	3.36	38.7%	15.3%	0.87×	0.965×	1.705×	1.152 / 1.154	0.984 / 0.998
rCM (4)	2.79	46.7%	18.9%	0.86×	0.967×	1.307×	1.128 / 1.124	0.977 / 1.006
rCM-14B (4)	7.93	16.5%	29.6%	1.10×	1.156×	1.223×	1.252 / 1.251	1.021 / 1.026
LTX dev (30)	12.23	1.3%	8.9%	0.14×	1.210×	1.478×	1.915 / 1.921 (15 steps)	0.698 / 0.931
LTX distilled (7)	1.92	8.4%	6.4%	0.12×	1.194×	1.460×	1.116 / 1.120	0.714 / 0.855

Table 10. The system axis on a B200 (D11; one Slurm-exclusive GPU, 2 warm-up and 5 timed runs, e2e std ≤ 0.01 s). Columns as in Tab. 9, plus the DiT speedup of `torch.compile` and the probe's predicted and the measured end-to-end speedups (students: call 2 reused; LTX distilled: call 2 reused; sparse: density 0.2, first call(s) dense as on the A800). `gmsa/analysis/d11_b200.py` regenerates the table.

Setting	keys	cov@0.1	top 256	top 1024
AnyFlow-FAR, chunk 0, $t = 1000$	1,560	0.730	0.803	0.972
AnyFlow-FAR, chunk 7, $t = 1000$	17,550	0.888	0.713	0.842
Self-Forcing, chunk 0, $t = 1000$	4,680	0.768	0.690	0.866
Self-Forcing, chunk 6, $t = 1000$	32,760	0.871	0.637	0.772
Wan teacher, $t = 999$	32,760	0.773	0.422	0.610
AnyFlow, mean over calls	32,760	0.877	0.638	0.776
LTX distilled, mean over calls	5,632	0.740	0.651	0.809

Table 11. Coverage at 10% of the keys and mass of the top 256 and 1024 keys (D13).

J. VBench dimensions and B200 replication (D15)

D15 was registered before any of its runs and answers two questions: do the held-out quality verdicts, measured on A800s, hold on another GPU, and do VBench's own quality dimensions agree with them? All runs are generated on a B200 and paired by (prompt, seed) against the same model's B200 base (33 held-out prompts, seeds 0 and 1, 66 videos per run; cluster bootstrap over prompts, 2000 resamples). VBench 0.1.5 ran one video at a time and matched its official `evaluate()` exactly on the validation videos. D15a: all 7 significant verdicts replicate and 13 of 14 no-drop verdicts hold; the flip is rCM-14B skip 2, aesthetic -0.067 $[-0.127, -0.014]$ (A800 -0.006). D15b: rCM's last skip lowers imaging quality by 0.129 at both scales. D15c: one of 24 checks has a CI below 0 (LTX skip 3, motion smoothness, -0.0001). D15d: uniform 2 lowers DINO consistency (-0.008 $[-0.019, -0.001]$), first4/rest2 does not (-0.000 $[-0.004, +0.003]$). Outside D15c, LTX INT8 on the B200 raises the aesthetic score ($+0.136$) and lowers DINO consistency (-0.006), where the A800 runs showed neither (their one change was motion, -0.7). FBCache skips the same calls as on the A800 (AnyFlow call 2 in 57/66, rCM and LTX call 3 in 66/66).

K. Two newer third-party students (D16)

D16 asks whether StudentProbe's sweep carries over to 2025 families. It was registered before any run of these models, and it is a partial test: 8 of 12 predictions hold and neither model passes all of its own, so neither counts as a held-out family. Both students are third-party distillations by LightX2V, with no paper or training details on their model cards. `wan22_4s` is Wan2.2 T2V A14B with the `Wan2.2-Lightning` 4-step LoRAs (folder `250928`) fused into both experts: Euler, shift 5, CFG 1 and $t = 1000, 937.5, 833.3, 625$, so calls 0–1 use the high-noise expert and calls 2–3 the low-noise one. `hy15_4s` is HunyuanVideo-1.5 480p T2V with LightX2V's 4-step DiT, at 480×848×81. Its model card gives shift 9.0 and LightX2V's config file 5.0; both gave valid videos in the validation, and the registered rule kept 9.0. The diffusers pipeline casts its timesteps to bf16, so the DiT sees $t = 1000, 964, 900, 752$. The HunyuanVideo-1.5 weights carry the tencent-hunyuan-community licence. Alice v1 (arXiv 2605.08115) was considered and excluded, because its paper describes a dense DiT while its repository holds two experts and needs its own loader. All quality and timing runs are on B200s; no teacher was run.

Wan2.2's VAE share is 18.3%, above the 15% threshold, which came from an A800 (D16a fails; HunyuanVideo 7.1%, reported only). One-call-fewer predictions are within 0.2% and 0.1% (D16b). Tuning LPIPS is 0.423, 0.272 and 0.164 for Wan2.2 and 0.373, 0.186 and 0.202 for HunyuanVideo at calls 1–3 (D16c holds on both), so reuse across Wan2.2's expert switch at call 2 does not change the order seen on AnyFlow. The sweep's pick holds out on Wan2.2 and is a near-tie on HunyuanVideo, where it picks call 2 but call 3 is lowest held out (-0.005 $[-0.016, +0.005]$; D16d). On Wan2.2 the last call lowers aesthetics (-0.080 $[-0.133, -0.029]$; D16e fails), while on HunyuanVideo the chosen call does not ($+0.025$ $[-0.028, +0.091]$). The quality check, which was not part of the registration, would then move Wan2.2 to call 2, as on AnyFlow. FBCache skips Wan2.2's call 3 in 8/8 tuning videos and call 2 in none (mean indicator 0.73, 0.82, 0.35; D16f). On HunyuanVideo it skips the costliest call in 8/8 and 66/66 videos (indicator 0.036, 0.067, 0.192), $+0.153$ $[+0.140, +0.166]$ LPIPS above StudentProbe's call (D16g fails). On Wan2.2, sparse attention

	AnyFlow				rCM			
Setting	LPIPS	− one call fewer	speedup	pred.	LPIPS	− one call fewer	speedup	pred.
one call fewer	.081		1.174×		.193		1.151×	
two calls fewer	.255	+.174 [+.160, +.188]	1.429×		.256	+.063 [+.057, +.068]	1.359×	
d=0.1, 1 dense	.214	+.133 [+.122, +.144]	1.127×	1.119×	.394	+.201 [+.189, +.212]	1.156×	1.154×
d=0.2, 1 dense	.162	+.081 [+.074, +.089]	1.089×	1.091×	.324	+.131 [+.118, +.144]	1.120×	1.118×
d=0.3, 1 dense	.132	+.051 [+.045, +.057]	1.062×	1.068×	.286	+.093 [+.081, +.104]	1.088×	1.087×
d=0.5, 1 dense	.084	+.003 [−.001, +.007]	1.014×	1.022×	.194	+.001 [−.013, +.014]	1.027×	1.027×
d=0.1, 2 dense	.101	+.020 [+.014, +.027]	1.074×	1.076×	.232	+.039 [+.031, +.048]	1.097×	1.097×
d=0.2, 2 dense	.065	−.016 [−.019, −.012]	1.056×	1.059×	.203	+.009 [−.004, +.023]	1.077×	1.075×
d=0.3, 2 dense	.049	−.032 [−.036, −.028]	1.041×	1.044×	.154	−.040 [−.052, −.026]	1.057×	1.056×
d=0.5, 2 dense	.026	−.056 [−.062, −.050]	1.007×	1.014×	.085	−.109 [−.119, −.098]	1.016×	1.018×

Table 12. Sparse attention at several densities (D14). LPIPS to each model's own base on the B200 (33 dev prompts, seed 0), the paired difference to one call fewer with its bootstrap 95% CI, and the measured and probe-predicted end-to-end speedups on an H100 (2 warm-up and 5 timed runs). One call fewer is the grid-matched drop of call 2; two calls fewer is each model's native 2-step setting. `gmsa/analysis/d14_density.py` regenerates the table.

Model	Run	LPIPS	Δ imaging q.	Δ motion smooth.	Δ flickering	Δ aesthetic	Δ DINO
AnyFlow	base (mean)		0.6992	0.9904	0.9816	6.7397	0.9683
	skip 1	0.204	−0.002 [−0.009, +0.005]	**−0.0006 [−0.0008, −0.0004]**	+0.0003 [−0.0001, +0.0007]	**−0.056 [−0.105, −0.001]**	**−0.005 [−0.010, −0.001]**
	skip 2	0.075	+0.001 [−0.003, +0.004]	−0.0000 [−0.0002, +0.0002]	**+0.0009 [+0.0003, +0.0014]**	−0.015 [−0.051, +0.020]	−0.001 [−0.004, +0.001]
	skip 3	0.073	−0.002 [−0.006, +0.002]	**−0.0011 [−0.0015, −0.0006]**	−0.0007 [−0.0016, +0.0003]	**−0.145 [−0.205, −0.089]**	**−0.010 [−0.012, −0.009]**
	FBCache	0.069	+0.001 [−0.003, +0.004]	−0.0001 [−0.0003, +0.0001]	**+0.0005 [+0.0000, +0.0009]**	**−0.055 [−0.097, −0.019]**	−0.001 [−0.004, +0.000]
	compiled	0.028	+0.000 [−0.001, +0.001]	−0.0000 [−0.0000, +0.0000]	−0.0000 [−0.0000, +0.0000]	−0.000 [−0.014, +0.014]	+0.001 [−0.000, +0.002]
rCM	base (mean)		0.6698	0.9838	0.9663	6.2532	0.9135
	skip 1	0.326	−0.004 [−0.010, +0.002]	+0.0004 [−0.0000, +0.0009]	**+0.0031 [+0.0025, +0.0038]**	+0.034 [−0.027, +0.097]	+0.001 [−0.008, +0.008]
	skip 2	0.245	**+0.007 [+0.001, +0.012]**	−0.0004 [−0.0008, +0.0000]	+0.0001 [−0.0007, +0.0008]	−0.043 [−0.106, +0.018]	−0.004 [−0.009, +0.001]
	skip 3	0.539	**−0.129 [−0.151, −0.108]**	**−0.0567 [−0.0590, −0.0543]**	**−0.0468 [−0.0486, −0.0450]**	**−1.651 [−1.875, −1.452]**	**−0.204 [−0.227, −0.180]**
	FBCache	0.267	**−0.031 [−0.048, −0.017]**	**−0.0115 [−0.0121, −0.0109]**	**−0.0086 [−0.0097, −0.0077]**	**−0.465 [−0.592, −0.344]**	**−0.081 [−0.099, −0.065]**
	compiled	0.130	−0.002 [−0.004, +0.000]	−0.0000 [−0.0003, +0.0002]	**+0.0002 [+0.0000, +0.0003]**	−0.024 [−0.065, +0.016]	+0.002 [−0.001, +0.005]
LTX	base (mean)		0.6545	0.9958	0.9896	5.8054	0.9518
	skip 1	0.263	+0.004 [−0.007, +0.016]	−0.0001 [−0.0004, +0.0001]	−0.0001 [−0.0008, +0.0005]	+0.020 [−0.054, +0.090]	−0.003 [−0.009, +0.002]
	skip 2	0.118	−0.002 [−0.007, +0.004]	+0.0000 [−0.0000, +0.0001]	+0.0001 [−0.0000, +0.0002]	−0.007 [−0.055, +0.041]	−0.002 [−0.006, +0.000]
	skip 3	0.115	+0.001 [−0.005, +0.008]	**−0.0001 [−0.0002, −0.0000]**	**+0.0002 [+0.0000, +0.0003]**	**+0.056 [+0.009, +0.103]**	+0.000 [−0.002, +0.002]
	skip 4	0.073	**+0.005 [+0.000, +0.011]**	**+0.0001 [+0.0001, +0.0002]**	**+0.0007 [+0.0005, +0.0010]**	**+0.044 [+0.004, +0.085]**	−0.002 [−0.006, +0.001]
	skip 5	0.022	+0.002 [−0.000, +0.005]	+0.0000 [−0.0000, +0.0001]	**+0.0005 [+0.0003, +0.0006]**	**+0.079 [+0.052, +0.107]**	**+0.002 [+0.001, +0.003]**
	skip 6	0.009	**+0.002 [+0.001, +0.004]**	+0.0000 [−0.0001, +0.0001]	**+0.0005 [+0.0004, +0.0007]**	**+0.058 [+0.040, +0.076]**	**+0.001 [+0.000, +0.002]**
	FBCache	0.115	+0.001 [−0.005, +0.007]	**−0.0001 [−0.0002, −0.0000]**	**+0.0002 [+0.0000, +0.0003]**	**+0.055 [+0.008, +0.101]**	+0.000 [−0.002, +0.002]
	INT8	0.461	−0.015 [−0.033, +0.004]	+0.0003 [−0.0001, +0.0008]	−0.0000 [−0.0018, +0.0016]	**+0.136 [+0.022, +0.249]**	**−0.006 [−0.011, −0.001]**
	compiled	0.059	−0.000 [−0.005, +0.004]	+0.0000 [−0.0000, +0.0000]	+0.0000 [−0.0001, +0.0001]	+0.011 [−0.034, +0.059]	−0.002 [−0.004, +0.001]
Self-Forcing	base (mean)		0.7113	0.9874	0.9740	6.4864	0.9415
	uniform 3	0.333	−0.003 [−0.010, +0.004]	+0.0003 [−0.0002, +0.0008]	**+0.0015 [+0.0007, +0.0024]**	−0.058 [−0.118, +0.004]	+0.002 [−0.002, +0.005]
	uniform 2	0.466	−0.005 [−0.018, +0.006]	−0.0006 [−0.0013, +0.0001]	+0.0009 [−0.0004, +0.0021]	+0.019 [−0.082, +0.123]	**−0.008 [−0.019, −0.001]**
	first4/rest2	0.186	−0.002 [−0.005, +0.002]	+0.0002 [−0.0003, +0.0008]	**+0.0015 [+0.0005, +0.0026]**	−0.007 [−0.035, +0.021]	−0.000 [−0.004, +0.003]
	first4/rest3	0.152	−0.001 [−0.003, +0.002]	−0.0001 [−0.0005, +0.0005]	+0.0006 [−0.0001, +0.0015]	−0.007 [−0.032, +0.014]	+0.001 [−0.001, +0.004]
rCM-14B	base (mean)		0.6931	0.9868	0.9691	6.4635	0.9432
	skip 2	0.169	−0.001 [−0.005, +0.004]	**−0.0009 [−0.0013, −0.0006]**	−0.0003 [−0.0008, +0.0002]	**−0.067 [−0.127, −0.014]**	**−0.003 [−0.006, −0.001]**
	skip 3	0.451	**−0.129 [−0.149, −0.109]**	**−0.0514 [−0.0543, −0.0484]**	**−0.0439 [−0.0463, −0.0418]**	**−1.452 [−1.696, −1.228]**	**−0.267 [−0.290, −0.245]**
	one call fewer	0.161	**+0.006 [+0.002, +0.010]**	**+0.0007 [+0.0004, +0.0010]**	**+0.0028 [+0.0021, +0.0034]**	+0.005 [−0.041, +0.053]	+0.002 [−0.001, +0.004]
	sparse d=0.2	0.337	−0.001 [−0.010, +0.007]	**+0.0011 [+0.0005, +0.0018]**	−0.0004 [−0.0015, +0.0005]	**−0.095 [−0.168, −0.023]**	**−0.024 [−0.030, −0.018]**
	compiled	0.130	−0.001 [−0.003, +0.002]	−0.0001 [−0.0003, +0.0001]	+0.0001 [−0.0001, +0.0003]	−0.022 [−0.050, +0.008]	−0.000 [−0.001, +0.001]

Table 13. Every D15 run on the B200: LPIPS to the model's base, and paired differences with 95% CIs (bold: CI excludes 0). Base rows give the means on which the differences are taken. VBench dimensions are on a 0–1 scale.

drifts more than one call fewer (+0.089 [+0.074, +0.104]; D16h) and is also slower (1.024× vs 1.242×; Amdahl, not registered: 1.022×), so the speeds are not equal. End-to-end time is 8.58 s for Wan2.2 and 38.24 s for HunyuanVideo. HunyuanVideo's self-attention takes 82.9% of it, and one call takes 147 ms in the model against 6.7 ms in the dense kernel benchmark at the same shape, which has no padding mask; we did not find the cause. Wan2.2 agrees with its benchmark (14.5 vs 15.2 ms).

Model	Run	LPIPS	Δ aesthetic	Δ DINO	Δ imaging q.	Δ motion smooth.	Δ flickering
Wan2.2 A14B	base (mean)		6.6295	0.9453	0.6897	0.9893	0.9771
	skip 1	0.395	**−0.088 [−0.155, −0.030]**	+0.001 [−0.003, +0.005]	−0.0040 [−0.0152, +0.0082]	+0.0001 [−0.0005, +0.0008]	**+0.0027 [+0.0017, +0.0036]**
	skip 2	0.275	−0.020 [−0.081, +0.046]	+0.001 [−0.002, +0.003]	−0.0011 [−0.0093, +0.0058]	**−0.0006 [−0.0008, −0.0003]**	**−0.0006 [−0.0011, −0.0002]**
	skip 3	0.163	**−0.080 [−0.133, −0.029]**	**−0.007 [−0.010, −0.005]**	+0.0014 [−0.0040, +0.0066]	**−0.0008 [−0.0011, −0.0005]**	−0.0005 [−0.0012, +0.0002]
	FBCache	0.172	**−0.116 [−0.178, −0.059]**	**−0.009 [−0.012, −0.006]**	+0.0023 [−0.0031, +0.0081]	**−0.0015 [−0.0019, −0.0011]**	**−0.0012 [−0.0020, −0.0003]**
	sparse d=0.2	0.369	−0.005 [−0.092, +0.082]	**−0.013 [−0.020, −0.006]**	**+0.0168 [+0.0072, +0.0263]**	**+0.0017 [+0.0009, +0.0025]**	+0.0006 [−0.0000, +0.0011]
	one call fewer	0.280	−0.022 [−0.078, +0.036]	+0.003 [+0.000, +0.006]	−0.0032 [−0.0109, +0.0035]	−0.0002 [−0.0006, +0.0001]	**+0.0007 [+0.0001, +0.0013]**
HunyuanVideo-1.5	base (mean)		6.0382	0.8876	0.6541	0.9932	0.9605
	skip 1	0.311	+0.070 [−0.018, +0.187]	**+0.004 [+0.001, +0.007]**	−0.0017 [−0.0091, +0.0055]	**+0.0001 [+0.0001, +0.0002]**	**+0.0044 [+0.0035, +0.0053]**
	skip 2	0.158	+0.025 [−0.028, +0.091]	+0.002 [−0.002, +0.006]	−0.0035 [−0.0076, +0.0003]	**+0.0001 [+0.0001, +0.0002]**	**+0.0037 [+0.0031, +0.0044]**
	skip 3	0.153	−0.056 [−0.132, +0.023]	**−0.012 [−0.018, −0.006]**	**−0.0096 [−0.0176, −0.0026]**	**−0.0009 [−0.0011, −0.0006]**	**+0.0055 [+0.0046, +0.0064]**
	FBCache	0.311	+0.069 [−0.020, +0.187]	**+0.004 [+0.001, +0.007]**	−0.0015 [−0.0086, +0.0057]	**+0.0002 [+0.0001, +0.0003]**	**+0.0044 [+0.0036, +0.0053]**

Table 14. D16 held-out runs on the B200 (33 prompts, seeds 0 and 1, 66 videos per run): LPIPS to the model's base, and paired differences with 95% CIs (bold: CI excludes 0). Base rows give the means. CLIP and motion differences are in `runs/summaries/d16_newfamilies.md`.

L. Frame sparsity, a lightweight decoder and stacks (D17)

D17 asks whether the grid's advantage extends to frame-interleaved sparsity, whether the decoder cap has a low-cost remedy, and whether accelerators on different axes compose. It was registered before any run. Stages T and H and all timing ran on one H100 NVL (94 GB) because the B200s were in use; this change was registered before those stages, so D17a and the planner use this GPU's own probe, while D17e's and D17h's thresholds, set for a B200, were kept. `fis` is our reimplementation of FIS-DiT from its paper, whose code is not listed: blocks 0, 38 and 39 and the last step run in full, with its anchors and interpolation, text tokens whole and both experts under the same rule, in PyTorch SDPA (they use FlashAttention on an H20). Read literally, its eq. 7–10 and 15 interpolate a block-level output, which gave coloured speckle on every frame at $n > 1$. The paper also defines the operator on "sub-modules" behind a "full-frame module interface"; with Wan's residual connections a skipped frame then keeps its own features and adds an interpolated residual, which gives clean videos. A rule committed before the second run took the first reading with valid videos, without consulting any score, so every `fis` value here uses the residual reading. Differences from their reported numbers concern our port only. At $n = 1$ the output is byte-identical to the base. The lightweight decoder is `lightx2v/Autoencoders` 043793f3, `lighttaew2_1.safetensors`, a trained replacement used as released (parallel decode, bf16). The stacks combine skip $k = 2$, compilation and the lightweight decoder, with INT8 on Wan2.2 A14B, where it helps.

The probe predicts `fis` at 1.708× and 1.898× against 1.670× and 1.864× measured at $n = 3$ and 5 (D17a: 2.3%, 1.8%). At $n = 3$, `fis` drifts more than two calls, $+0.087$ $[+0.070, +0.103]$ at 1.670× vs 1.721× (D17b); at $n = 5$ the difference is $+0.124$ $[+0.103, +0.143]$ at 1.864×. None of the five quality scores of `fis` $n = 3$ has a CI below 0 (D17c), but motion falls by 1.878 $[1.179, 2.654]$ on a base of 4.231, and in stage V small moving objects were less visible. We predicted that the two-call grid would lower aesthetics or DINO, and it does not (D17d fails): the grid 1000+500, the tuning sweep's choice, gives $+0.018$ $[-0.066, +0.104]$ and $+0.000$ $[-0.003, +0.003]$. By the registered rule, this grid shows no significant decline in aesthetics or DINO on this student, although motion falls by 0.579 $[0.175, 1.055]$, and the other two grids lower aesthetics. The lightweight decoder gives 1.495× on AnyFlow, 1.629× on rCM and 1.176× on Wan2.2 A14B (D17e). We predicted that it would lower imaging quality, and on all three students it does not (D17f fails); the cost shows in DINO on all three, in rCM's aesthetics, and in peak memory, from 17.9 to 30.6 GB at 1.3B and from 73.4 to 81.7 GB on Wan2.2 A14B. On one H100 it is faster than one call fewer on AnyFlow (1.495× vs 1.180×) and slower on Wan2.2 A14B (1.176× vs 1.261×). The stacks fall within 0.1% (1.903×, Wan2.2 A14B) and 4.8% (2.474×, AnyFlow) of the prediction (D17g). At 720p the VAE takes 10.9% and self-attention 58.5%, and `fis` $n = 3$ reaches 1.859× (D17h, timing only). H100 bases cannot be compared with the B200 hashes of the earlier account, so the AnyFlow and rCM held-out bases were regenerated on a B200; all 132 videos are byte-identical to those hashes.

M. Allocating compute across calls (D18)

D18 tested a training-free allocator with predicted speed and drift. Its registered rule withdrew the method claim if D18c or D18d failed, and both did. Related allocators work on other settings: ALLOT (arXiv 2610.29441) allocates attention density over layer–step cells on 50-step teachers, PASA (2604.12219) shifts attention budget over steps, DFSAttn (2605.23445) skips sparsity at the first step of a 4-step student, and LeMiCa (2511.00090), DPCache (2602.22654) and SenCache (2602.24208) plan or adapt caches. D18 allocated across axes on few-step students, compared against dropping steps and calibrated in minutes; we found no such combination, in a search that was not exhaustive. A plan

Model	Run	Speed	LPIPS	Δ aesthetic	Δ DINO	Δ CLIP	Δ motion	Δ imaging q.	Δ motion smooth.	Δ flickering
Wan2.2 A14B	fis 3,3,3,1 (dense last)	1.670×	**+0.490 [+0.463, +0.517]**	−0.061 [−0.143, +0.019]	**+0.021 [+0.011, +0.032]**	+0.069 [−0.229, +0.338]	**−1.878 [−2.654, −1.179]**	+0.0041 [−0.0081, +0.0164]	**+0.0028 [+0.0017, +0.0040]**	**+0.0085 [+0.0062, +0.0107]**
	fis 1,3,3,3 (dense first)	1.663×	**+0.428 [+0.404, +0.451]**	**−0.143 [−0.254, −0.049]**	−0.006 [−0.014, +0.002]	**+0.336 [+0.123, +0.564]**	**−2.048 [−3.738, −0.718]**	**−0.0304 [−0.0505, −0.0117]**	−0.0000 [−0.0012, +0.0012]	**+0.0051 [+0.0023, +0.0080]**
	fis 1,3,3,1	1.371×	**+0.380 [+0.356, +0.403]**	−0.034 [−0.113, +0.039]	**+0.011 [+0.005, +0.018]**	+0.158 [−0.040, +0.344]	**−1.426 [−2.284, −0.727]**	+0.0011 [−0.0086, +0.0113]	**+0.0023 [+0.0012, +0.0034]**	**+0.0069 [+0.0051, +0.0087]**
	lightweight decoder	1.176×	**+0.036 [+0.030, +0.043]**	−0.009 [−0.045, +0.026]	**−0.004 [−0.006, −0.002]**	−0.081 [−0.216, +0.050]	**−0.062 [−0.131, −0.010]**	**+0.0046 [+0.0023, +0.0067]**	−0.0001 [−0.0006, +0.0005]	**+0.0023 [+0.0017, +0.0029]**
	skip 2 + decoder + compilation	1.903×	**+0.400 [+0.381, +0.421]**	−0.041 [−0.138, +0.066]	−0.003 [−0.008, +0.001]	−0.148 [−0.464, +0.140]	+0.230 [−0.068, +0.516]	+0.0053 [−0.0014, +0.0118]	**−0.0009 [−0.0017, −0.0001]**	**+0.0013 [+0.0002, +0.0024]**
	top plan + decoder + compilation	3.206×	**+0.507 [+0.482, +0.532]**	−0.065 [−0.170, +0.044]	**+0.019 [+0.009, +0.031]**	−0.006 [−0.392, +0.350]	**−2.318 [−3.314, −1.432]**	+0.0092 [−0.0006, +0.0201]	**+0.0033 [+0.0022, +0.0044]**	**+0.0106 [+0.0081, +0.0131]**
	fis 5,5,5,1	1.864×	**+0.526 [+0.501, +0.552]**	−0.076 [−0.175, +0.018]	**+0.028 [+0.016, +0.042]**	+0.000 [−0.337, +0.311]	**−2.938 [−4.167, −1.908]**	+0.0084 [−0.0048, +0.0234]	**+0.0036 [+0.0023, +0.0049]**	**+0.0110 [+0.0079, +0.0140]**
	two calls, 1000+750	1.708×	**+0.457 [+0.435, +0.481]**	**−0.132 [−0.238, −0.033]**	**−0.014 [−0.020, −0.009]**	**+1.087 [+0.644, +1.543]**	**−0.532 [−1.135, −0.030]**	**−0.0246 [−0.0331, −0.0158]**	**−0.0026 [−0.0036, −0.0015]**	+0.0008 [−0.0007, +0.0024]
	two calls, 1000+500	1.721×	**+0.403 [+0.382, +0.424]**	+0.018 [−0.066, +0.104]	+0.000 [−0.003, +0.003]	**+0.574 [+0.274, +0.920]**	**−0.579 [−1.055, −0.175]**	−0.0078 [−0.0204, +0.0035]	−0.0000 [−0.0008, +0.0008]	**+0.0037 [+0.0026, +0.0049]**
	two calls, 1000+250	1.728×	**+0.447 [+0.427, +0.469]**	**−0.175 [−0.262, −0.088]**	−0.003 [−0.008, +0.003]	**+0.602 [+0.278, +0.920]**	−0.378 [−0.986, +0.153]	**−0.0351 [−0.0564, −0.0143]**	+0.0003 [−0.0006, +0.0013]	**+0.0055 [+0.0037, +0.0072]**
	plan 1,0,0,1	1.710×	**+0.432 [+0.411, +0.453]**	**−0.187 [−0.280, −0.094]**	+0.001 [−0.004, +0.006]	**+0.383 [+0.061, +0.715]**	−0.540 [−1.187, +0.020]	**−0.0173 [−0.0338, −0.0014]**	+0.0004 [−0.0005, +0.0012]	**+0.0039 [+0.0026, +0.0052]**
	plan 3,5,0,1	1.973×	**+0.505 [+0.479, +0.530]**	−0.070 [−0.162, +0.026]	**+0.020 [+0.010, +0.032]**	−0.017 [−0.397, +0.318]	**−2.289 [−3.307, −1.373]**	+0.0046 [−0.0053, +0.0151]	**+0.0026 [+0.0015, +0.0037]**	**+0.0081 [+0.0057, +0.0106]**
AnyFlow	fis 3,3,3,1 (dense last)	1.383×	**+0.523 [+0.496, +0.550]**	**−0.224 [−0.314, −0.126]**	−0.004 [−0.010, +0.002]	−0.059 [−0.402, +0.288]	**−2.016 [−3.334, −0.843]**	+0.0097 [−0.0027, +0.0212]	+0.0009 [−0.0002, +0.0021]	**+0.0047 [+0.0016, +0.0080]**
	fis 1,3,3,3 (dense first)	1.386×	**+0.283 [+0.255, +0.311]**	**−0.220 [−0.294, −0.147]**	**−0.010 [−0.016, −0.004]**	+0.168 [−0.201, +0.500]	**−1.349 [−2.569, −0.197]**	−0.0048 [−0.0146, +0.0043]	**−0.0014 [−0.0021, −0.0007]**	+0.0018 [−0.0007, +0.0042]
	fis 1,3,3,1	1.209×	**+0.256 [+0.229, +0.282]**	**−0.146 [−0.209, −0.089]**	−0.002 [−0.007, +0.002]	+0.157 [−0.168, +0.427]	**−2.003 [−3.333, −0.880]**	+0.0035 [−0.0030, +0.0102]	−0.0003 [−0.0009, +0.0002]	**+0.0039 [+0.0022, +0.0056]**
	lightweight decoder	1.495×	**+0.028 [+0.023, +0.032]**	−0.038 [−0.077, +0.001]	**−0.002 [−0.003, −0.001]**	**−0.126 [−0.229, −0.023]**	**−0.078 [−0.168, −0.009]**	**+0.0029 [+0.0012, +0.0046]**	−0.0001 [−0.0006, +0.0003]	**+0.0016 [+0.0011, +0.0023]**
	skip 2 + decoder + compilation	2.474×	**+0.097 [+0.086, +0.109]**	−0.046 [−0.106, +0.017]	−0.002 [−0.004, +0.000]	−0.168 [−0.393, +0.063]	**−0.308 [−0.785, −0.034]**	+0.0021 [−0.0016, +0.0061]	+0.0003 [−0.0002, +0.0008]	**+0.0030 [+0.0020, +0.0041]**
	top plan + decoder + compilation	4.332×	**+0.583 [+0.555, +0.611]**	**−0.544 [−0.728, −0.379]**	**−0.038 [−0.050, −0.027]**	−0.511 [−1.139, +0.085]	**−0.873 [−1.796, −0.001]**	−0.0211 [−0.0446, +0.0017]	**−0.0016 [−0.0029, −0.0003]**	+0.0010 [−0.0018, +0.0037]
	two calls	1.444×	**+0.250 [+0.231, +0.268]**	**−0.088 [−0.134, −0.045]**	**−0.006 [−0.013, −0.002]**	+0.100 [−0.178, +0.362]	**−0.534 [−1.156, −0.166]**	+0.0008 [−0.0054, +0.0064]	−0.0001 [−0.0004, +0.0002]	**+0.0010 [+0.0002, +0.0017]**
	plan 1,0,0,1	1.437×	**+0.323 [+0.298, +0.348]**	**−0.329 [−0.415, −0.250]**	**−0.014 [−0.020, −0.009]**	−0.166 [−0.723, +0.343]	−0.362 [−1.197, +0.166]	**−0.0222 [−0.0432, −0.0037]**	**−0.0013 [−0.0017, −0.0009]**	+0.0005 [−0.0005, +0.0016]
	plan 3,0,0,1	1.637×	**+0.579 [+0.550, +0.606]**	**−0.596 [−0.785, −0.422]**	**−0.038 [−0.050, −0.027]**	−0.548 [−1.249, +0.102]	−0.730 [−1.636, +0.097]	**−0.0256 [−0.0495, −0.0030]**	**−0.0020 [−0.0032, −0.0007]**	−0.0022 [−0.0048, +0.0005]
rCM	fis 3,3,3,1 (dense last)	1.351×	**+0.711 [+0.688, +0.732]**	−0.107 [−0.276, +0.066]	**+0.028 [+0.013, +0.044]**	+0.331 [−0.138, +0.779]	**−4.694 [−6.872, −2.937]**	**−0.0334 [−0.0541, −0.0127]**	**+0.0068 [+0.0049, +0.0087]**	**+0.0161 [+0.0127, +0.0194]**
	fis 1,3,3,3 (dense first)	1.363×	**+0.503 [+0.473, +0.531]**	**−0.194 [−0.302, −0.083]**	**+0.015 [+0.001, +0.028]**	+0.271 [−0.147, +0.691]	**−4.806 [−7.014, −2.970]**	**−0.0486 [−0.0662, −0.0316]**	**+0.0055 [+0.0041, +0.0070]**	**+0.0147 [+0.0115, +0.0178]**
	fis 1,3,3,1	1.211×	**+0.485 [+0.456, +0.513]**	−0.075 [−0.157, +0.013]	**+0.023 [+0.010, +0.036]**	+0.166 [−0.200, +0.519]	**−4.330 [−6.375, −2.654]**	**−0.0227 [−0.0338, −0.0124]**	**+0.0046 [+0.0036, +0.0057]**	**+0.0130 [+0.0104, +0.0155]**
	lightweight decoder	1.629×	**+0.037 [+0.032, +0.042]**	**−0.047 [−0.081, −0.015]**	**−0.007 [−0.011, −0.004]**	−0.062 [−0.221, +0.078]	−0.012 [−0.126, +0.115]	+0.0004 [−0.0013, +0.0020]	−0.0004 [−0.0007, −0.0000]	**+0.0020 [+0.0016, +0.0023]**
	two calls	1.377×	**+0.255 [+0.232, +0.278]**	+0.042 [−0.033, +0.115]	−0.002 [−0.010, +0.004]	**+0.224 [+0.006, +0.458]**	−0.218 [−0.601, +0.206]	**+0.0075 [+0.0022, +0.0131]**	**+0.0010 [+0.0005, +0.0015]**	**+0.0037 [+0.0026, +0.0048]**
	plan 1,0,0,1	1.372×	**+0.428 [+0.409, +0.445]**	**−0.125 [−0.232, −0.022]**	−0.008 [−0.020, +0.002]	**+0.449 [+0.073, +0.866]**	**−1.571 [−3.047, −0.556]**	+0.0098 [−0.0023, +0.0215]	+0.0009 [+0.0000, +0.0018]	**+0.0028 [+0.0011, +0.0045]**
	plan 5,0,0,1	1.609×	**+0.723 [+0.706, +0.738]**	−0.153 [−0.361, +0.065]	−0.000 [−0.016, +0.016]	+0.402 [−0.266, +1.074]	**−5.048 [−7.254, −3.224]**	−0.0067 [−0.0265, +0.0129]	**+0.0051 [+0.0033, +0.0069]**	**+0.0134 [+0.0099, +0.0169]**

Table 15. Held-out runs of D17 and D18 on one H100 NVL (33 prompts, seeds 0 and 1, 66 videos per run): end-to-end speedup, LPIPS to the model's base, paired differences with 95% CIs (bold: CI excludes 0). A plan lists each of the four calls: 1 full, $n \geq 2$ frame sparsity at stride n, 0 reuse. `fis` is our reimplementation, residual reading. Generated from `runs/summaries/d17_gap.md`.

gives each of four calls one option: full, frame sparsity at stride 3 or 5, or reuse. The planner measures 11 single-call options on 8 tuning prompts, predicts a plan's drift as the quadrature sum of its single-call costs and its speed from the probe, and picks the lowest predicted drift at each budget. Reuse is excluded at the first call. An amendment before stage T also keeps the last call full (47 candidate plans instead of 191), because plans with a sparse last call showed visible block artifacts on Wan2.2 A14B on one tuning prompt, by eye. The planner covers Wan2.2 A14B, AnyFlow and rCM; other families have no `fis` adapter here. It chose 1,3,3,1, 1,0,0,1 and 3,5,0,1 on Wan2.2 A14B at 1.3×, 1.6× and 2.0×, and 1,0,0,1 and 3,0,0,1 on AnyFlow and 1,0,0,1 and 5,0,0,1 on rCM at 1.3× and 1.6×.

On Wan2.2 A14B a per-video oracle over the three two-call grids gains 2.6% (0.403 to 0.392; wins 12, 52, 2; D18a). On existing data, the same count over one-call skips gains 7% on AnyFlow, 4% on rCM, 0% on rCM-14B (positions 2 and 3), 0% on LTX, 0% on Wan2.2 A14B and 8% on HunyuanVideo-1.5; one position wins in all 66 videos on LTX and Wan2.2 A14B (exploratory). At speeds within 1%, a dense first call drifts less than a dense last one on all three students (D18b): -0.062 $[-0.082, -0.042]$, -0.240 $[-0.272, -0.211]$, -0.208 $[-0.229, -0.188]$. Quality goes the other way: imaging quality -0.0345 $[-0.0517, -0.0187]$, -0.0146 $[-0.0253, -0.0034]$, -0.0152 $[-0.0375, +0.0050]$; motion smoothness and flickering fall in 5 of 6 CIs and DINO in 2 of 3. Quadrature composes single-call costs well enough only on AnyFlow (D18c: median error 11.6%, Spearman 0.85); on Wan2.2 A14B (27.3%, 0.94) and rCM (24.7%, 0.70) it misses the threshold. The plain sum errs by 79–125%, and quadrature overpredicts 29 of 30 plans, so it ranks better than it sizes. The planner's held-out predictions are close (D18d-1: speed within 2.1%, 1.7%, 0.8%; median drift error 16.2%, 13.8%, 8.0%), but its 1.6× plan drifts less than the grid at equal latency on no student (D18d-2). Against dense-last `fis` $n = 3$ it differs by -0.058 $[-0.074, -0.040]$, $+0.056$ $[+0.046, +0.066]$ and $+0.012$ $[+0.002, +0.022]$, and against two calls by $+0.029$ $[+0.021, +0.038]$, $+0.329$ $[+0.303, +0.353]$ and $+0.467$ $[+0.446, +0.489]$. Speeds are within 2.4% on Wan2.2 A14B, while on AnyFlow and rCM the plan is 13–19% faster than both, so there the comparison is not at equal speed. With the decoder and compilation, the planner's top budget reaches 3.206× on Wan2.2 A14B and 4.332× on AnyFlow (D18e; on AnyFlow its top feasible budget is 1.6×), at a quality cost: on Wan2.2 A14B LPIPS is 0.507 and motion falls by 2.318 $[1.432, 3.314]$ on a base of 4.231, with none of the five scores falling; on AnyFlow LPIPS is 0.583 and aesthetics fall by 0.544 $[0.379, 0.728]$. Chosen after these results (exploratory): with the same two full calls and speeds within 1%, reuse at calls 2 and 3 drifts more than the student's own two-call grid by $+0.029$ $[+0.021, +0.038]$, $+0.073$ $[+0.059, +0.086]$ and $+0.173$ $[+0.157, +0.189]$, and scores -0.206, -0.240 and -0.167 in aesthetics (all CIs below 0).

Among frame-sparse plans, keeping the first call dense drifts less than keeping the last (-0.062, -0.240, -0.208 on three students), as amplification suggests, but lowers VBench's scores in 7 of 9 comparisons, so drift and quality favor different calls.

N. Two calls, a lightweight decoder and compilation (D19)

D19 is the StudentProbe-guided baseline of Sec. 6 of the paper: it spends the calls where the grid says and removes the costs outside the DiT, using released components only. It was registered after all of D17 and D18, so

on Wan2.2 A14B, AnyFlow and rCM it tests only whether known costs compose; rCM-14B is the one student without an earlier run of this kind and the only held-out test. Both H100 NVL cards were taken by another project just after registration, so before any run D19 moved to one PRO6000 BW type (96 GB), where its bases, two-call runs and lightweight-decoder runs were regenerated. Every pair and speedup stays within that type; scoring and VBench ran on 4090s for all runs. The combination is the two-call grid (on Wan2.2 A14B 1000+500, the H100 sweep's choice; on the others `--steps 2`), the lightweight decoder and `torch.compile` of every block. It uses no INT8, so the same combination applies to every student; INT8 is reported separately on Wan2.2 A14B.

Model order: Wan2.2 A14B, AnyFlow, rCM, rCM-14B. Speeds predicted from separately timed parts, 2.670×, 3.462×, 3.325× and 2.503×, are within 0.6%, 0.3%, 1.0% and 0.9% of the measurements (D19a), which reach 2.686×, 3.451×, 3.360× and 2.525× (D19b; thresholds 2.5× and, at 14B, 2.0×). LPIPS stays within 2.6%, 2.4%, 8.2% and 14.0% of the two-call grid's (D19c). D19d fails on Wan2.2 A14B and rCM-14B and holds on AnyFlow: on top of the grid, the decoder and compilation lower aesthetics by 0.049 $[0.004, 0.089]$ and 0.143 $[0.089, 0.205]$, and DINO by 0.002 and 0.003. rCM, which had no threshold, loses 0.061 $[0.008, 0.114]$ and 0.004. VBench's three scores do not fall on any. At 2.632× vs 2.686× on Wan2.2 A14B, fis $n = 3$ with the decoder and compilation drifts further ($+0.081$ $[+0.063, +0.098]$, D19e), lowers aesthetics and loses more motion, while scoring higher on DINO, smoothness and flickering. Against the base (Tab. 16), none of the five scores falls on Wan2.2 A14B or rCM; aesthetics and DINO fall on AnyFlow, about as much as for the grid alone; DINO falls on rCM-14B; motion falls on all four; and peak memory rises by 9–13 GB. This GPU reproduces the H100 values of the shared runs within 4% (two calls: LPIPS 0.403, 0.250, 0.266 vs 0.403, 0.250, 0.255; lightweight decoder: 0.036, 0.028, 0.033 vs 0.036, 0.028, 0.037), and re-timing the base at the end moved it by at most 0.6%. Related combinations exist on other models: Sol (arXiv 2606.23743) composes five other techniques (over 2×), and Flash-VAED (arXiv 2602.19161) replaces only the decoder (up to 36% end to end). We use the combination as an empirical speed reference on the evaluated students.

O. Which selector closes the sparse gap (D20)

D20 tests our earlier account, that sparse attention loses because selection loses attention mass; before it, no video had been generated with an oracle selection. It was registered before any run. Density 0.2, first call dense, held-out prompts, AnyFlow and rCM; everything, including base and one call fewer, was generated on 4090s. `oracle_block` keeps, per query block, the 20% of key blocks with the most true mass, computed from the exact softmax, and then runs the paper's kernel. `oracle_token` keeps, per query token, the 20% of keys with the most true mass; because a video took more than 2 minutes, a rule fixed in stage V limited it to 12 prompts at seed 0. `sap` is our port of SVG2's semantic-aware permutation (arXiv 2505.18875; Apache-2.0 code read at svg-project/Sparse-VideoGen). It runs k-means per head on queries and keys after RoPE, with 200 and 1000 centroids, 50 iterations seeded on the dense call and then 2 warm-started iterations, and keeps key clusters by size-weighted softmax over centroids up to cumulative p, with at least 10% kept and layer 0 dense; these values come from their 480p script. Attention is exact on the kept pairs. Their FlashInfer kernel and fast k-means are not reproduced, so the results describe the selection rule rather than SVG2's implementation. $p = 0.9$ is their value; the tuned p, chosen from $\{0.5, \ldots, 0.9\}$ by pair density closest to 0.2 on 8 tuning prompts without reading fidelity, is 0.8 on both models. None of these was timed. Each is given an ideal kernel on the H100 NVL's latency probe, with attention time proportional to density and selection free, which favors them.

An oracle choice of blocks still drifts further than one call fewer (D20a: $+0.072$ $[+0.065, +0.080]$, $+0.135$ $[+0.125, +0.145]$). An oracle choice of tokens was predicted to drift less and drifts much more (D20b: $+0.227$ $[+0.199, +0.254]$, $+0.375$ $[+0.348, +0.401]$); in stage V it already looked more saturated on one tuning prompt. Removing the tail of each query's attention may bias the output even though little mass is lost; we did not test this. Pooled selection drifts more than the oracle block choice on AnyFlow and less on rCM (D20c: $+0.010$ $[+0.007, +0.012]$, -0.024 $[-0.030, -0.019]$). `sap` at $p = 0.9$ (density 0.23) ties one call fewer on AnyFlow and drifts less on rCM ($+0.002$ $[-0.001, +0.004]$, -0.017 $[-0.026, -0.009]$), so D20d fails there; at the tuned p (density 0.19) it drifts more ($+0.026$ $[+0.022, +0.030]$, $+0.021$ $[+0.011, +0.029]$). At $p = 0.9$ it trails one call fewer on quality: aesthetics -0.055 $[-0.099, -0.012]$ and -0.055 $[-0.115, +0.004]$, DINO -0.003 $[-0.005, -0.001]$ and -0.008 $[-0.013, -0.003]$, and imaging quality on rCM -0.0078 $[-0.0137, -0.0020]$.

Numerical-path control (D31). `oracle_token` and `sap` compute attention in float32 in PyTorch, whereas base and one call fewer use the model's bf16 SDPA, and `pooled` and `oracle_block` our FlexAttention kernel. D31, registered before any run, reruns D20b's 12 prompts (seed 0) on RTX 3090s with every key kept in each path (`oracle_token` at density 1, `sap` with every cluster, FlexAttention with every block) and measures each selector against the full run of its own path. The full paths stay within 0.02 LPIPS of the base (0.004–0.005 on AnyFlow, 0.019 on rCM, against 0.071 and 0.191 for one call fewer; D31a holds in all six cases). Selection alone still drifts further than one

Model	Run	Speed	GB	LPIPS	Δ aesthetic	Δ DINO	Δ CLIP	Δ motion	Δ imaging q.	Δ motion smooth.	Δ flickering
Wan2.2 A14B	base (mean)	29.65 s	72.8		6.616	0.945	25.179	4.256	0.689	0.989	0.977
	two calls + decoder + compilation	2.686×	81.6	**+0.414 [+0.392, +0.435]**	−0.041 [−0.129, +0.039]	−0.002 [−0.005, +0.002]	**+0.602 [+0.221, +1.055]**	**−0.679 [−1.198, −0.213]**	−0.0045 [−0.0180, +0.0085]	**+0.0015 [+0.0007, +0.0024]**	**+0.0070 [+0.0055, +0.0086]**
	the same + INT8	3.103×	54.8	**+0.440 [+0.421, +0.459]**	−0.068 [−0.141, +0.005]	−0.002 [−0.006, +0.002]	**+0.445 [+0.103, +0.825]**	−0.369 [−0.921, +0.117]	−0.0054 [−0.0188, +0.0088]	**+0.0012 [+0.0004, +0.0020]**	**+0.0062 [+0.0047, +0.0078]**
	fis 3,3,3,1 + decoder + compilation	2.632×	82.0	**+0.495 [+0.468, +0.522]**	**−0.097 [−0.177, −0.019]**	**+0.019 [+0.009, +0.030]**	+0.130 [−0.201, +0.445]	**−1.904 [−2.639, −1.243]**	+0.0092 [−0.0038, +0.0229]	**+0.0036 [+0.0026, +0.0046]**	**+0.0111 [+0.0086, +0.0135]**
	two calls	1.719×	72.8	**+0.403 [+0.382, +0.425]**	+0.007 [−0.067, +0.081]	+0.001 [−0.003, +0.004]	**+0.636 [+0.298, +1.034]**	**−0.636 [−1.164, −0.182]**	−0.0085 [−0.0215, +0.0044]	−0.0001 [−0.0008, +0.0007]	**+0.0037 [+0.0025, +0.0049]**
	lightweight decoder	1.168×	81.6	**+0.036 [+0.030, +0.043]**	−0.018 [−0.057, +0.018]	**−0.003 [−0.005, −0.002]**	−0.014 [−0.158, +0.122]	−0.133 [−0.328, +0.009]	**+0.0045 [+0.0019, +0.0068]**	−0.0001 [−0.0007, +0.0005]	**+0.0023 [+0.0017, +0.0030]**
AnyFlow	base (mean)	8.89 s	17.9		6.732	0.968	26.351	2.864	0.700	0.990	0.982
	two calls + decoder + compilation	3.451×	30.5	**+0.256 [+0.237, +0.273]**	**−0.101 [−0.166, −0.039]**	**−0.007 [−0.012, −0.003]**	−0.014 [−0.311, +0.261]	**−0.616 [−1.319, −0.192]**	+0.0034 [−0.0020, +0.0087]	+0.0003 [−0.0002, +0.0008]	**+0.0030 [+0.0019, +0.0041]**
	two calls	1.468×	17.9	**+0.250 [+0.231, +0.267]**	**−0.092 [−0.141, −0.043]**	**−0.006 [−0.013, −0.001]**	+0.131 [−0.148, +0.411]	**−0.544 [−1.200, −0.163]**	+0.0013 [−0.0044, +0.0066]	−0.0001 [−0.0003, +0.0002]	**+0.0010 [+0.0003, +0.0017]**
	lightweight decoder	1.446×	30.5	**+0.028 [+0.023, +0.032]**	**−0.042 [−0.081, −0.002]**	**−0.002 [−0.003, −0.001]**	−0.104 [−0.213, +0.008]	**−0.082 [−0.185, −0.006]**	**+0.0030 [+0.0015, +0.0045]**	−0.0001 [−0.0006, +0.0003]	**+0.0016 [+0.0011, +0.0023]**
rCM	base (mean)	7.66 s	17.9		6.369	0.938	25.232	4.267	0.680	0.987	0.974
	two calls + decoder + compilation	3.360×	30.6	**+0.288 [+0.268, +0.309]**	+0.024 [−0.061, +0.114]	+0.000 [−0.005, +0.005]	+0.179 [−0.082, +0.433]	**−0.373 [−0.622, −0.180]**	**+0.0091 [+0.0024, +0.0163]**	**+0.0023 [+0.0018, +0.0028]**	**+0.0062 [+0.0049, +0.0076]**
	two calls	1.415×	17.9	**+0.266 [+0.245, +0.287]**	**+0.086 [+0.005, +0.163]**	+0.005 [−0.000, +0.009]	+0.212 [−0.023, +0.442]	**−0.321 [−0.492, −0.172]**	**+0.0072 [+0.0010, +0.0132]**	**+0.0009 [+0.0005, +0.0013]**	**+0.0037 [+0.0026, +0.0047]**
	lightweight decoder	1.566×	30.6	**+0.033 [+0.029, +0.038]**	**−0.042 [−0.075, −0.010]**	**−0.004 [−0.007, −0.002]**	**−0.126 [−0.257, −0.001]**	−0.010 [−0.039, +0.020]	+0.0019 [+0.0000, +0.0038]	**+0.0004 [+0.0001, +0.0006]**	**+0.0018 [+0.0014, +0.0022]**
rCM-14B	base (mean)	27.50 s	42.0		6.628	0.952	25.685	3.274	0.695	0.989	0.978
	two calls + decoder + compilation	2.525×	54.6	**+0.260 [+0.232, +0.289]**	−0.060 [−0.135, +0.002]	**−0.004 [−0.008, −0.002]**	+0.039 [−0.155, +0.224]	**−0.264 [−0.476, −0.116]**	−0.0042 [−0.0119, +0.0031]	**+0.0020 [+0.0013, +0.0030]**	**+0.0069 [+0.0050, +0.0090]**
	two calls	1.798×	42.0	**+0.228 [+0.199, +0.258]**	**+0.083 [+0.028, +0.138]**	−0.001 [−0.004, +0.001]	**+0.193 [+0.002, +0.377]**	−0.174 [−0.464, +0.089]	−0.0056 [−0.0144, +0.0026]	−0.0002 [−0.0009, +0.0006]	**+0.0032 [+0.0018, +0.0049]**

Table 16. D19 held-out runs on PRO6000 BW (33 prompts, seeds 0 and 1, 66 videos per run): end-to-end speedup, peak memory, LPIPS to the model's base, paired differences with 95% CIs (bold: CI excludes 0). Base rows give means. `fis` is our reimplementation, residual reading. Generated from `runs/summaries/d19_recipe.md`.

call fewer for the token oracle ($+0.230$ $[+0.201, +0.257]$, $+0.375$ $[+0.352, +0.398]$; D31b holds), and D20b replicates on this GPU (D31c). Pooled selection alone exceeds one call fewer by $+0.077$ and $+0.103$, and `sap` ties on AnyFlow ($+0.002$) and drifts less on rCM (-0.025), as in D20. D20's drifts are therefore selection effects, not numerical ones.

P. Teacher and student on the same states, and error gain (D21)

D21 tests the central finding where the evidence was weakest. It was registered before any run. While a student samples, its teacher runs on each state the student sees (same latents, time and text, conditional branch; 8 tuning prompts, seed 0). The pairs are AnyFlow and rCM with Wan2.1-1.3B, rCM-14B with Wan2.1-14B, and LTX distilled with LTX dev; klein is not covered. Cosines are between block residuals, averaged over blocks per call and then over calls per prompt. Table 18 gives D21a-1 and D21a-2. D21b applies state errors as in D2, $\epsilon\|x_k\|$ after the step and on rCM through its own sampler's factor, at every call on 33 prompts with seed 0; each model stays on one GPU type with its own compile reference (Tab. 19). On LTX (D21b-1) the first-call values exceed the compile reference, while the last-call values stay below it at every ϵ, so the paper gives absolute values rather than the ratio. On the other students (D21b-2), first minus last call at $\epsilon = 0.1$ is $+0.051$ $[+0.042, +0.060]$, $+0.126$ $[+0.113, +0.139]$, $+0.129$ $[+0.115, +0.143]$ and $+0.221$ $[+0.197, +0.246]$ for AnyFlow, rCM, rCM-14B and Wan2.2. Across three query seeds, the largest range of coverage at 10% of the keys, per prompt and call, is 0.0011, 0.0015 and 0.0011 on AnyFlow, rCM and LTX (D21c; SD across prompts at most 0.013).

Q. rCM's last call, and the same states held out (D22, D23)

D22 asks whether the degradation of rCM's last skip comes from the student or from its sampler, which draws fresh noise before every call: $x_{k+1} = (1 - t_{k+1})(x_k - t_k v_k) + t_{k+1}\epsilon_{k+1}$, with output $x_k - t_k v_k$ and $t \approx 0.61$ at the last call. A velocity predicted on a state built from noise ϵ_k is about $\epsilon_k - \hat{x}_0$, so reusing v_{k-1} at the last call k gives $x_k - t_k v_{k-1} \approx \hat{x}_0 + t_k(\epsilon_k - \epsilon_{k-1})$: the output keeps $t_k \approx 0.61$ times the difference of two noise draws. It compares reuse at the last call, as in the paper, with stopping before that call; stopping keeps the base's first three calls and noise draws and outputs the third call's clean estimate. Held-out prompts, 66 videos, H100 NVL, both scales. Against the base, stopping does not lower aesthetics significantly at 1.3B ($+0.005$ $[-0.029, +0.036]$) and raises them at 14B ($+0.121$ $[+0.084, +0.158]$), with no significant decline in DINO ($+0.000$, -0.002), at LPIPS 0.124 and 0.080. Reuse scores -1.635 $[-1.830, -1.446]$ and -1.537 $[-1.769, -1.328]$ in aesthetics below stopping, and its LPIPS is higher by $+0.442$ $[+0.420, +0.465]$ and $+0.393$ $[+0.377, +0.410]$. Against stopping, reuse also lowers DINO (-0.196, -0.248) and imaging quality (-0.129, -0.126); its RAFT motion rises by 25 and 12, which measures noise rather than motion. D23 repeats D21a on the 33 held-out prompts (seed 0, H100 NVL) with the same thresholds. On all four pairs the ratio is 1.059 $[1.048, 1.070]$, 1.052 $[1.043, 1.063]$, 1.056 $[1.054, 1.058]$ and 1.018 $[1.012, 1.024]$ for AnyFlow, rCM, rCM-14B and LTX, and teacher–student minus student step is $+0.164$ $[+0.157, +0.171]$, $+0.201$ $[+0.190, +0.211]$, $+0.445$ $[+0.439, +0.451]$ and $+0.102$ $[+0.099, +0.106]$. The tuning-prompt values replicate within 0.04, and LTX's fourth call again goes the other way (0.810 vs 0.838).

R. A fine grid, untouched prompts and anchoring at equal budget (D24–D26)

All three were registered after D23, each before its own runs (Tab. 1); the record lists the results available then. D24 runs the redundancy probe on AnyFlow, an any-step student, at 50 and 16 steps on the 4 tuning prompts (B200). The block cosine between consecutive calls, averaged over the calls

Model	Run	n	Density	Ideal speed	LPIPS	Δ aesthetic	Δ DINO	Δ CLIP	Δ motion	Δ imaging q.	Δ flickering
AnyFlow	one call fewer	66		1.178×	0.079	−0.004 [−0.040, +0.031]	−0.001 [−0.003, +0.001]	+0.015 [−0.160, +0.182]	**−0.226 [−0.627, −0.002]**	−0.0012 [−0.0045, +0.0024]	**+0.0006 [+0.0003, +0.0008]**
	pooled top-k	66	0.200	1.194×	0.161	**−0.090 [−0.135, −0.045]**	+0.000 [−0.001, +0.002]	**+0.135 [+0.011, +0.248]**	−0.102 [−0.397, +0.092]	**+0.0048 [+0.0006, +0.0096]**	**−0.0006 [−0.0010, −0.0002]**
	oracle block	66	0.200	1.194×	0.151	**−0.050 [−0.092, −0.007]**	+0.001 [−0.002, +0.004]	+0.103 [−0.053, +0.235]	−0.053 [−0.374, +0.156]	**+0.0068 [+0.0023, +0.0117]**	−0.0003 [−0.0006, +0.0000]
	oracle token	12	0.200	1.194×	0.299	**−0.370 [−0.540, −0.186]**	−0.006 [−0.017, +0.002]	−0.238 [−0.777, +0.277]	−0.274 [−0.837, +0.285]	+0.0012 [−0.0060, +0.0089]	−0.0010 [−0.0025, +0.0004]
	SVG2-style, $p = 0.9$	66	0.231	1.185×	0.080	**−0.059 [−0.091, −0.029]**	**−0.003 [−0.006, −0.001]**	+0.011 [−0.089, +0.116]	−0.042 [−0.096, +0.006]	−0.0008 [−0.0031, +0.0013]	+0.0001 [−0.0000, +0.0003]
	SVG2-style, tuned p	66	0.192	1.197×	0.105	**−0.072 [−0.105, −0.039]**	**−0.004 [−0.008, −0.001]**	+0.031 [−0.077, +0.143]	+0.113 [−0.048, +0.371]	−0.0011 [−0.0037, +0.0014]	+0.0000 [−0.0001, +0.0002]
rCM	one call fewer	66		1.161×	0.218	**+0.061 [+0.018, +0.106]**	**+0.007 [+0.002, +0.011]**	+0.056 [−0.095, +0.215]	−0.193 [−0.401, +0.087]	**+0.0068 [+0.0021, +0.0114]**	**+0.0032 [+0.0027, +0.0038]**
	pooled top-k	66	0.200	1.239×	0.328	−0.006 [−0.072, +0.060]	−0.001 [−0.005, +0.004]	+0.272 [−0.016, +0.590]	+0.123 [−0.103, +0.380]	−0.0054 [−0.0134, +0.0029]	**+0.0023 [+0.0016, +0.0031]**
	oracle block	66	0.200	1.239×	0.353	−0.010 [−0.084, +0.058]	+0.000 [−0.006, +0.006]	+0.160 [−0.147, +0.454]	+0.073 [−0.146, +0.302]	+0.0065 [−0.0006, +0.0145]	**+0.0012 [+0.0004, +0.0021]**
	oracle token	12	0.200	1.239×	0.589	−0.067 [−0.242, +0.140]	+0.018 [−0.029, +0.061]	−0.075 [−1.011, +0.728]	**−1.911 [−3.550, −0.631]**	−0.0131 [−0.0554, +0.0218]	**+0.0080 [+0.0036, +0.0127]**
	SVG2-style, $p = 0.9$	66	0.234	1.227×	0.201	+0.005 [−0.036, +0.050]	−0.001 [−0.005, +0.003]	−0.049 [−0.231, +0.124]	**−0.254 [−0.401, −0.124]**	−0.0010 [−0.0052, +0.0033]	**+0.0005 [+0.0002, +0.0009]**
	SVG2-style, tuned p	66	0.193	1.242×	0.239	−0.005 [−0.049, +0.043]	+0.002 [−0.002, +0.006]	+0.007 [−0.228, +0.225]	**−0.151 [−0.273, −0.016]**	+0.0011 [−0.0034, +0.0056]	**+0.0011 [+0.0006, +0.0017]**

Table 17. D20 held-out runs on 4090s: LPIPS to the model's base and paired differences with 95% CIs (bold: CI excludes 0). Generated from `runs/summaries/d20_selector.md`; motion smoothness is in that file.

Pair	Share, same states	Share, matched noise	T–S − S step	T–S by call
AnyFlow	1.089 [1.066, 1.115]	0.84	+0.145 [0.133, 0.155]	.67, .79, .83, .87
rCM	1.052 [1.043, 1.059]	0.59	+0.238 [0.227, 0.249]	.62, .75, .79, .83
rCM-14B	1.055 [1.052, 1.059]	0.55	+0.449 [0.442, 0.456]	.77, .85, .92, .98
LTX	1.021 [1.009, 1.034]	about 1	+0.100 [0.090, 0.109]	.77, .76, .79, .82, .87, .91, .93

Table 18. D21a. Share: $(1 - \cos)$ of the teacher over $(1 - \cos)$ of the student between consecutive calls, summed over calls. T–S: cosine of teacher and student on one state. S step: the student's cosine across one step.

Model	Ref.	LPIPS by call at $\epsilon = 0.03; 0.1; 0.3$
AnyFlow	.028	.019 .004 .001 .003; .072 .019 .005 .021; .295 .102 .036 .152
rCM	.126	.067 .022 .004 .003; .150 .060 .018 .024; .404 .189 .094 .173
rCM-14B	.138	.078 .017 .002 .003; .150 .044 .009 .021; .388 .131 .036 .148
Wan2.2 A14B	.172	.177 .054 .011 .003; .245 .101 .032 .024; .346 .178 .072 .173
LTX	.076	.082 .050 .028 .004 .001 .000 .000; .175 .114 .067 .014 .003 .001 .001; .301 .210 .143 .051 .017 .007 .008

Table 19. D21b: LPIPS to the model's base after a state error at each call, with the compile reference on the same GPU.

whose timestep falls in each interval of Tab. 2, is 0.963, 0.991 and 0.994 at 50 steps (12, 13 and 12 calls) and 0.822, 0.948 and 0.969 at 16 steps, against 0.48, 0.75 and 0.80 at 4 steps and the Wan teacher's 0.99, 0.99 and 0.995.

D25 tests the baseline on 33 VBench prompts that no earlier run used (drawn with a fixed seed, three per VBench dimension file, excluding every committed prompt set), two seeds each, generated on PRO6000 BW. Margins were fixed before any run from the D19 base runs: 0.2 SD of each released model's per-video score, and 10% of its mean for motion. Table 20 gives every difference. Smoothness and flickering are non-inferior on all four students (D25b) and motion on none (D25d), while aesthetics are non-inferior only on rCM (D25a fails) and imaging quality only on two students (D25c fails). The aesthetic costs on these prompts exceed those on the held-out prompts of D19 (-0.04, -0.10, $+0.02$, -0.06). For scale, the mean aesthetic difference between two seeds of one prompt is 0.19–0.53.

D26 compares Self-Forcing schedules at 16 calls on the same untouched prompts (B200). Against the 4-step output, aesthetics change by -0.014 $[-0.057, +0.030]$ with the two extra steps on the first chunk, by -0.199 $[-0.285, -0.116]$ with them on the last chunk and by -0.198 $[-0.286, -0.115]$ with uniform 2 steps (14 calls). First chunk minus last chunk: aesthetics $+0.185$ $[+0.096, +0.281]$, LPIPS to the 4-step output -0.231 $[-0.263, -0.200]$, DINO $+0.003$ $[-0.007, +0.013]$ (D26a fails). DINO falls against the 4-step output with uniform 2 steps (-0.015 $[-0.029, -0.003]$, D26b holds) and with the anchored schedule (-0.011 $[-0.022, -0.003]$, D26c fails); VBench's three scores do not separate the two 16-call schedules.

S. Method attempts and released alternatives (D27–D30)

All four were registered before any run, after the paper's other results: two attempts to turn its findings into a method (D27, D28), and two tests of released or published alternatives for the problems it found (D29, D30).

D27 asked whether the stock VAE decoder, which decodes the 21 latent frames of an 81-frame Wan video one at a time, leaves a large GPU idle. It splits the latent frames into S temporal windows, primes each later window's causal caches with w latent frames of context, decodes all windows as one batch and drops the context frames' outputs. On one tuning prompt (AnyFlow, PRO6000 BW) the parallel decode approaches the stock decode as the context grows (LPIPS 0.0112, 0.0048 and 0.0019 for $S = 2$ and $w = 1, 2, 4$), but it is slower: 4.17 s at $S = 4$, $w = 2$ and 4.91 s at $S = 8$, $w = 2$, against 2.89 s, about in proportion to the extra context frames it decodes. At 480p the decode is compute-bound on this GPU, so D27 stopped after this validation run, as registered, and makes no claim.

D28 asked whether multiplying a cache's indicator at call k by the student's measured sensitivity there (S4 of Sec. 5.1 of the paper, on the 8 tuning prompts) makes the cache rank skip positions. It does not (Tab. 21). S4 is largest at the last call on AnyFlow, rCM and klein, where noise added late stays visible, while the costliest skip is the first on AnyFlow and klein: a cache's indicator of local change does not measure how large the reuse error is at each call, and a per-call sensitivity does not supply it. Weighting repairs rCM, whose costliest skip is the last; on klein it breaks a ranking the plain indicator had. Weighted TeaCache orders calls 1 and 2 on rCM but not on AnyFlow. As registered, the held-out com-

Student	Δaesthetic	ΔDINO	Δmotion	Δimaging	Δsmoothness	Δflickering
Wan2.2 A14B	†−0.115 [−0.194, −0.026]	−0.002 [−0.008, +0.004]	†−16.7% [−32.7, −4.4]	†−0.0166 [−0.0307, −0.0035]	+0.0027 [+0.0016, +0.0039]	+0.0086 [+0.0064, +0.0110]
AnyFlow	†−0.180 [−0.233, −0.124]	†−0.008 [−0.013, −0.004]	†−17.8% [−34.0, −7.0]	−0.0016 [−0.0117, +0.0065]	+0.0008 [+0.0003, +0.0014]	+0.0034 [+0.0026, +0.0044]
rCM	−0.050 [−0.117, +0.016]	−0.005 [−0.011, +0.000]	†−6.7% [−12.6, −1.3]	+0.0060 [−0.0010, +0.0132]	+0.0029 [+0.0023, +0.0036]	+0.0075 [+0.0062, +0.0089]
rCM-14B	†−0.079 [−0.159, −0.002]	−0.001 [−0.005, +0.002]	†−13.0% [−19.0, −7.8]	†−0.0134 [−0.0220, −0.0056]	+0.0027 [+0.0021, +0.0034]	+0.0079 [+0.0064, +0.0094]

Table 20. D25: baseline − released on untouched prompts (33 prompts × 2 seeds). †: not non-inferior (CI lower bound below −margin). Motion relative to the released mean. LPIPS to the released output 0.437, 0.266, 0.306, 0.278.

parisons (D28b–d) were not run. Both attempts support the paper's conclusion that, with so few calls, StudentProbe's direct measurement on the student is more reliable than a low-cost proxy.

Student	LPIPS of skipping call k	S4 at call k	pairs, plain / weighted
AnyFlow	.231 .103 .100	.0028 .0018 .0365	1/3, 1/3
rCM	.356 .267 .573	.0119 .0070 .0344	1/3, 3/3
klein	.287 .160 .122	.0017 .0007 .0037	3/3, 1/3
LTX	.252 .097 .116 .078 .027 .011	.0062 .0033 .0013 .0006 .0006 .0009	5/15, 9/15

Table 21. D28 stage T (8 tuning prompts, seed 0, B200): fidelity cost of skipping call k (LPIPS to the student's own output), S4 at call k, and the position pairs the plain and the weighted FBCache indicators order as the fidelity cost does. Registered threshold: 8 of 9 over the three 4-step students and 13 of 15 on LTX.

D29 asked whether a released decoder removes the decoder's share without the tiny decoder's quality cost. On the same latents (two calls and compilation, 66 held-out videos per student, B200), LightVAE (lightvaew2_1, the Wan2.1 VAE pruned to a quarter of its channels, then trained and distilled; loaded with LightX2V's model code) decodes in 0.40 s against 1.30 s for the stock VAE and keeps the baseline above 2.5× on AnyFlow and rCM (predicted 2.68× and 2.53×, against 3.71× and 3.70× with the tiny decoder; D29c, D29d hold). It does not score better than the tiny decoder (Tab. 22): it raises CLIP but lowers aesthetics on three students and imaging quality on all four, and it is further from the stock decode (D29a, D29b fail). The stock VAE scores higher aesthetics than the tiny decoder on three students, so the baseline's decoder cost is real; the newer tiny decoder (lighttaew2_1_v2, reported without a prediction) raises CLIP on three students at the same speed.

Student	LPIPS to the stock decode tiny / v2 / LightVAE	LightVAE − tiny Δaes.	ΔCLIP	Δimaging	tiny v2 − tiny Δaes.	ΔCLIP	stock − tiny Δaes.
AnyFlow	.027 / .035 / .040	**−.078**	**+.30**	**−.012**	−.027	**+.33**	+.013
rCM	.038 / .046 / .078	**−.117**	**+.18**	**−.022**	−.025	+.02	**+.053**
rCM-14B	.041 / .052 / .098	+.007	**+.33**	**−.030**	**+.074**	**+.33**	**+.130**
Wan2.2	.035 / .044 / .088	**−.098**	+.14	**−.021**	**+.054**	**+.28**	**+.055**

Table 22. D29 (66 held-out videos per student, B200): decoders on the same latents. Median LPIPS to the stock VAE's decode, and paired differences against the baseline's tiny decoder (bold: 95% CI excludes 0). Decode time: stock 1.30 s (Wan2.2 1.57 s), LightVAE 0.40 s, tiny decoders 0.05 s.

D30 asked whether compensating for the unselected keys closes the gap of Sec. 5.2. We ported Rectified SpaAttn (arXiv 2511.19835; its repository has no license, so its attention is re-implemented on our kernel and only its Hilbert ordering helper is imported) with its settings for the distilled Wan2.1 (20% of blocks plus those holding 30% of the pooled mass, Hilbert neighbors and first-frame blocks, the first two layers dense), and read its speed under an ideal kernel on the B200 probe, as in Sec. 5.2. Keeping every block, the port reproduces dense attention (LPIPS 0.007). With every call sparse it drifts most (Tab. 23); with the first call dense it uses 12% more attention than pooled selection and drifts about as much (D30a fails); at pooled selection's budget (R1-eq, its drop rate chosen on two tuning prompts before stage H) it drifts further. On rCM, at equal latency, it drifts further than one call fewer (D30b holds); on AnyFlow every sparse setting, pooled selection included, is slower than one call fewer under this probe, so D30b is not scored there.

Run	AnyFlow fraction	speed	LPIPS	Δaes.	rCM fraction	speed	LPIPS	Δaes.
one call fewer	–	1.153×	.079	+.005	–	1.128×	.194	+.039
pooled, density 0.2	.402	1.100×	.161	**−.079**	.402	1.128×	.323	−.000
R (theirs, every call)	.268	1.126×	.481	−.094	.268	1.161×	.689	**−.641**
R1 (first call dense)	.451	1.091×	.156	**−.086**	.451	1.116×	.305	**−.227**
R1-eq (pooled's budget)	.405	1.100×	.192	**−.106**	.406	1.127×	.361	**−.307**

Table 23. D30 (66 held-out videos, B200): key-block fraction over all self-attention calls, speed under an ideal kernel on the B200 probe, LPIPS to the student's own output and the paired aesthetic change (bold: CI excludes 0).

T. Supplementary video

The accompanying file `supp_video.mp4` (75 s) shows side-by-side outputs of the same prompt and seed for five claims: rCM-1.3B with call 2 vs call 3 skipped (Tab. 4); LTX distilled as released vs call 1 skipped; AnyFlow with call 2 vs call 3 skipped; Self-Forcing with 4 steps on the first chunk and 2 on the rest vs uniformly 2 steps; and rCM-14B with one call fewer vs sparse attention at equal speed (Tab. 7). Every section uses the first two prompts of a 12-prompt subset of the held-out set (seed 0), chosen by index, not by appearance. The four 1.3B and LTX comparisons were generated on an A6000 and the rCM-14B one on a B200; both sides of each pair come from the same GPU.

U. StudentProbe's probes and method details

Teacher–student pairs. Table 24 lists the pairs of Sec. 3 of the paper with their released sampling settings. Every model runs with its official or model-card sampling settings, including VAE precision and decode settings; for LTX we run the single-scale first pass, without the optional latent upsampler and refinement pass.

Family	Model	Steps	CFG	Output
Wan2.1-1.3B	teacher	50	5.0	$480\times832\times81$
	AnyFlow, rCM	4	–	
	AnyFlow-FAR, Self-Forcing (AR)	4/chunk	–	
Wan2.1-14B	teacher (probes only)	50	5.0	as 1.3B
	rCM (held-out scale)	4	–	
LTX-Video 13B	dev	30	3.0	$512\times704\times121$
	distilled	7	–	
FLUX.2-klein 4B	base	50	4.0	1024^2 image
	distilled	4	–	

Table 24. Teacher–student pairs, run with their released sampling settings (CFG: classifier-free guidance scale).

Table 25 lists which models enter each analysis.

Which GPU is behind each table and figure of the paper. Tab. 1: A800, last row PRO6000 BW. Fig. 1: per panel, in its caption; Fig. 2 is a diagram. Tab. 2, Tab. 3, Figs. 3–5 and Tabs. 3 and 9 here: A800. Tab. 4: PRO6000 BW. Sec. 4.1: A800 (redundancy), B200 (AnyFlow on a fine grid), H100 NVL (same states, held out), A100 (same states, tuning prompts). Sec. 4.2: A800 (LTX diagnostic), 4090 and A100 (gain on four students). Sec. 5.2: A800, B200 and H100 (densities), 4090 (selectors). Sec. 5.1: A800, B200 (VBench), H100 NVL (rCM's last call). Sec. 5.3: A800, B200 (VBench, equal budget). Sec. 6: H100 NVL (decoder, FIS, allocation), PRO6000 BW (baseline, also on untouched prompts, with VBench on a B200). Compile references, LPIPS between compiled and eager outputs: on the A800, AnyFlow 0.027, LTX dev 0.025, LTX distilled 0.075, Wan 0.099, rCM 0.145; on 4090s, AnyFlow 0.028, rCM 0.126; on A100s, rCM-14B 0.138, Wan2.2 A14B 0.172, LTX distilled 0.076.

Which GPU ran what. All timings of P1–P6 and D2–D9g ran on one NVIDIA A800 80GB PCIe GPU (14B: a second one); those servers were then reclaimed, so later experiments ran on B200s, H100s, a PRO6000 BW and, for drift only, 4090s and A100s. A800 80GB PCIe: P1–P6, D2–D8, D9 up to D9g. B200: D9h, D9i, D10, D11, D13, D14 (quality), D15, D16, and the byte-identity check of D17. H100 at a second site: D14 (timing). H100 NVL: D17, D18. PRO6000 BW: D19 (generation, timing). 4090: scoring and VBench of D19; D20; D21 on AnyFlow and rCM. A100: D21 on rCM-14B, Wan2.2 and LTX, its same-state and coverage probes. D24 and D26 (generation and scoring) and D25's VBench: B200. D25 generation and its other scores: PRO6000 BW. D27: PRO6000 BW. D28–D30: B200. D31: RTX 3090. Every pair and speedup stays within one GPU type.

Further values. Our reimplementation of FIS-DiT uses the residual reading, because the literal one gives speckled frames; on Wan2.2 A14B (H100) it lowers motion by 1.88 (on 4.23), and at about equal speed the two-call grid drifts less (-0.087 $[-0.103, -0.070]$) with motion -0.58. Calibration time: the 8-prompt sweep took 0.6–6.9 GPU-minutes of generation per student on an A800 (32–56 videos), and the quality check on 30 prompts generates 60 videos per candidate, so with one re-pick the whole calibration stays under half an hour of generation per student. A per-video choice of the skip position would add little: an oracle gains 2.6% over the best fixed two-call grid on Wan2.2 (registered) and, in an exploratory count, 0–8% over the best fixed skip on six students. Within a step there is little to reuse, since only the first two blocks and the last stay redundant, which caps block-level reuse at about 10%. In AR students the first chunk behaves like a bidirectional student (0.64–0.78 in AnyFlow-FAR, 0.41–0.57 in Self-Forcing), and the context raises later chunks to 0.88–0.93 and 0.65–0.80. Coverage on the image pair stays within 0.02 of the teacher's at matched noise level; in Self-Forcing, the mass of a fixed 256 keys falls from 0.69 to 0.64 across chunks. The unexplained redundancy gap is 0.09–0.24 on rCM, 0.12–0.21 on klein and 0.16–0.29 at 14B. In the middle blocks, 84–100% of the tokens change their residual by under 20% between teacher steps, but 0–1% for every student; Self-Forcing's last block stays token-stable (93–99%). `torch.compile` gives 1.10–1.25× on Wan and 1.18× on LTX; on LTX, INT8 moves the output far (LPIPS 0.469, 0.357). Sparse attention at a single call costs LPIPS 0.430, 0.136, 0.047 and 0.023 on AnyFlow and 0.649, 0.307, 0.195 and 0.081 on rCM for calls 0 to 3. At density 0.2 with the first call dense, sparse attention drifts 0.163 vs 0.081 for one call fewer on AnyFlow ($+0.082$ $[0.074, 0.089]$) and 0.352 vs 0.221 on rCM ($+0.131$ $[0.119, 0.142]$); held out the differences are $+0.082$ and $+0.118$, and $+0.176$ at 14B. On LTX, sparse attention reaches LPIPS 0.335 and reusing the last call 0.009. On AnyFlow, slower sparse settings add operating points that a 4-step grid lacks (density 0.2, two dense calls: 1.06×, -0.016). Paired against uniform 3 steps, the anchored AR schedules differ by -0.182 and -0.158. The perturbation gain (S4) misses the flow-map students because Gaussian noise added at the last call stays visible, whereas a structured reuse error there costs little fidelity.

Block probe. Forward hooks on every transformer block record the residual (output minus input) of the conditional branch. For CFG-batched pipelines (LTX dev), only the

Model	Coverage	Redundancy	Same states	Error gain	Skip sweep, quality	Caches	Sparse	Latency split	AR schedule	Baseline
Wan2.1-1.3B teacher	•	•	•	–	–	–	○	•	–	–
AnyFlow	•	•	•	•	•	•	•	•	–	•
rCM	•	•	•	•	•	•	•	•	–	•
AnyFlow-FAR (AR)	•	•	–	–	–	–	–	•	–	–
Self-Forcing (AR)	•	•	–	–	–	–	○	•	•	–
Wan2.1-14B teacher	•	•	•	–	–	–	–	–	–	–
rCM-14B (held-out scale)	•	•	•	•	•	•	•	•	–	•
LTX-Video dev	•	•	•	–	–	–	○	•	–	–
LTX-Video distilled (held-out family)	•	•	•	•	•	•	•	•	–	–
FLUX.2-klein base	•	•	–	–	–	–	–	–	–	–
FLUX.2-klein distilled	•	•	–	–	•	○	○	•	–	–
Wan2.2 A14B, 4-step (LightX2V)	–	–	–	•	•	•	•	•	–	•
HunyuanVideo-1.5, 4-step (LightX2V)	–	–	–	–	•	•	–	•	–	–

Table 25. Which models enter which analysis. •: measured; ○: signal ranking only (caches) or kernel speed at the model's attention shape only (sparse). Coverage includes the AR chunks at equal key count (D13) and the 14B pair (D10); redundancy includes AnyFlow at 16 and 50 steps (D24); same states are D21 and D23, error gain D21b; caches are student-tuned FBCache and TeaCache (D7, D9h, D16f, D16g) and the low-cost signals of Tab. 3; sparse is drift against one call fewer (Sec. 5.2, D9i, D14, D16h, D20, D30, D31); the baseline is Tab. 4 of the paper, D19 and D25.

conditional element is recorded. The cosine to the previous call's residual of the same block is averaged over blocks and prompts. Teacher calls are matched to a student call by nearest noise level.

Oracle coverage. For 256 random queries in every fourth self-attention layer, we compute the exact softmax over all keys and sum the largest p fraction of the probabilities. For FAR, masked and padded keys are excluded.

Residual reuse. At a skipped call, the whole block stack is replaced by the previous computed call's stack residual, added to the current input. Embeddings and the output head run as normal. Taylor forecasting extrapolates the residual to first order from the last two computed calls. Wherever it is defined, it costs more fidelity than plain reuse at the same call: LPIPS 0.131 vs 0.078 on AnyFlow, 0.379 vs 0.283 on rCM, 0.160 vs 0.106 on klein, and $+0.017$ to $+0.028$ at LTX calls 2–6.

FBCache and TeaCache on students (D7). Our FBCache follows ParaAttention's rule: block 0 always runs, and if the relative L1 change of its residual since the last fully computed call is below τ, the remaining blocks are skipped and their cached residual is added. TeaCache uses the official Wan2.1-1.3B polynomial and accumulate-and-reset rule. Thresholds were tuned on the 8 tuning prompts (seed 0) from always-compute runs that log each signal: for FBCache, τ is the midpoint between sorted signal values whose simulated mean number of skipped calls is closest to one; for TeaCache, the smallest such midpoint at which exactly one call is skipped. A verification run at the chosen τ skipped exactly one call in every tuning video for all three students.

Perturbation diagnostics. *State mode* (D2) adds Gaussian noise δ to the model output at call k such that $|\Delta\sigma_k| \, \|\delta\| = \epsilon \|x_k\|$. *Output mode* (D3, signal S4) uses $\|\delta\| = 0.1 \|v_k\|$. The noise direction uses a fixed seed for all prompts.

Numerics. Quality runs disable TF32 in matrix multiplications and cuDNN convolutions, for reproducibility. Timings of as-released configurations use PyTorch's defaults, under which cuDNN convolutions run in TF32. Disabling TF32 slows an fp32 video decoder about 6.7× (Wan: 39 s vs 5.8 s) and leaves bf16 components unchanged. It therefore enters teacher-to-student ratios: with the Wan teacher's fp32 decoder, the teacher-to-AnyFlow speedup is 15.0× under PyTorch's defaults (14.8× with both decoders in bf16), but 17.5× with cuDNN's TF32 disabled. Our first speed runs disabled cuDNN TF32 as well. For the bf16 models this changes nothing: AnyFlow takes 12.75 ± 0.05 s with TF32 disabled and 12.72 ± 0.06 s under the defaults, on two hosts with identical A800s. Every configuration with an fp32 decoder was re-timed under the defaults, and every ratio involving one uses runs on the same GPU.