

Sparsity Should Be ALLOTted by Output: Training-Free Allocation of Block-Sparse Attention for Video Diffusion Transformers

Tianze Xia^{1,2,4*} Xiaoling Zhou^{3,4} Ruya Lan¹ Yiqiao Liao⁴ Chengru Song⁴ Shiming Xiang^{1,2†}
¹UCAS ²CASIA ³PKU ⁴KlingAI Research

Abstract

Full 3D attention is costly in video Diffusion Transformers. Block-sparse attention reduces this cost by computing only selected attention blocks. However, existing methods either overlook the potential to optimize sparsity allocation or rely on heuristic rules or local metrics to guide it. In this work, we observe that sparsifying attention in different layers and at different denoising steps affects final-video fidelity to substantially different degrees. Yet the identities of the most sensitive layer-step pairs remain nearly unchanged across inputs. Building on these findings, we propose Allocating Layer-step Latency by Output Tests (ALLOT), a training-free sparsity allocation method. Treating each layer-step pair as a cell, ALLOT estimates each cell’s contribution to final-video error through (i) cell probing and (ii) trajectory calibration, then jointly optimizes cell densities under a measured-latency budget. It preserves existing selection strategies and kernels, replacing only the allocation component. At equal or shorter wall-clock time, ALLOT outperforms every compared allocation strategy with XAttention, with an 11.8 dB PSNR gain on HunyuanVideo. Extensive experiments across four additional published pipelines on Wan2.1, Wan2.2, and HunyuanVideo show that ALLOT outperforms or ties the default in all nine settings.

1. Introduction

Diffusion Transformers (DiTs) [17] have been widely adopted for video generation. Models such as Wan [19] and HunyuanVideo [11] produce high-quality, spatio-temporally consistent videos, yet the cost of their full 3D attention grows quadratically with the number of tokens [18].

On a single GPU, Wan2.1-14B takes roughly one hour to generate a 720p, 81-frame video, and attention accounts for more than half of the end-to-end time.

Sparse attention has been widely studied as a means of reducing this computational cost, with existing approaches broadly categorized as either training-free or training-based.

Many training-free methods [14, 24, 25, 29] adopt *block-sparse attention*, which partitions query and key tokens into blocks. These methods estimate the importance of each query-key block pair and compute dense attention only for selected pairs. These works primarily address two questions: which blocks participate in the attention computation, and how to execute the resulting sparse computation efficiently at the kernel level.

Notably, beyond determining which blocks participate in the computation, the number of blocks retained in each layer at each denoising step also significantly affects generation quality and latency. In threshold-based methods [14, 24, 29], the thresholds serve as an implicit allocation rule. With all other conditions identical, changing only the allocation can substantially alter final-video fidelity. Despite the existence of some explicit allocation approaches, existing methods generally have the following two limitations:

- **Heuristic rules do not transfer reliably across settings.** Many works use fixed rules, such as keeping the first layer or the first 10–30% of denoising steps dense [12, 14, 25]. Our measurements in §3 show that effective allocations vary across models and selection strategies. Consequently, a rule tuned for one setting does not reliably transfer to another and must be re-tuned.
- **Local metrics do not capture final-video effects.** Automatic allocation methods commonly rely on local proxies, such as cumulative attention mass [14, 24], Hellinger distance [36], and one-step output error [3, 35], rather than direct measurements of final-video error. Our analysis in §3 shows that the rankings induced by these proxies disagree with those based on final-video error, and allocations guided by these metrics can even yield lower final-video fidelity than a uniform baseline.

To address these limitations, we term each (layer, step) pair a *cell* and evaluate it directly by its *sensitivity*, defined as the degradation it induces in the final video relative to the dense output. Building on this formulation, we propose Allocating Layer-step Latency by Output Tests (ALLOT), a training-free latency allocation method.

ALLOT estimates each cell’s sensitivity from two factors: cell probing measures the error injected at its step, and tra-

*Work done during an internship at KlingAI. †Corresponding author.



Figure 1. **Changing only the allocation at equal wall-clock time.** Left: dense, baseline and + ALLOT (top to bottom), with blue-box insets; prompts are near the median gain. Right: paired PSNR gains (ALLOT – baseline) with 95% CIs for Tabs. 1 and 2; open markers denote ties.

jectory calibration measures how subsequent denoising steps amplify or attenuate that error.

It then jointly allocates all cell densities under a measured-latency budget, so warm-up length and dense layers emerge from the solution rather than hand-set rules. Because sensitivity rankings remain stable across inputs for a fixed model and selection strategy (§3), we perform both measurements once offline and reuse their results across prompts. During inference, ALLOT looks up densities in a precomputed table and changes only the allocation in existing block-sparse attention pipelines.

Overall, our main contributions are threefold:

- We show that final-video sensitivity varies by orders of magnitude across layers and steps and is misjudged by local metrics, while sensitivity rankings remain stable across inputs.
- We propose ALLOT, which estimates each cell’s error by combining a one-step cell probe with a trajectory gain and jointly solves for all cell densities under a measured-latency budget, replacing only the allocation of existing pipelines.
- Extensive experiments across six selection strategies and three model families show that ALLOT outperforms or ties every compared allocation at equal or shorter wall-clock time (§5).

2. Related Work

Block-sparse attention. Block-sparse attention methods for video DiTs are either trained or training-free. Trained methods (VSA [30], VMoBA [21], NABLA [15]) train or fine-tune the model with sparse attention. Training-free methods either apply static masks [12, 31, 36] or select blocks at run time [14, 22–25, 29]. ALLOT retains the selection strategies and kernels of existing methods, replacing only the allocation (§4.5).

Allocation in existing methods. Existing methods allocate density, a threshold or a sparse pattern per layer [3], per (layer, head) [1, 14, 35], per step [9, 33], per (layer, step) [28], or per (step, layer, head) [27, 36], based on attention mass, Hellinger distance, one-step output error or other local metrics. Across steps, most use a heuristic warm-up or a preset shape, or treat all steps equally. ALLOT also allocates per (layer, step) but calibrates each cell against the final video to capture cross-step error amplification and solves for all cells jointly under a measured-latency budget.

3. Motivation and Key Observations

Setup. Let ℓ index layers and s index denoising steps. The sensitivity $S(\ell, s; b)$ of cell (ℓ, s) measures how much sparsifying that cell to density b degrades the final video relative to the dense reference. Measuring this directly for every cell would require thousands of full generations and is therefore impractical. We instead decompose it into two cheaper signals: the one-step output error when one layer is sparsified and the final-video error when a window of steps is sparsified (§4.2, §4.3).

Observation 1: Layer sensitivity varies widely and is not reliably reflected by attention-map metrics. As shown in Figure 2(a), the first one or two layers have one-step errors up to 154 times the median-layer error. When evaluated on the final video, the most sensitive layers coincide with these high-error layers. Attention-map metrics (attention mass [14] and Hellinger distance [36]) yield different sensitivity rankings, and allocations based on them degrade fidelity (Appendix B). Cell probes therefore provide a more reliable basis for layer allocation. The most sensitive layer varies across models, and a fixed “first layer dense” rule therefore does not transfer reliably across settings.

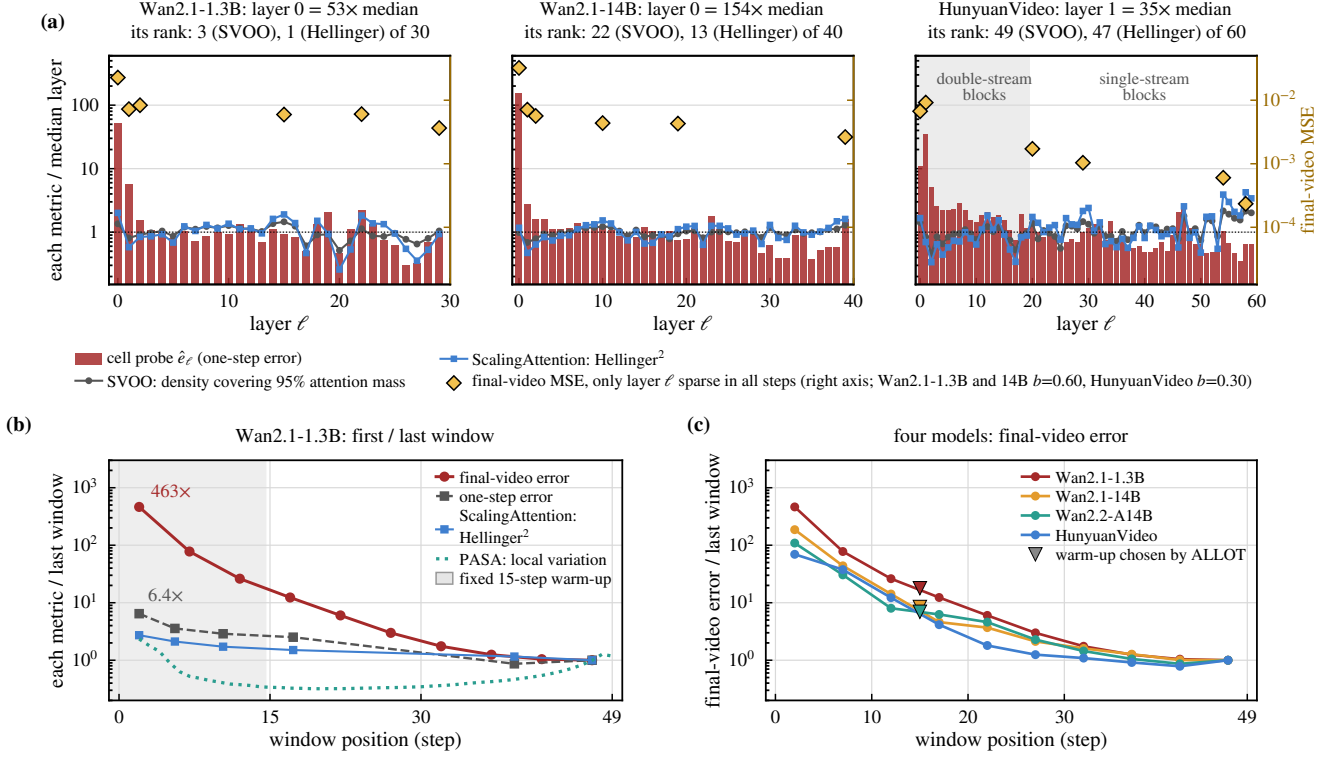


Figure 2. **Layer and step sensitivity.** (a) Single-layer one-step error ($b=0.30$, $t=0.85$; bars), attention-map metrics [14, 36] (lines), and final-video MSE with that layer sparse throughout (diamonds). Titles give the most sensitive layer and its metric ranks. (b) Wan2.1-1.3B window errors versus local metrics [33]; annotations give first/last-window ratios. (c) Window errors across four models.

Observation 2: Final-video effects vary sharply across denoising steps and are understated by one-step errors. As shown in Figure 2(b, c), earlier windows produce markedly larger final-video errors than later ones under the same sparsification configuration. The first-to-last-window MSE ratio reaches several hundred for some models. By contrast, the corresponding one-step errors show a much smaller difference and even reverse the ordering of the first and last windows on Wan2.2. One-step errors therefore require trajectory calibration before guiding per-step density.

Observation 3: Sensitivity rankings remain stable across inputs for a fixed model and selection strategy. Across all tested models, per-input sensitivity rankings closely match rankings aggregated over calibration inputs, with median Spearman correlations of 0.80–1.00 (Appendix D). Prior work reports similar input stability for attention-map metrics [14, 36]. Our results extend this observation to output-error sensitivity, allowing cell probes and trajectory gains to be measured offline and reused across prompts.

4. ALLOT

ALLOT jointly optimizes the density of every (layer, step) cell under a measured-latency budget, minimizing the esti-

mated total sensitivity. It consists of three modules: cell probing (§4.2), trajectory calibration (§4.3) and latency-constrained allocation (§4.4), built on the first-order analysis and calibrated estimate of §4.1. §4.5 describes inference and integration with existing pipelines.

4.1. Problem formulation

With thousands of cells in one denoising process, measuring each one on the final video is prohibitively expensive. We therefore approximate the final-video error as a sum of per-cell contributions and let a solver optimize the result.

First-order motivation and calibrated estimate. The error introduced at one step is carried to the final video by every subsequent step [7, 13]. Let ϵ_s denote the error that step s 's sparse approximation introduces in the velocity prediction, and $\epsilon_{\ell s}$ denote the velocity-prediction change when only layer ℓ is sparsified. A first-order perturbation analysis around the dense trajectory approximates the final-latent deviation as

$$\delta \approx \sum_s P_s \epsilon_s, \quad (1)$$

where P_s is the linearized propagation operator from a velocity error at step s to its effect on the final latent (Ap-

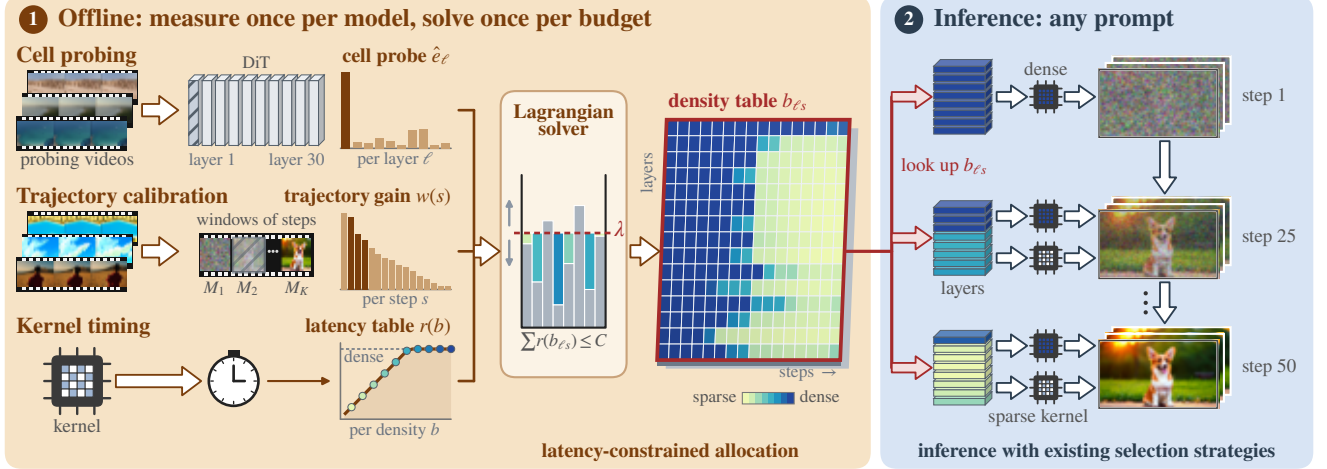


Figure 3. **ALLOT overview.** ① Offline: measure cell probes $\hat{e}_\ell$, trajectory gains $w(s)$ and kernel latency $r(b)$; solve a density table for each latency budget C (§4.4). ② Inference: look up each cell’s density for any prompt, using dense attention when $b_{\ell s}=1$ and the existing sparse kernel otherwise.

pendix E). We make two approximations. **(a) Additivity:** layer perturbations superpose to first order within one step ($\epsilon_s \approx \sum_\ell \epsilon_{\ell s}$), and errors of different cells are orthogonal in expectation. **(b) Separability:** downstream amplification after step s depends only on s , not on the layer (Appendix E). Under these approximations,

$$\mathbb{E}\|\delta\|^2 \approx \sum_{\ell, s} g(s) \mathbb{E}\|\epsilon_{\ell s}\|^2, \quad (2)$$

where $g(s)$ is the propagation gain after step s in the first-order model.

Equation (2) motivates factorizing each cell’s contribution into within-step error and downstream amplification. Because scalar within-step MSE does not specify how an error propagates, the analysis does not determine $w(s)$ (Appendix E). We calibrate $w(s)$ using final videos generated with one window sparsified (§4.3). Together with the cell probes, this gives the estimate of the sensitivity defined in §3:

$$\hat{S}(\ell, s; b) = w(s) \hat{e}_\ell(t_s, b). \quad (3)$$

Here t_s is the step’s noise level and $\hat{e}$ is the within-step cell probe (§4.2). The two factors serve complementary roles: the cell probe distinguishes layers and candidate densities within a step, whereas $w(s)$ places different steps on a common final-video-error scale. Only relative sensitivities affect allocation: uniformly scaling the estimated error does not change the optimal table at a fixed latency budget.

Optimization problem. Given a latency budget C , find the density $b_{\ell s}$ of every cell:

$$\min_{b_{\ell s}} \sum_{\ell, s} \hat{S}(\ell, s; b_{\ell s}) \quad \text{s.t.} \quad \sum_{\ell, s} r(b_{\ell s}) \leq C, \quad (4)$$

where $b_{\ell s}=1$ means fully dense (error counted as zero). $r(b)$ is the measured wall-clock time of a single cell’s attention at density b (the latency table, §4.4). r and C are both normalized so that a dense cell equals 1. Because non-attention computation does not change with allocation, constraining attention time approximately constrains end-to-end latency.

4.2. Cell probing on the model output

The cell probe $\hat{e}_\ell(t, b)$ measures a layer’s output error within one step. It is used to compare layers within the same step (Observation 1) and to compare one cell’s error across densities. We add noise at level t to a small number of real videos, sparsify only layer ℓ to density b while keeping all other layers dense, and measure the MSE between this step’s velocity prediction and the dense model’s prediction [3]. At deployment, all other layers are sparse at the same time, unlike the measurement condition. This difference is analyzed in Appendix F.

Each measurement needs only one forward pass. By Observation 3, layer rankings are input-stable, so a few probing videos suffice (§5.6). Since rankings differ across selection strategies, probes are by default measured under the selection strategy used at deployment. We tabulate probes on a grid over density and noise level, average over the probing videos, and interpolate in $\log b - \log \hat{e}$ when solving (grid in §5.1).

4.3. Trajectory calibration against the final video

The trajectory gain $w(s)$ supplies the inter-step amplification that one-step errors cannot capture (Observation 2). It reuses the window experiments of Figure 2(b, c). Denoising steps are divided into contiguous windows. Only window Ω_k has all layers sparsified to a fixed window den-

sity b_0 , with the remaining steps dense. The generated video’s pixel MSE against the dense reference is M_k . Let $E_k = \sum_{s \in \Omega_k} \sum_{\ell} \hat{e}_{\ell}(t_s, b_0)$ be the sum of the probes of all cells sparsified in this run. The gain is the final-video error per unit of probe error, shared by all steps in the window:

$$w(s) = M_k / E_k, \quad s \in \Omega_k. \quad (5)$$

By construction, summing Equation (3) over the window’s cells recovers the measured M_k . The solver uses only the product $w(s) \hat{e}_{\ell}(t_s, b)$. Calibrating on sparse-window runs retains the error structure of the sparse-attention path used to measure M_k . The resulting $w(s)$ captures downstream propagation, average cross-layer interactions and latent-to-pixel conversion. Appendix E provides an oracle diagnostic of this choice. Sharing $w(s)$ across layers is the separability approximation. Each window therefore needs one generation per prompt with all layers sparsified together, independent of the number of layers and the density grid (settings in §5.1).

4.4. Latency-constrained allocation

Latency table. $r(b)$ is the time of one attention call at density b divided by that of dense FlashAttention [2] on the same shape, measured with the kernel and resolution used at deployment. $r(b)$ is not proportional to density. A sparse kernel is faster than dense only below a crossover density. Assuming cost proportional to density or FLOPs would therefore overestimate the gain from sparsity. When solving, r is clipped to 1, and cells whose density would lie above the crossover are set to dense.

Solver. Equation (4) is solved by Lagrangian relaxation [3]. The candidate density set $\mathcal{B}$ is a set of equally spaced sparse densities plus dense ($b=1$). Cell probes are interpolated to $\mathcal{B}$ and the step’s noise level, then multiplied by $w(s)$ to obtain $\hat{S}(\ell, s; b)$ for each candidate. The solver runs a bisection on λ . Given λ , each cell independently picks the density minimizing $\hat{S}(\ell, s; b) + \lambda r(b)$ over $\mathcal{B}$, yielding a density table. If the table’s total cost exceeds C , λ is increased. Otherwise, it is decreased. This process continues until convergence (Algorithm 1, Appendix F).

Proposition 1 *For any $\lambda \geq 0$, the density table b^{λ} obtained by independently minimizing $\hat{S}(\ell, s; b) + \lambda r(b)$ over $\mathcal{B}$ for each cell is an exact optimum of Eq. (4) (restricted to $\mathcal{B}$) when the budget equals the table’s own total cost [6]. The result applies to the discrete estimated objective and requires no convexity (Appendix E).*

The bisection locates the two Lagrangian tables bracketing C , which typically differ in only one cell (Appendix E). Warm-up length, which layers stay dense and how density varies across steps all emerge from the solver without heuristic rules (Appendices F and J).

4.5. Inference and integration with existing selection strategies

Inference. The density table is solved offline, once per model, selection strategy and latency budget. At generation time, the density of each layer is set from the table before each step s . Cells with $b_{\ell s}=1$ use the original dense attention. All other cells are sparsified by the selection strategy at density $b_{\ell s}$ and computed by the original kernel. The same table is used for all prompts. Changing the budget only requires re-solving one table.

Integration. ALLOT keeps each selection strategy’s kernel and ranking, replacing only the cutoff that determines how many ranked blocks or clusters are retained. Threshold-based defaults (SVOO, SVG2, SpargeAttn and XAttention) are converted to a density cutoff. For SpargeAttn, this uses its official top- k API. This removes native per-head or per-input adaptation. Splitting by SVOO’s head densities is 1.44 dB worse than an equal split (Appendices B and F). Throughout, “changing only the allocation” refers to this integration.

Supported interfaces. ALLOT applies when a selection strategy supports a controllable per-cell density and its latency can be measured. Fixed-mask strategies such as Radial are restricted to densities reachable by their mask parameters. Density denotes retained blocks for mean-pooled top- k , SpargeAttn, Radial and XAttention, and retained key tokens for the variable-size clusters of SVOO and SVG2.

5. Experiments

5.1. Experimental setup

Models and data. Four models: Wan2.1-T2V-1.3B and 14B (hereafter 1.3B and 14B), Wan2.2-T2V-A14B [19] and HunyuanVideo [11]. Resolution and frame count are given in each table, following a published setting where one exists. The analyses in §3 use 480p×33. All use 50 denoising steps. Wan2.2 results are in Tabs. 2 and 3, and HunyuanVideo results are in Tab. 2 and Appendices G and J. The evaluation set consists of 32 VBench [10] prompts.

Metrics. Fidelity metrics (PSNR, SSIM [20] and LPIPS [32]) measure similarity to the dense reference (same prompt and seed) rather than absolute generation quality. Perceptual quality is measured by VBench [10], weighted over the six dimensions used by HEART [35]. At matched or shorter wall-clock time, a win requires a positive lower bound of the paired 95% PSNR-difference interval and lower mean LPIPS, while a loss requires a negative upper bound and higher mean LPIPS. All other cases are ties, and VBench does not enter the verdict (Appendix A).

Baselines. Published pipelines [12, 14, 24, 25, 29] follow their documented settings (SVG2 uses HEART’s five-step warm-up), while allocation baselines [3, 9, 27, 28, 33, 35, 36] are re-implemented from their papers (Appendices G and H). For mean-pooled top- k , the uniform baseline keeps the first 15 steps dense. The strengthened baseline additionally keeps layer 0 dense and tunes the warm-up on 1.3B (Appendix J).

Implementation details. Mean-pooled top- k (the coarse stage of VSA [30]) reorders tokens cube by cube ($3 \times 4 \times 4$ cubes) and groups every 64 consecutive tokens into a block. Cell probes are measured on four OpenVid [16] real videos over a grid of 8 densities $\times$ 9 noise levels. Trajectory gains are measured using 10 five-step windows at window density $b_0=0.30$.

5.2. Comparison with other allocation methods

To compare allocation strategies in isolation, we fix XAttention’s selection strategy and kernel, changing only the allocation (Tab. 1). ALLOT outperforms every compared allocation method, all of which use hand-set rules or local metrics, by 1.53–7.31 dB.

The effect of density truncation depends on the selection strategy: it lowers SVOO’s PSNR but improves XAttention’s. ALLOT further improves on XAttention’s density-cutoff default at matched wall-clock time (Appendix F).

VBench scores remain close to those of the baselines. ALLOT also outperforms every compared allocation method on additional XAttention settings spanning higher resolution and HunyuanVideo, all at equal or shorter wall-clock time (Appendix G).

5.3. Integration with published pipelines

We integrate ALLOT into the official pipelines of SVOO [14], SVG2 [25], SpargeAttn [29] and Radial [12], changing only the allocation (Tab. 2, details in Appendix H). ALLOT outperforms the default in six of the nine settings (by up to 5.03 dB in PSNR) and ties in the other three (Appendix H). Notably, on HunyuanVideo (13B) with SVOO, ALLOT improves PSNR by 3.38 dB while increasing end-to-end speedup from $1.75\times$ to $1.90\times$. Figure 4 shows qualitative comparisons.

To compare across speeds, we solve ALLOT’s tables at several latency budgets on three published pipelines (1.3B 720p) and plot PSNR against end-to-end speedup (Figure 5). All of the pipelines’ own settings (open circles) lie below ALLOT’s curve.

5.4. Cross-model comparison

On the three Wan models with mean-pooled top- k , ALLOT outperforms both the uniform and the strengthened baseline (Tab. 3). We extend this comparison to HunyuanVideo at

equal density in Appendix J. Figure 6 shows what a fixed rule misses: the first few layers and certain deeper layers stay dense longest, while density decreases over several intermediate steps rather than dropping at a single step.

The best rule also changes with the budget, so we test the baselines at other budgets: ALLOT remains above the uniform baseline across budgets at 720p $\times$ 81 (1.3B and 14B, Appendix H). On 1.3B 480p $\times$ 33, it also outperforms the strengthened baseline with its warm-up re-searched at each of three budgets (Appendix J).

5.5. Ablation study

Each ablation replaces one component in the XAttention setup of Tab. 1 at ALLOT’s cost. Trajectory calibration is dominant: setting $w(s) = 1$ for every step lowers PSNR by 7.05 dB, and DiTFastAttn’s greedy rule on the same one-step errors reaches a similar 19.38 dB (Tabs. 1 and 4). A shared density across layers and a parametric shape rule also underperform, whereas density-proportional cost makes no significant difference after wall-clock alignment. Mean-pooled top- k ablations and an LPIPS-calibrated variant are reported in Appendix I.

5.6. Verification, robustness and offline cost

Verifying the sensitivity estimate. The solver ranks density tables by their estimated total sensitivity, the sum of $\hat{S}$ over all cells (Eq. (3)). We validate this ranking against measured final-video error. At fixed total cost, permuting ALLOT’s densities among cells lowers PSNR on every prompt, and the measured ordering matches the prediction. Across 80 tables at multiple budgets on held-out prompts, the predicted and measured rankings agree strongly (pooled Kendall $\tau = 0.83$, Appendix E).

Number of calibration inputs. Cell probes use four probing videos and trajectory calibration uses eight prompts. With a single probing video or calibration prompt, most cells keep the same dense/sparse status. On the XAttention setting of Tab. 1, eight probing videos bring no significant gain over four, and one or two lose less than a tenth of ALLOT’s gain (Appendix K). Window density and probe resolution have mild effects. Calibration with SpargeAttn rather than mean-pooled top- k gives a 0.993 correlation in w and 96.0–96.5% cell-decision agreement (Appendix K).

Offline cost. Measurement takes roughly 6–23 GPU-hours per model (over 30 for the two experts of Wan2.2) and is done once per model and selection strategy (Appendix F). Changing the latency budget or the number of steps only requires re-solving the table, which takes about 4 s on CPU. With 30 steps, CFG 7.5 or the Euler sampler, ALLOT still outperforms the default without re-measuring (Appendix K).

Table 1. **Comparison with other allocation methods** (XAttention on Wan2.1-1.3B 480p×81, $n=32$). Default: XAttention’s published configuration. Rows grouped by wall-clock time, with ALLOT re-solved for each of the first two groups. The third group applies HEART’s cross-step mask reuse (TMR) to both sides. Bold: best PSNR/SSIM/LPIPS in group. Details in Appendix G.

Allocation	PSNR↑	SSIM↑	LPIPS↓	VBench↑	Wall-clock (s)	Speedup
Dense	—	—	—	72.35	183.8	1.00×
Default	18.14	0.659	0.268	70.27	148.7	1.24×
CalibAtt-style	19.67	0.700	0.222	70.79	149.6	1.23×
ALLOT	25.37	0.835	0.106	71.24	148.6	1.24×
HEART (EBC)	19.43	0.699	0.221	72.02	151.7	1.21×
Efficient-vDiT	19.08	0.702	0.225	72.32	151.1	1.22×
DiTFastAttn-style	19.38	0.708	0.219	72.27	152.2	1.21×
ScalingAttention (FAST)	22.75	0.790	0.141	71.09	152.9	1.20×
PASA	21.91	0.768	0.163	71.25	151.8	1.21×
DFSAttn-style	24.86	0.834	0.107	71.20	151.3	1.21×
ALLOT	26.39	0.854	0.089	71.38	151.3	1.21×
HEART (EBC + TMR)	19.54	0.706	0.214	71.62	145.0	1.27×
ALLOT + TMR	26.13	0.847	0.094	71.34	146.0	1.26×

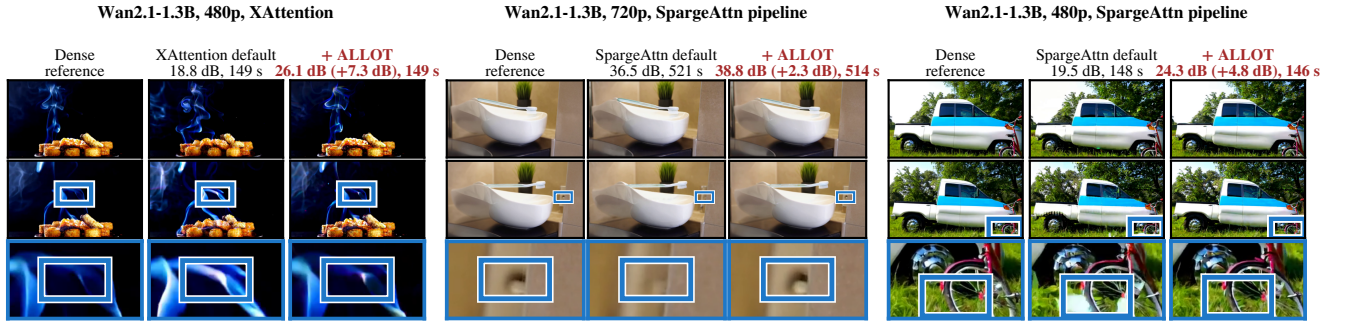


Figure 4. **Qualitative comparison.** Dense, baseline and +ALLOT (left to right) at the first and last frames; bottom rows enlarge the boxed regions, with inner boxes marking larger baseline deviations from dense. Headers give PSNR to dense, gain and wall-clock time. Prompts are near the median gain. Settings: Tab. 1 (XAttention), Appendix H (SpargeAttn 720p), and Tab. 2 (SpargeAttn 480p).

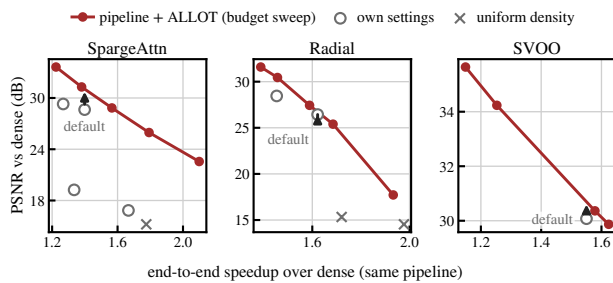


Figure 5. **Fidelity-speedup curves** (Wan2.1-1.3B 720p; prompts as in Tab. 2; y-ranges differ). Open circles: pipeline settings (SpargeAttn: default, two released and one retuned checkpoint; Radial: default and decay 1.0; SVOO: default). Arrows show gains at the default’s speed.

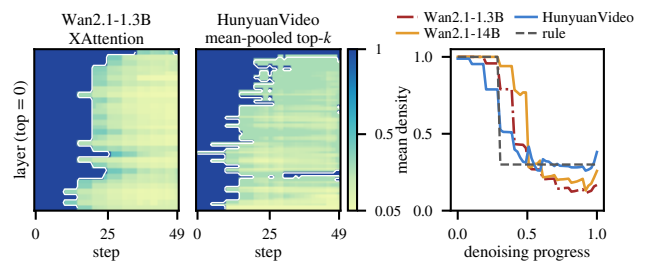


Figure 6. **Solved density tables** for XAttention (Tab. 1) and HunyuanVideo (equal density; Appendix J); white contours mark dense cells. Right: mean density per step, including Wan2.1-14B (Tab. 3), versus a rule keeping the first 30% of steps dense and using 0.30 thereafter.

6. Conclusion

We showed that layer-step sensitivity varies across models and selection strategies, is not reliably captured by local

Table 2. **Integration with published pipelines, changing only the allocation** ($n=32$; speedup = dense wall-clock / row wall-clock; dense times and paired intervals in Appendix H). Bold: best PSNR/SSIM/LPIPS in each setting.

Pipeline & setup	Allocation	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	VBench $\uparrow$	Wall-clock (s)	Speedup
Wan2.1-1.3B 480p $\times$ 81 SpargeAttn	Default	22.67	0.768	0.150	70.03	147.7	1.24 $\times$
	+ ALLOT	27.70	0.876	0.070	71.58	145.8	1.26 $\times$
Wan2.1-1.3B 720p $\times$ 81 Radial	Default	26.44	0.832	0.143	71.71	448.9	1.62 $\times$
	+ ALLOT	26.83	0.837	0.138	71.08	441.5	1.65 $\times$
Wan2.1-1.3B 720p $\times$ 81 SVG2	Default	24.25	0.808	0.151	71.09	410.9	1.66 $\times$
	+ ALLOT	25.60	0.819	0.156	70.75	410.5	1.66 $\times$
Wan2.1-1.3B 720p $\times$ 81 SVOO	Default	30.08	0.893	0.082	72.12	451.6	1.55 $\times$
	+ ALLOT	30.73	0.890	0.095	71.45	447.7	1.56 $\times$
Wan2.1-14B 720p $\times$ 81 Radial	Default	24.75	0.839	0.128	71.70	2296.9	1.56 $\times$
	+ ALLOT	25.28	0.850	0.115	71.65	2290.7	1.56 $\times$
Wan2.1-14B 720p $\times$ 81 SVOO	Default	28.15	0.903	0.068	71.67	2603.5	1.31 $\times$
	+ ALLOT	31.70	0.930	0.044	71.59	2586.6	1.32 $\times$
Wan2.2-A14B 720p $\times$ 81 SVOO	Default	25.36	0.853	0.089	74.25	2735.8	1.23 $\times$
	+ ALLOT	30.23	0.911	0.044	73.93	2732.2	1.23 $\times$
HunyuanVideo 720p $\times$ 129 Radial	Default	28.08	0.893	0.112	70.96	1814.2	1.76 $\times$
	+ ALLOT	27.53	0.892	0.121	70.55	1805.2	1.77 $\times$
HunyuanVideo 720p $\times$ 129 SVOO	Default	27.55	0.884	0.118	71.71	1842.7	1.75 $\times$
	+ ALLOT	30.93	0.923	0.071	71.46	1697.0	1.90 $\times$

Table 3. **Cross-model comparison** (mean-pooled top- k selection with the VSA kernel; groups differ only in density tables; Δ PSNR is the mean paired difference). Details in Appendix J.

Model & setup	Speedup	n	PSNR $\uparrow$ (uniform $\rightarrow$ ALLOT)	Δ PSNR vs. uniform	Δ PSNR vs. strengthened
Wan2.1-1.3B, 480p $\times$ 33	1.14 $\times$	32	19.59 $\rightarrow$ 25.99	+6.40	+2.55
Wan2.1-14B, 720p $\times$ 81	1.29 $\times$	16	24.87 $\rightarrow$ 31.25	+6.38	+0.76
Wan2.2-A14B, 480p $\times$ 33	1.09 $\times$	32	19.36 $\rightarrow$ 25.86	+6.50	+1.67

Table 4. **Ablation** (XAttention, Wan2.1-1.3B 480p $\times$ 81, $n=32$; one module replaced per row at the same cost as ALLOT; full metrics in Appendix I).

Module	Replaced with	PSNR	Δ PSNR
—	ALLOT	26.39	—
Cell prob.	Same density for all layers	25.32	−1.07
Traj. cal.	$w \equiv 1$	19.34	−7.05
Per-cell solve	Parametric shape rule	24.63	−1.76
Latency table	Density-proportional cost	26.27	−0.12

metrics, yet is nearly input-invariant for a fixed model and strategy. ALLOT turns these findings into a training-free allocator by calibrating one-step errors against final-video damage and optimizing a latency-constrained density table. Ablations identify trajectory calibration as the dominant component, while the same offline measurements remain effective across budgets, step counts, guidance scales and samplers without re-measurement. Across three model fami-

lies, ALLOT outperforms every compared allocation method on XAttention and outperforms or ties the default on four other published pipelines, all at equal or shorter wall-clock time.

Limitations. ALLOT requires offline calibration, and switching selection strategies requires new cell probes. Its additive, layer-separable estimate predicts rankings more reliably than error magnitudes, while the common density interface replaces native threshold adaptation. Modeling cell interactions and preserving adaptive thresholds are directions for future work.

References

- [1] Pengtao Chen, Xianfang Zeng, Maosen Zhao, Mingzhu Shen, Wei Cheng, Gang Yu, and Tao Chen. Sparse-vDiT: Unleashing the power of sparse attention to accelerate video diffusion transformers. In *AAAI*, 2026. arXiv:2506.03065.
- [2] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024. arXiv:2307.08691.
- [3] Hangliang Ding, Dacheng Li, Runlong Su, Peiyuan Zhang, Zhijie Deng, Ion Stoica, and Hao Zhang. Efficient-vDiT: Efficient video diffusion transformers with attention tile. In *ICML 2025 Workshop on Efficient Systems for Foundation Models (ES-FoMo III)*, 2025. arXiv:2502.06155.
- [4] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P. Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE TPAMI*, 44(5):2567–2581, 2022.
- [5] Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. FlexAttention: A programming model for generating fused attention variants. In *Proceedings of Machine Learning and Systems*, 2025. arXiv:2412.05496.
- [6] Hugh Everett, III. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11(3):399–417, 1963.
- [7] Huanlin Gao, Ping Chen, Fuyuan Shi, Chao Tan, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. LeMiCa: Lexicographic minimax path caching for efficient diffusion-based video generation. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2511.00090.
- [8] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [9] Jie Hu, Zixiang Gao, Yutong He, and Kun Yuan. DFSAttn: dynamic fine-grained sparse attention for efficient video generation. In *International Conference on Machine Learning*, 2026. arXiv:2605.23445.
- [10] Ziqi Huang, Yanan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [11] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.
- [12] 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. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2506.19852.
- [13] Yuhan Liu, Zongwei Hong, Jinglun Li, Linze Li, Shen Zhang, and Yao Tang. EpaCache: Error-propagation-aware caching for accelerating diffusion-based visual generation. *arXiv preprint arXiv:2608.29264*, 2026.
- [14] Jiayi Luo, Jiayu Chen, Jiankun Wang, Cong Wang, Hanxin Zhu, Qingyun Sun, Chen Gao, Zhibo Chen, and Jianxin Li. Attention sparsity is input-stable: Training-free sparse attention for video generation via offline sparsity profiling and online QK co-clustering. In *International Conference on Machine Learning*, 2026. arXiv:2603.18636.
- [15] Dmitrii Mikhailov, Aleksey Letunovskiy, Maria Kovalova, Vladimir Arkhipkin, Vladimir Korviakov, Vladimir Polovnikov, Viacheslav Vasilev, Evelina Sidorova, and Denis Dimitrov. NABLA: Neighborhood adaptive block-level attention. *IEEE Access*, 14:64655–64665, 2026. arXiv:2507.13546.
- [16] Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. OpenVid-1M: A large-scale high-quality dataset for text-to-video generation. In *International Conference on Learning Representations*, 2025.
- [17] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.
- [19] Wan Team. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.
- [20] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
- [21] Jianzong Wu, Liang Hou, Haotian Yang, Xin Tao, Ye Tian, Pengfei Wan, Di Zhang, and Yunhai Tong. VMoBA: Mixture-of-block attention for video diffusion models. In *International Conference on Learning Representations*, 2026. arXiv:2506.23858.
- [22] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, Jianfei Chen, Ion Stoica, Kurt Keutzer, and Song Han. Sparse VideoGen: Accelerating video diffusion transformers with spatial-temporal sparsity. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. arXiv:2502.01776.
- [23] Yifei Xia, Suhan Ling, Fangcheng Fu, Yujie Wang, Huixia Li, Xuefeng Xiao, and Bin Cui. Training-free and adaptive sparse attention for efficient long video generation. *arXiv preprint arXiv:2502.21079*, 2025.
- [24] Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. XAttention: Block sparse attention with antidiagonal scoring. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. arXiv:2503.16428.
- [25] 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. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2505.18875.
- [26] Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. FlashInfer: Efficient and customizable attention engine for LLM inference

- serving. In *Proceedings of Machine Learning and Systems*, 2025. arXiv:2501.01005.
- [27] Shai Yehezkel, Shahrar Yadin, Noam Elata, Yaron Ostrovsky-Berman, and Bahjat Kavar. Accelerating text-to-video generation with calibrated sparse attention. *arXiv preprint arXiv:2603.05503*, 2026.
 - [28] Zhihang Yuan, Hanling Zhang, Pu Lu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. DiTFastAttn: Attention compression for diffusion transformer models. In *Advances in Neural Information Processing Systems*, 2024.
 - [29] Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. SpargeAttention: Accurate and training-free sparse attention accelerating any model inference. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. arXiv:2502.18137.
 - [30] Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. VSA: Faster video diffusion with trainable sparse attention. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2505.13389.
 - [31] Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast video generation with sliding tile attention. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. arXiv:2502.04507.
 - [32] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
 - [33] Wentai Zhang, Ronghui Xi, Shiyao Peng, Jiayu Huang, Hao-ran Luo, Zichen Tang, and Haihong E. Ride the wave: Precision-allocated sparse attention for smooth video generation. *arXiv preprint arXiv:2604.12219*, 2026.
 - [34] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, 2023. arXiv:2302.04867.
 - [35] Xuzhe Zheng, Yuexiao Ma, Jing Xu, Xiawu Zheng, Rongrong Ji, and Fei Chao. HEART: Exploiting head heterogeneity in sparse attention for video diffusion. *arXiv preprint arXiv:2605.14513*, 2026.
 - [36] Ruiliang Zhou, Xuecheng Wu, Kang He, Guangyun Han, Bin Liu, Qinqin Chen, Wende Xu, Qingjie Zhao, and Chengru Song. ScalingAttention: Discovering intrinsic sparse attention topology for video diffusion transformers. *arXiv preprint arXiv:2606.23019*, 2026.

Sparsity Should Be ALLOTted by Output: Training-Free Allocation of Block-Sparse Attention for Video Diffusion Transformers

Supplementary Material

A. Statistical Conventions and Timing

A.1. Metrics and verdicts

Ratios of final-video error are ratios of pixel MSEs.

Verdicts. A *win* requires the lower bound of the paired PSNR interval to be > 0 and the mean LPIPS to be lower; a *loss* requires the upper bound of the PSNR interval to be < 0 and the mean LPIPS to be higher; everything else is a *tie*. The appendix tables additionally list DISTs [4] as a supplement only. Paired differences are ALLOT – baseline, followed by the 95% t -interval over prompts and the numbers of prompts on which ALLOT is better/worse.

Trajectory gain. The trajectory gain is calibrated once per model family (Wan2.1 on 1.3B 480p $\times$ 33 with mean-pooled top- k , Wan2.2-A14B, and HunyuanVideo 480p $\times$ 33), and its window experiments are shared by all tables of that family across selection strategies, resolutions and model sizes. SVG2’s official pipeline uses a different flow shift, so its window experiments are rerun on that schedule.

A.2. Timing and equal wall-clock time

Wall-clock time is the per-video median in seconds, including text encoding, denoising and VAE decoding. Speedups are relative to each pipeline’s own dense time (the dense times are in the Dense row of Table 1 and in Appendix H). ALLOT’s wall-clock time is considered matched when it is no more than $1.01\times$ the baseline’s.

B. Observation 1: Across Layers

B.1. Cell probes and attention-map metrics

With each layer sparsified alone to $b=0.30$ (mean-pooled top- k), we record the cell probe and two attention-map metrics: the density needed to cover 95% of the attention mass (SVOO’s definition) and Hellinger² (the square of ScalingAttention’s metric).

The layer rankings of the attention-map metrics agree partly with that of the cell probe on 1.3B, are essentially unrelated to it on 14B, and are reversed on HunyuanVideo. The magnitudes also differ sharply: on 1.3B the coverage density of layer 0 is only $1.36\text{--}1.84\times$ that of the median layer, while its cell probe is $53\times$, $8\times$ and $15\times$ at $t = 0.85$, 0.45 and 0.15 (at $t = 0.45$ the most sensitive layer is layer 1, at $27\times$). On 14B the coverage ratio is $0.99\text{--}1.17\times$ and the cell-probe ratio is $154\times$, $75\times$ and $71\times$. On 14B the

Spearman correlation between the cell probe and SVOO’s released per-layer densities is -0.14 ($+0.59$ on 1.3B with probes under SVOO’s selection strategy, Figure B.1).

B.2. Allocating by other cross-layer metrics

Hellinger². We replace only the solver’s cross-layer signal, using Hellinger² instead of the cell probe while leaving all other inputs unchanged. Because the solver optimizes all cells jointly, re-solving also changes the step allocation. The paired PSNR differences in dB (ALLOT minus the Hellinger²-guided allocation) are 1.3B 480p $+3.62$ [$+3.10, +4.15$] 32/0, 14B 480p $+5.88$ [$+5.13, +6.64$] 32/0, and HunyuanVideo 480p $+1.38$ [$+1.07, +1.68$] 128/32 ($n=160$: the evaluation set plus 128 VBench all-dimension prompts).

SVOO’s default density. We compare ALLOT with an allocation proportional to the actual per-cell densities produced by SVOO’s default run (hereafter the *default density*). Both use the same dense cells and the same total budget at each step, differing only in how density is distributed across layers. On HunyuanVideo 480p $\times$ 33, the paired PSNR difference in dB is $+2.40$ [$+1.93, +2.87$] 31/1; LPIPS and DISTs also favor ALLOT (31/1 each).

B.3. Per-head allocation within a cell

Each cell in the table takes a single density, and every head in that cell keeps the same fraction of its blocks. On SVOO 1.3B 720p $\times$ 81, we compare this with a per-head split that keeps each cell’s total but gives each head blocks in proportion to its density in SVOO’s default run (more blocks for heads with dispersed attention, fewer for heads with concentrated attention). ALLOT’s equal split yields a PSNR 1.44 dB higher than the per-head split ($+1.44$ [$+1.05, +1.84$] 16/0), and LPIPS and DISTs are also better (16/0 and 15/1; $n=16$).

C. Observation 2: Across Steps

C.1. Window experiments

The 50 steps are split into 10 windows of 5 steps. Each run sparsifies all layers to 0.30 within one window only, keeps the other steps dense, and uses 8 VBench prompts per window.

C.2. One-step error with all layers sparse

At every step of the first and last windows, the one-step error with all layers sparse at 0.30 is measured and averaged over

Table B.1. Cell probes and attention-map metrics. Ratio: cell probe of the most sensitive layer $\div$ that of the median layer; ranks are in decreasing order.

Model	t	Most sensitive layer	Ratio	Rank: coverage density	Rank: H^2	Spearman (coverage, $\hat{e}$)	Spearman (H^2 , $\hat{e}$)
Wan2.1-1.3B	0.85	0	53 $\times$	3/30	1/30	+0.33	+0.43
Wan2.1-1.3B	0.45	1	27 $\times$	26/30	26/30	+0.29	+0.45
Wan2.1-1.3B	0.15	0	15 $\times$	1/30	1/30	+0.33	+0.39
Wan2.1-14B	0.85	0	154 $\times$	22/40	13/40	+0.03	+0.08
Wan2.1-14B	0.45	0	75 $\times$	11/40	6/40	-0.16	-0.02
Wan2.1-14B	0.15	0	71 $\times$	7/40	4/40	-0.15	+0.11
HunyuanVideo	0.85	1	35 $\times$	49/60	47/60	-0.26	-0.20
HunyuanVideo	0.45	1	23 $\times$	51/60	48/60	-0.44	-0.36
HunyuanVideo	0.15	1	30 $\times$	48/60	42/60	-0.51	-0.42

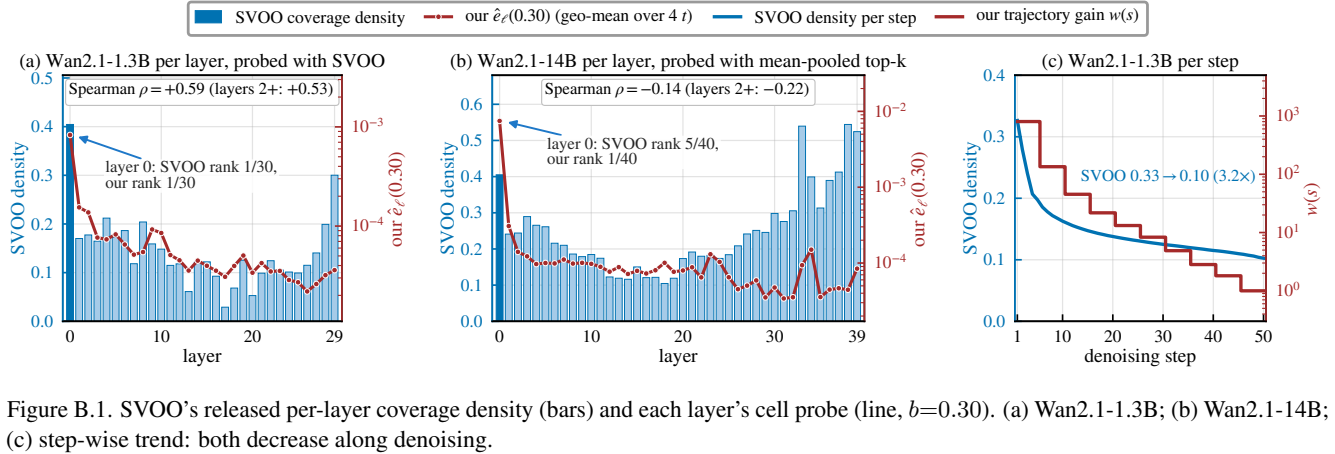


Figure B.1. SVOO’s released per-layer coverage density (bars) and each layer’s cell probe (line, $b=0.30$). (a) Wan2.1-1.3B; (b) Wan2.1-14B; (c) step-wise trend: both decrease along denoising.

Table C.1. Window experiments. Each entry: PSNR (in parentheses, final-video MSE relative to window 10, from 8-prompt mean).

Window	Steps	Wan2.1-1.3B	Wan2.1-14B	Wan2.2-A14B	HunyuanVideo
1	0–4	10.71 (463)	10.70 (186)	12.95 (109)	20.89 (69)
2	5–9	18.46 (78)	16.99 (44)	18.48 (31)	23.55 (38)
3	10–14	23.19 (26)	21.89 (14)	24.28 (8)	28.43 (12)
4	15–19	26.46 (12)	26.77 (5)	25.36 (6)	33.12 (4)
5	20–24	29.58 (6)	27.73 (4)	26.70 (5)	36.74 (2)
6	25–29	32.61 (3)	30.13 (2)	29.74 (2)	38.34 (1)
7	30–34	34.94 (2)	31.35 (2)	31.72 (1)	38.93 (1)
8	35–39	36.41 (1)	32.35 (1)	33.10 (1)	39.69 (1)
9	40–44	37.18 (1)	33.36 (1)	33.95 (1)	40.36 (1)
10	45–49	37.37 (1)	33.40 (1)	33.32 (1)	39.30 (1)
7 windows of steps 15–49: max / min MSE		12.3 $\times$	4.6 $\times$	7.2 $\times$	5.3 $\times$

the window. The first/last-window ratio is 6.4 $\times$, 2.4 $\times$, 31 $\times$ and 0.18 $\times$ on 1.3B, 14B, HunyuanVideo and Wan2.2 (on Wan2.2 the two windows belong to two different experts). The corresponding final-video ratios are 463 $\times$, 186 $\times$, 69 $\times$ and 109 $\times$, so the one-step error understates them by 72 $\times$, 78 $\times$, 2.3 $\times$ and about 600 $\times$. Measuring the amplification share as $1 - \log(\text{one-step ratio}) / \log(\text{final-video ratio})$ gives 0.70 on 1.3B and 0.83 on 14B: the gap between steps comes mainly from amplification in subsequent denoising. On HunyuanVideo it is 0.19: the gap is mostly already in the one-step error.

C.3. Local variation and attention concentration

PASA allocates steps by the L1 difference between the velocity predictions of adjacent steps (the local variation ν_s). On the dense trajectory it is U-shaped, and its Spearman correlation with the final-video error of each window is +0.02. ScalingAttention attributes the differences between steps to attention becoming increasingly concentrated along denoising. In the same runs as §C.2, the first/last-window ratio of Hellinger² is 2.72 $\times$ (0.112 / 0.041), and that of the density needed to cover 90% of the attention mass is 2.20 $\times$ (0.423 / 0.192). Both are smaller than the 6.4 $\times$ of the one-step error, and far smaller than the 463 $\times$ of the final video.

C.4. Comparison differing only in step-wise allocation

The two settings give all layers the same density in every step and have the same budget, differing only in step-wise allocation: one allocates density across steps using the trajectory gain; the other keeps the first 15 steps dense and uses a single density for all remaining steps. At the three 1.3B 480p budgets (0.20 / 0.30 / 0.45), the former is higher by +1.77 [+1.46, +2.08], +3.18 [+2.62, +3.74] and +4.46 [+3.73, +5.18] dB, each 31/1, with wall-clock time differing by $\leq 1.3\%$. On 14B 480p it is higher by +3.00 [+2.70, +3.29] 32/0 (wall-clock time 264.5 / 269.9 s). LPIPS is better in all four groups (28/4–32/0).

D. Observation 3: Across Inputs

For each video, the per-layer cell probes ($b=0.30$) are measured once separately, and their layer ranking is compared with that of the mean over the 4 probing videos. On 32 videos generated by 1.3B, the median Spearman correlation ($t=0.85/0.45/0.15$) is 0.84 / 0.91 / 0.90, close to the agreement among these videos themselves (0.93 / 0.95 / 0.95). At $t=0.85$ the most sensitive layer is layer 0 for 32/32 videos on 1.3B and 64/64 on 14B. On 64 OpenVid videos per model, excluding the 4 probing videos among them gives 0.80 / 0.87 / 0.85 on 14B and 0.91 / 0.89 / 0.81 on HunyuanVideo, within 0.01 of the values before exclusion.

Across steps, the leave-one-out Spearman correlation between each prompt’s ranking of the 10 windows and the ranking of the mean over the other prompts (on log MSE) has a median of 1.00 on 1.3B (0.99 with SparseAttn selection), 0.98 on 14B, 0.99 on Wan2.2 and 0.95 on HunyuanVideo. The first/last-window MSE ratio varies considerably across prompts (1.3B median 363 $\times$), but the ranking is stable.

E. Derivation and Verification of the Estimate

The estimate relies on two approximations, separability and additivity (precise forms in §E.1). Both are supported for rankings, but magnitudes remain approximate. The ratio “sparse together $\div$ sum of each alone” in §E.4 tests the first-order superposition and the orthogonality parts at once.

E.1. First-order error propagation

Let the sampler update of step s be $x_{s+1} = \Phi_s(x_s, v_s)$ ($s=0$ is the noisiest step). The sparse approximation adds an error ϵ_s to the velocity prediction v_s at this step. Linearizing around a reference dense trajectory, the linearized deviation satisfies $\delta_{s+1} = A_s \delta_s + B_s \epsilon_s$ (A_s is the total Jacobian of one denoising step with respect to the state, including the chain term through v_s ; B_s is the step-size coefficient). Writing $\delta \equiv \delta_N$, unrolling from $\delta_0 = 0$ yields the linear term in Eq. (1); for a one-step sampler, $P_s = A_{N-1} \cdots A_{s+1} B_s$.

The UniPC sampler [34] used for the Wan models is a multistep sampler whose update also depends on past velocities. Linearizing after including the past velocities in the state leaves the first-order representation $\delta \approx \sum_s P_s \epsilon_s$ unchanged; P_s is no longer the product above. At first order, VAE decoding replaces P_s by $J_{\text{dec}} P_s$. Below, expectations are over the probe inputs, and P_s takes its value on the reference trajectory. For a fixed model and velocity-tensor shape, the probe MSE $\hat{e}_\ell(t_s, b)$ estimates $\mathbb{E} \|\epsilon_{\ell s}\|^2$ up to a common per-element normalization, which is absorbed into $w(s)$.

Precise form of the two approximations. Additivity has two parts: within one step, the perturbations of the layers superpose at first order, $\epsilon_s = \sum_\ell \epsilon_{\ell s}$; and the errors of different cells are orthogonal in expectation, $\mathbb{E}[\epsilon_{\ell s} \epsilon_{\ell' s'}^\top] = 0$ for $(\ell, s) \neq (\ell', s')$. We impose the full cross-moment condition rather than scalar uncorrelatedness. It makes the propagated cross term $\mathbb{E}[\epsilon_{\ell s}^\top P_s^\top P_{s'} \epsilon_{\ell' s'}] = \text{tr}(P_s^\top P_{s'} \mathbb{E}[\epsilon_{\ell s} \epsilon_{\ell' s'}^\top])$ vanish for every pair of propagation operators and allows systematic bias in sparse-attention errors. Separability holds if, within one step, the error second-moment matrices of the layers differ only by scale: $\Sigma_{\ell s} \equiv \mathbb{E}[\epsilon_{\ell s} \epsilon_{\ell s}^\top] = \alpha_{\ell s} \bar{\Sigma}_s$, $\text{tr} \bar{\Sigma}_s = 1$. Then $\mathbb{E} \|P_s \epsilon_{\ell s}\|^2 = \text{tr}(P_s^\top P_s \Sigma_{\ell s}) = \alpha_{\ell s} \text{tr}(P_s^\top P_s \bar{\Sigma}_s)$, while $\mathbb{E} \|\epsilon_{\ell s}\|^2 = \text{tr} \Sigma_{\ell s} = \alpha_{\ell s}$, so their ratio $g(s) = \text{tr}(P_s^\top P_s \bar{\Sigma}_s)$ does not depend on ℓ . The isotropic case $\bar{\Sigma}_s = I/d$ (d the latent dimension) is a special case, with $g(s) = \|P_s\|_F^2/d$. Equation (2) motivates ALLOT’s factorized sensitivity estimate. Proposition 1 establishes exact optimization of the empirically calibrated surrogate objective in Eq. (4) at the attained budget. A sufficient condition for separability is that the layers at a given step share the same normalized error second moment, while P_s can remain anisotropic. This distinction matters because low-frequency velocity perturbations have a larger effect on the final video (Table 1 of HEART [35]).

The gain $g(s)$ therefore depends on both the trajectory and the shared error structure. ALLOT captures this step dependence with a trajectory gain calibrated directly from sparse-window generations, setting $w(s) = M_k/E_k$ for $s \in \Omega_k$. The calibration also captures average cross-layer interactions and latent-to-pixel conversion. The oracle diagnostic in §E.7 shows why the actual sparse error is used: energy-preserving or energy-matched transformations produce different final-video errors. The resulting allocation is evaluated in §E.5 and §5.5.

E.2. Proof of Eq. (2) and Proposition 1

Eq. (2). Under additivity and separability, the squared energy of the linearized deviation takes the form in Eq. (2), with $g(s)$ depending only on the step s . Let $\delta^{(1)} = \sum_{\ell, s} P_s \epsilon_{\ell s}$ denote the linearized deviation in Eq. (1). Then

$$\mathbb{E} \|\delta^{(1)}\|^2 = \sum_{(\ell, s), (\ell', s')} \text{tr}(P_s^\top P_{s'} \mathbb{E}[\epsilon_{\ell' s'} \epsilon_{\ell s}^\top]).$$

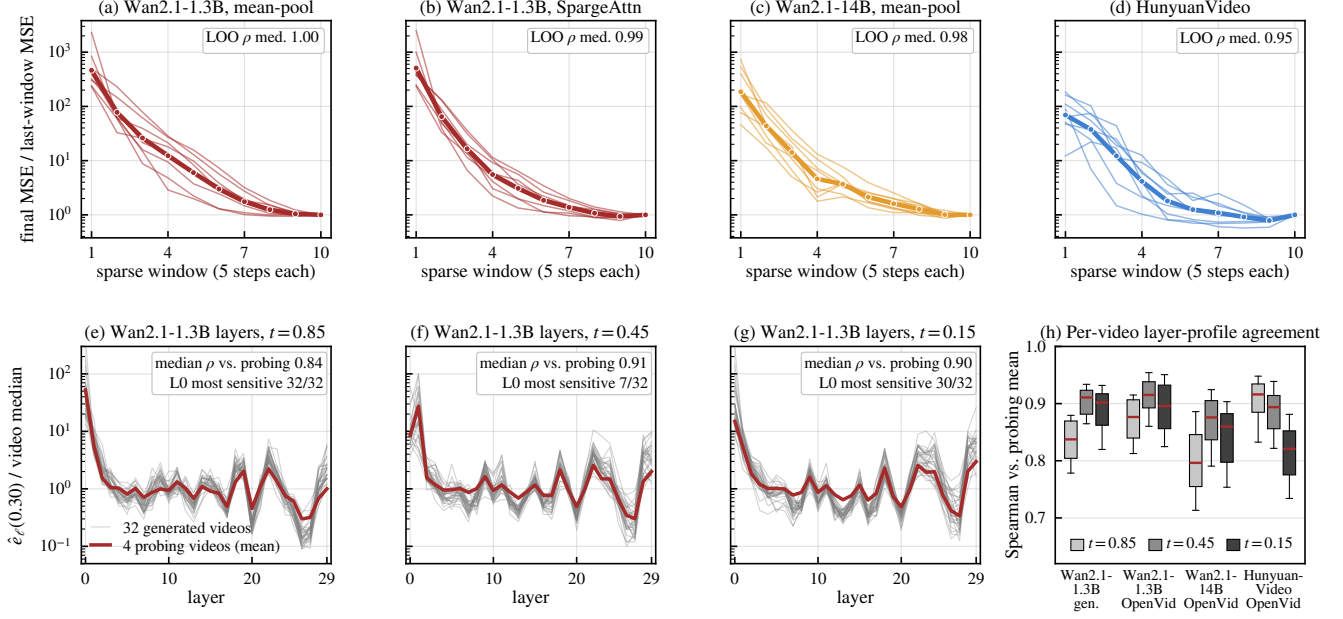


Figure D.1. Sensitivity is stable across inputs. (a–d) Final-video MSE when only one window is sparsified (8 prompts each; boxes: median leave-one-out Spearman correlation). (e–h) Per-video cell probes and layer-ranking Spearman on 1.3B and OpenVid videos.

By orthogonality the terms with $(\ell, s) \neq (\ell', s')$ vanish, leaving $\sum_{\ell, s} \text{tr}(P_s^\top P_s \Sigma_{\ell s})$; by separability, $\Sigma_{\ell s} = \alpha_{\ell s} \bar{\Sigma}_s$, so each term equals $\alpha_{\ell s} \text{tr}(P_s^\top P_s \bar{\Sigma}_s) = g(s) \mathbb{E} \|\epsilon_{\ell s}\|^2$, using $\mathbb{E} \|\epsilon_{\ell s}\|^2 = \text{tr} \Sigma_{\ell s} = \alpha_{\ell s}$. Substituting gives Eq. (2) at first order. ■

Proposition 1. Let b^λ minimize $\hat{S}(\ell, s; b) + \lambda r(b)$ per cell over the candidate set $\mathcal{B}$, with $\lambda \geq 0$. Write $S_{\text{tot}}(b) = \sum_{\ell, s} \hat{S}(\ell, s; b_{\ell s})$ and $r_{\text{tot}}(b) = \sum_{\ell, s} r(b_{\ell s})$. For any table b with values in $\mathcal{B}$ and $r_{\text{tot}}(b) \leq r_{\text{tot}}(b^\lambda)$, per-cell optimality gives $S_{\text{tot}}(b) + \lambda r_{\text{tot}}(b) \geq S_{\text{tot}}(b^\lambda) + \lambda r_{\text{tot}}(b^\lambda)$; rearranging,

$$S_{\text{tot}}(b) \geq S_{\text{tot}}(b^\lambda) + \lambda [r_{\text{tot}}(b^\lambda) - r_{\text{tot}}(b)] \geq S_{\text{tot}}(b^\lambda).$$

Thus b^λ is the exact discrete optimum of Eq. (4) over $\mathcal{B}$ at the attained budget $r_{\text{tot}}(b^\lambda)$ [6]. ■

Gap for a given budget C . With discrete candidate densities, not every budget is exactly attainable. When the bisection terminates, it has a table $b^{\lambda_{\text{hi}}}$ whose cost does not exceed C and a table $b^{\lambda_{\text{lo}}}$ whose cost exceeds C (§F.4). Any table b with cost $\leq C$ also satisfies $r_{\text{tot}}(b) < r_{\text{tot}}(b^{\lambda_{\text{lo}}})$, so by Proposition 1 $S_{\text{tot}}(b) \geq S_{\text{tot}}(b^{\lambda_{\text{lo}}})$; the optimum under budget C therefore lies between $S_{\text{tot}}(b^{\lambda_{\text{lo}}})$ and $S_{\text{tot}}(b^{\lambda_{\text{hi}}})$. Algorithm 1 returns one of the two: $b^{\lambda_{\text{hi}}}$ is within their difference of this optimum, and $b^{\lambda_{\text{lo}}}$ is no worse than it but exceeds C ; the two tables usually differ in only one cell. On 15 tables used in the comparisons of the main text and Appendices G–H, this difference is 0.002%–0.5% of the estimated error.

E.3. Final video with a single layer sparse over all steps

Each run sparsifies one of six representative layers over all 50 steps while keeping the other layers dense. This tests the layer ranking predicted by $\sum_s w(s) \hat{e}_\ell(t_s, b)$. The measured ranking matches the prediction exactly on 1.3B and HunyuanVideo (Spearman 1.00) and nearly matches it on 14B (0.94), where only layers 19 and 39 swap.

On HunyuanVideo, coverage density instead ranks layers 54 and 58 as the two most sensitive, although sparsifying them yields 11.83 and 15.93 dB higher PSNR than sparsifying layer 1.

E.4. Separability and additivity

Table E.1 summarizes the tests. The ratio “sparse together ÷ sum of each alone” tests the first-order superposition and the orthogonality parts at once.

Direct layer-group check. On 1.3B, we sparsify one of three disjoint layer groups within a five-step window and divide each video’s final MSE by the group’s measured injected energy summed over that window ($n=8$; paired log-ratio intervals). The injected energy is the mean MSE of the two CFG branches before guidance, summed over the perturbed steps. At density 0.05, the groups are layer 0, layers 1 and 19 (plus layer 22 in window 6), and all remaining layers; layer 0 in window 6 produces too little final error for a stable ratio and is excluded. In windows 1–2, sparsifying layer 0 at density 0.30 drives the final-video error toward satura-

Table E.1. Tests of the two approximations.

Test	Reading	Conclusion
Separability: single layer sparse over all steps (§E.3)	ranking of the 6 layers identical on 1.3B and HunyuanVideo, Spearman 0.94 on 14B; ratio between the most and the least sensitive layer predicted $9.3\times$, measured $4.8\times$ (1.3B), predicted $146\times$, measured $7.7\times$ (14B), predicted $286\times$, measured $30\times$ (HunyuanVideo)	ranking holds, magnitude compressed
Layer groups within one window: final MSE per unit of measured injected energy	at density 0.05, largest/smallest group ratio 2.17 [1.89, 2.50] in window 4 and 1.25 [0.91, 1.71] in window 6; energy-matched layer 0/comparison-group ratios in windows 1–2 are 0.56–0.82 ($n=8$)	point estimates within $2\times$ in window 6 and the energy-matched early windows; $2.17\times$ in window 4 approximately additive
Across steps: two windows sparse together $\div$ sum of each alone	over 7 window pairs, 1.3B 0.86–1.26 and 14B 0.83–1.36; adjacent windows within steps 15–34 super-additive by $1.16\text{--}1.36\times$ (4 of 6 pairs at $p \leq 0.04$, $n=8$)	
Across layers: all layers sparse together $\div$ sum of each layer alone	1.3B 6.5 at high noise to 1.1 at low noise, 14B 1.9 to 0.8, HunyuanVideo 0.74–0.89 to 1.2–1.5; for two layers sparse together within one step, median ratio 1 ± 0.08	not additive in magnitude; absorbed by w
Additive prediction of isolated probes vs. marginals measured in context	Spearman: 1.3B 0.63 in the deployed table’s context and 0.36 in a uniform 0.15 context; HunyuanVideo 0.77 in the deployed table’s context	less accurate in sparser contexts
Additive prediction vs. final video for arbitrary tables (80 tables, §E.5)	Kendall τ : pooled 0.83 (R^2 0.90), within one budget 0.78 (per budget 0.70 / 0.77 / 0.86); by family of tables: convex combinations 0.91, jitters 0.82, random tables 0.72, log–log slopes 0.98–1.14 / 0.48–0.75 / 0.26–0.44	rankings within one budget usable; weaker for tables further from the solved ones

tion, so we raise its density to 0.40–0.50 to bring its injected energy within $1.8\times$ of the comparison groups. In point estimates, the measured ratios of final MSE to injected energy differ by less than $2\times$ in window 6 and the energy-matched early-window comparisons; window 4 shows a $2.17\times$ group spread. The test covers only these groups and densities.

E.5. Prediction vs. measurement for arbitrary tables

On 1.3B 480p $\times$ 33 we construct 80 tables (log-convex combinations of ten existing tables at the deployed budget, random tables with a log-uniform density per block of 5 layers $\times$ 5 steps, and log-normal jitters of the solved table), run each on 4 held-out prompts, and compare the additive-model prediction with the final-video pixel MSE.

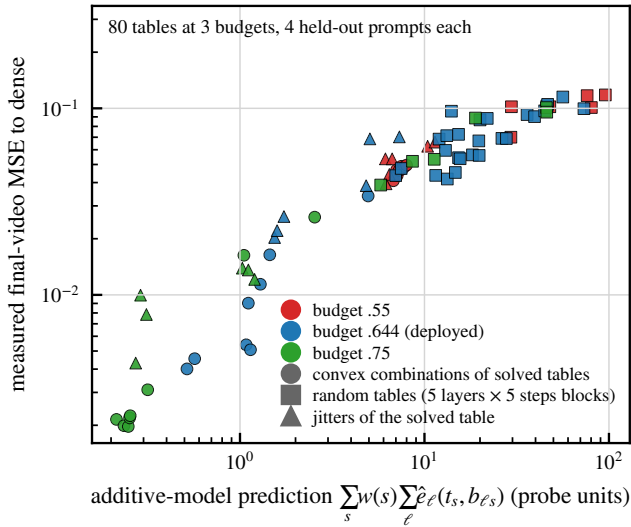


Figure E.1. Additive-model prediction vs. final-video MSE for 80 tables at three budgets (color: budget; marker: family of the table; 4 held-out prompts each). Kendall τ per budget and pooled in the last row of Table E.1.

E.6. Reassigning the same densities

The setting is the same as in §I.4 (1.3B 480p $\times$ 33, mean-pooled top- k , VSA, $n=32$). The densities of ALLOT’s table are reassigned unchanged: the multiset is the same and the cost identical. Only which cell gets which density changes. We test four types of reassignment, yielding five reassigned tables: shuffling layers within each step, shuffling steps within each layer, fully random reassignment with two seeds, and assignment using the reversed estimate (cells with higher estimated sensitivity receive lower density).

All five reassignments are 0/32 on all three metrics, and the measured order matches the order predicted by the additive model (Kendall $\tau = 1.0$). Shuffling layers within each step is still 3.40 dB lower, of the same order as the drop from “same density for all layers” (−3.22 dB), showing that the layer ranking has an effect of its own.

E.7. Oracle diagnostic of trajectory calibration

Protocol. To test whether a generic perturbation reproduces the downstream effect of the actual sparse-attention error, we modify that error within one five-step window. At each run’s current latent state x_s , the sampler receives

$$v_s^{\text{dense}}(x_s) + T_s(\epsilon_s(x_s)), \quad \epsilon_s(x_s) = v_s^{\text{sparse}}(x_s) - v_s^{\text{dense}}(x_s). \quad (6)$$

Identity $T_s(\epsilon) = \epsilon$ reproduces the ordinary sparse-window run. A random-sign transform independently flips each element of ϵ_s , preserving its magnitude, while a Gaussian control replaces ϵ_s with independent noise matched to its mean-square magnitude after classifier-free guidance. All other steps remain dense. These oracle interventions require the dense velocity and need not correspond to realizable attention masks; ALLOT itself uses only ordinary sparse-window generations. Because each transform is applied at its run’s current state, the experiment measures end-to-end responses to changes in sparse-error structure rather than isolated linear propagation gains. The top- k and fixed sliding-window masks have densities 0.30 and 0.304, respectively.

Table E.2. Reassigning the same densities (Δ = row – ALLOT).

Assignment	PSNR	Δ PSNR	Additive-model prediction
ALLOT	25.99	–	0.37
Shuffle layers within each step	22.60	–3.40 [–3.94, –2.86] 0/32	1.07
Shuffle steps within each layer	13.77	–12.23 [–13.60, –10.85] 0/32	10.6
Fully random (seed 1)	10.21	–15.78 [–17.48, –14.09] 0/32	45.6
Fully random (seed 0)	9.82	–16.17 [–17.85, –14.49] 0/32	56.1
Reversed estimate	7.95	–18.05 [–19.65, –16.44] 0/32	146

Table E.3. Oracle diagnostic: mean final-video PSNR (dB) to each model’s dense reference. All rows use 8 prompts, 480p×33, 50 steps and seed 0; windows 1 and 5 cover steps 0–4 and 20–24. Only the indicated window is perturbed.

Model and mask	Window	Sparse	Random signs	Gaussian
1.3B, top- k	1	10.71	29.57	25.68
1.3B, top- k	5	29.58	38.55	38.64
1.3B, sliding window	1	9.16	28.23	26.66
1.3B, sliding window	5	26.21	38.23	38.48
14B, top- k	1	10.70	25.93	26.54
14B, top- k	5	27.73	37.17	37.45

Result. Across all six settings, both oracle controls yield higher PSNR to the dense reference than the ordinary sparse run for every one of the eight prompts, spanning both model sizes and both mask types (Table E.3). This consistent direction shows that the downstream response depends on the perturbation pattern, supporting the use of actual sparse-window errors when measuring M_k . The layer-group and independent-table tests in §E.4–E.5 evaluate weight sharing and allocation ranking separately.

Scope. The diagnostic spans six Wan2.1 settings across two model sizes, two mask types and two windows, with eight prompts per setting. Corresponding interventions on Wan2.2 and HunyuanVideo remain untested. The first window is nonlinear and approaches saturation, so these results characterize pattern sensitivity rather than a universal propagation gain.

F. Method Details: Cell Probes, In-Context Check, Latency Tables, Solving and Integration

F.1. Cell probes: definition, grid and interpolation

Cell probe $\hat{e}_\ell(t, b)$: only layer ℓ is sparsified to density b with the selection strategy used at deployment, the other layers stay dense, one teacher-forced forward pass is run at noise level t , and the MSE between the velocity prediction and that of the dense model is taken. The inputs are the VAE latents of 4 OpenVid real videos with noise added at level t . Different selection strategies have a layer-ranking

correlation of only 0.48–0.83 at the same nominal density, so probes are by default measured under the selection strategy used at deployment.

The mean-pooled top- k probes are measured at 8 densities $(0.05\text{--}0.60) \times 9$ noise levels $(0.85\text{--}0.05)$. They are interpolated onto the candidate densities and steps as in line 2 of Algorithm 1 (§F.4).

F.2. In-context check

The cell probes are measured with the other layers dense, whereas at deployment the other layers are also sparse. The in-context check examines this difference. With all layers at a uniform $b_0=0.30$ as the context, a group of layers (layer 0, layer 1, layers 0+1, or one depth band) is restored to dense, and the actual reduction of this step’s velocity error is divided by the sum of this group’s cell probes (8 OpenVid videos including the 4 probing videos, one value per noise level):

$$\kappa = \frac{e(\text{all layers at } b_0) - e(\text{this group dense, others at } b_0)}{\sum_{\ell \in \text{group}} \hat{e}_\ell(t, b_0)},$$

where $e(\cdot)$ is this step’s one-step error under the setting in parentheses. $\kappa \approx 1$ means the isolated measurement still holds in context; $\kappa \ll 1$ means the probes overestimate the role of this group in context, and $\kappa \gg 1$ means that they underestimate it.

κ is far from 1, in model-specific ways: layer 0 is at 1.9–15.5 on Wan2.1-1.3B for $t \geq 0.35$ and at 0.09–0.44 in the high-noise expert of Wan2.2, and on HunyuanVideo restoring layers 0+1 removes at most 25% of the in-context error. We tested the correction below on final videos. It raises PSNR on HunyuanVideo (+0.46 [+0.23, +0.69] 100/60, $n=160$; LPIPS and DISTS not significant) but lowers it by 6.09 dB on Wan2.1-1.3B and by 0.78 dB on both Wan2.2 experts (0/32 each). Since its effect does not carry over across models, no table in the paper is corrected. Against the uniform baseline at equal density ($n=160$), the uncorrected HunyuanVideo table gives a non-significant +0.27 [–0.04, +0.57] 95/65 and the corrected one +0.73 [+0.47, +0.99] 111/49.

Correction. Each layer takes the κ of its group, $c_\ell(t)$: layers 0 and 1 their own, and every other layer that of its

depth band, the model’s layers being cut into six bands of consecutive layers. c_ℓ is clipped to $[0.25, 8]$ (a negative κ , where restoring the group increases the error, counts as 0.25) and rescaled within each noise level so that $\sum_\ell c_\ell \hat{e}_\ell(t, b_0) = \sum_\ell \hat{e}_\ell(t, b_0)$. It is then interpolated log-linearly in t onto each step and multiplies the cell probes at every candidate density, $\hat{e}'_\ell(t_s, b) = c_\ell(t_s) \hat{e}_\ell(t_s, b)$. It thus changes only the layers’ shares within one step: each step’s total remains calibrated by the trajectory gain, and the latency table is unchanged.

F.3. Latency tables of each path

The latency table $r(b)$ is the time of one attention call at density b divided by that of dense FlashAttention on the same shape. In the solver, r is clipped to 1.

Only changing the kernel moves the crossover. Cells above the crossover are set to dense, and assuming cost proportional to density or FLOPs would overestimate the benefit of sparsity.

F.4. Solving: Algorithm 1, candidate densities and density cap

Algorithm 1. Latency-constrained density allocation

Input: cell probes $\hat{e}_\ell(t, b)$ (measured noise levels T , densities D), trajectory gain $w(s)$, latency table $r(b)$, budget C , density cap $b_{\max}$.

- 1: Candidate densities $\mathcal{B}$: 200 uniform points on $[0.05, \min(b_{\max}, 0.99)]$, $\cup \{1\}$.
- 2: For each cell (ℓ, s) : enforce $\hat{e}$ non-decreasing from high to low density; interpolate linearly in $\log b - \log \hat{e}$ onto $\mathcal{B}$, and linearly in t onto t_s (endpoint values when t_s is outside T).
- 3: $\hat{S}(\ell, s; b) = w(s) \cdot \hat{e}_\ell(t_s, b)$, with 0 for dense ($b=1$); $R(b) = \min(r(b), 1)$, with 1 for dense.
- 4: Take $\lambda_{\text{lo}}, \lambda_{\text{hi}}$ (initially $[10^{-12}, 10^3]$) such that the total cost is above and below C , respectively.
- 5: Repeat 80 times: $\lambda = \sqrt{\lambda_{\text{lo}} \lambda_{\text{hi}}}$;
- 6: for every cell independently, $b^*(\ell, s) = \arg \min_{b \in \mathcal{B}} \hat{S}(\ell, s; b) + \lambda R(b)$;
- 7: if $\sum_{\ell, s} R(b^*(\ell, s)) > C$ then $\lambda_{\text{lo}} \leftarrow \lambda$, else $\lambda_{\text{hi}} \leftarrow \lambda$.
- 8: Return the table b^* at $\lambda = \sqrt{\lambda_{\text{lo}} \lambda_{\text{hi}}}$ (its cost may exceed C by one marginal cell).

$b_{\max}$ is the highest probed density of the selection strategy (0.57–0.90); on HunyuanVideo at equal density (§J.1) it is the uniform density, 0.30.

F.5. Integration

ALLOT keeps the selection strategy’s scoring and ranking, and the kernel is the same as the baseline’s (XAttention’s official kernel is replaced by FlashInfer on both sides, see §G.2; for Radial’s kernel paths see §H.3); only each cell’s cutoff is set to the density in the table.

- **XAttention:** keeps blocks by block score up to the table’s density, replacing its per-head cumulative threshold.
- **SVOO:** keeps key clusters in order of centroid score until the fraction of key tokens reaches the table value, bypassing its top- p threshold, its per-(layer, head) density clamping and its dense warm-up steps and first layer.

- **SVG2:** likewise keeps key clusters in its ranking order until the fraction of key tokens reaches the table value, bypassing its top- p threshold, its minimum key-cluster ratio and its first-steps dense rule.
- **SpArgeAttn:** uses its built-in top- k entry.
- **Radial:** takes the nearest reachable `decay_factor` that does not exceed the table value.

Cells marked dense in the table all use the exact dense path.

Effect of density truncation. The effect of replacing threshold truncation with density truncation depends on the selection strategy. On the SVOO 480p×33 pipeline, we preserve its default per-cell densities and dense-cell rules (§B.2) and change only the truncation rule. This lowers PSNR by 3.00 dB on 1.3B (−3.00 [−3.53, −2.46] 0/32) and by 1.96 dB on HunyuanVideo (−1.96 [−2.69, −1.24] 4/28).

For XAttention, density truncation has the opposite effect. At equal wall-clock time, ALLOT with density truncation exceeds its threshold-based counterpart by 4.50 dB (+4.50 [+4.00, +5.01] 32/0). The threshold-based variant assigns each cell a cumulative-mass threshold with the same cost as truncating that cell at its mean density. For a head with concentrated attention, a low threshold may retain only one or two blocks. Replacing each threshold in XAttention’s default allocation with the density it induces raises PSNR by 2.27 dB (+2.27 [+1.69, +2.85] 30/2) at 1.9% more wall-clock time. Even against this strengthened default, ALLOT remains 5.98 dB higher at equal wall-clock time (+5.98 [+5.46, +6.50] 32/0; §G.2).

F.6. Offline cost

Table F.1. Offline cost per model (GPU-h, including model loading).

Model	Cell probes	Trajectory gain	Latency table	Total
Wan2.1-1.3B	2.8	3.1	minutes	≈6
Wan2.1-14B	17.4	shared with 1.3B	minutes	≈17
HunyuanVideo	18.9	3.8	minutes	≈23

On Wan2.2-A14B the cell probes of the two experts take 26 GPU-h and its own trajectory gain 6.5 GPU-h of generation. When switching the selection strategy, we re-measure the cell probes (3.0 GPU-h each for SVOO and SpArgeAttn on 1.3B, 22.6 GPU-h for Radial on the 720p latent, and 57 GPU-h for Radial on HunyuanVideo) and the latency table (minutes), and reuse the trajectory gain measured with mean-pooled top- k (Appendix K).

G. Comparison with Other Allocation Methods

G.1. Paired intervals

Table G.1 gives the paired differences in all four metrics against the default and HEART at 480p×81 (Table 1) and

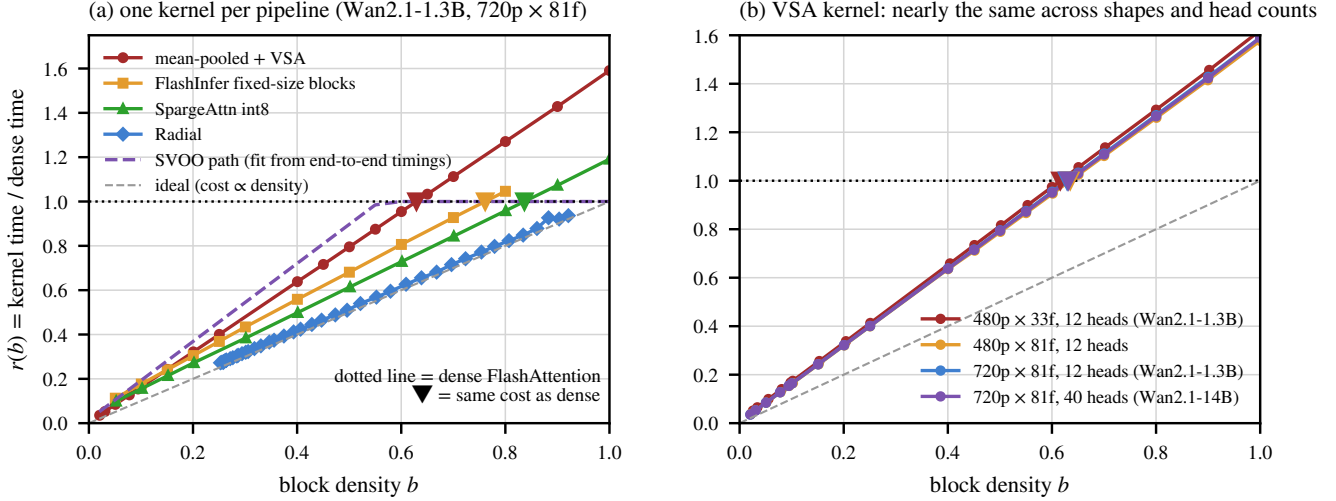


Figure F.1. Latency tables. (a) Kernels of each pipeline on Wan2.1-1.3B 720p×81; ▼ marks the density at which cost equals dense. (b) The VSA kernel is nearly identical across shapes and head counts.

720p×81 (§G.4), with verdicts as in Appendix A; the other baselines are compared in Table G.2.

G.2. XAttention and HEART

Setting. Following the setting of HEART’s ablation [35] (Wan2.1-1.3B, 832×480×81, 50 steps, classifier-free guidance (CFG [8]) scale 5, UniPC, $n=32$), all settings keep the first 5 steps dense. Selection uses the official block estimation [24]. Attention uses the FlashInfer fixed 128-block block-sparse kernel [26]. On real inputs the two produce identical block masks. The budget is the cost of the default setting under the same latency table, 0.734 of dense.

HEART’s EBC. We re-implement EBC from HEART’s description, without TMR (cross-step mask reuse): only one head at a time is sparsified to one of the thresholds {0.85, 0.90, 0.95}, one step is sampled from each of four step segments to measure the frequency-weighted error of the CFG-combined velocity, and an integer program then picks one threshold per head such that the mean actual sparsity is no lower than that of a shared threshold of 0.90. Calibration uses 10 prompts that do not overlap with the evaluation set. EBC gives +1.29 [+1.00, +1.57] 32/0 over the default.

HEART (EBC + TMR). Following HEART’s §4.2 and Algorithm 7.3, we implement TMR, which reuses masks across steps: for each layer, CFG branch and head, the token-wise means of query and key at the last re-scoring are stored together with that step’s block scores and mask; if the sum of the L1 drifts of the two at the current step does not exceed $\delta_{\text{TMR}}=30$ (HEART’s default), the previous result is reused, otherwise the head is re-scored. When fewer than 40% of a layer’s heads need re-scoring, the whole layer reuses its

masks; when more than 80% need it, the whole layer is re-scored; otherwise, heads are handled individually (HEART’s values).

ScalingAttention (FAST). On EBC’s 10 calibration trajectories, FAST measures, for each layer, head and candidate density, the attention mass within the selected blocks and converts it to a Hellinger distance. A three-parameter fidelity surface $A = \beta[1 - (1 - x)^\gamma]$ then sets a fidelity requirement for each (step, layer), where $x = (\ell' + \omega t')/(1 + \omega)$ and $\ell', t' \in [0, 1]$ are the normalized layer depth and noise level (1 for shallow layers and early steps), so the requirement is higher for shallow layers and early steps, following the description in its main text; each head takes the sparsest density meeting it. γ and ω are grid-searched, β is set by bisection to meet the budget, and the combination maximizing the minimum fidelity is selected ($\gamma=4$, $\omega=1$). $\gamma=4$ lies on the edge of the grid. Extending γ to 64, the table selected by the same criterion is 1.94 dB lower in PSNR, so $\gamma=4$ is kept. Deviations from the original: its static topology and kernel are not reproduced, and a 5-step warm-up is added. ALLOT is +3.63 [+3.04, +4.23] 31/1.

PASA. PASA keeps the first 10 steps dense, then gives each step one density shared by all layers, proportional to the local variation ν_s , with the coefficient set by bisection to meet the budget. ALLOT is +4.47 [+4.14, +4.81] 32/0.

CalibAtt-style and DFSAttn-style rules. The exponential threshold curve takes $\theta_\infty=0.842$ and $\theta_0=0.99$ from the appendix of CalibAtt [27]; the three levels of DFSAttn [9] are multiplied by one common factor of 1.52 to match the

Table G.1. Paired differences (ALLOT – baseline) for the XAttention rows default, EBC and HEART (EBC + TMR) of Table 1 and XAttention on Wan2.1-1.3B 720p×81 (§G.4); verdicts follow §A.1.

Setting	Baseline	n	Δ PSNR	Δ SSIM	Δ LPIPS	Δ DISTS	Verdict
XAttention 1.3B 480p×81 (ALLOT at the default’s wall-clock time)	default	32	+7.23 [+6.60, +7.85] 32/0	+0.176 [+0.148, +0.204] 32/0	−0.163 [−0.187, −0.138] 32/0	−0.059 [−0.073, −0.044] 31/1	win
XAttention 1.3B 480p×81	HEART (EBC)	32	+6.96 [+6.35, +7.56] 32/0	+0.155 [+0.127, +0.183] 32/0	−0.132 [−0.154, −0.111] 32/0	−0.045 [−0.055, −0.035] 31/1	win
XAttention 1.3B 480p×81 (TMR on both sides)	HEART (EBC + TMR)	32	+6.59 [+5.99, +7.19] 32/0	+0.142 [+0.115, +0.168] 32/0	−0.120 [−0.141, −0.099] 32/0	−0.040 [−0.050, −0.030] 31/1	win
XAttention 1.3B 720p×81 (ALLOT at the default’s wall-clock time)	default	32	+6.35 [+5.61, +7.09] 32/0	+0.110 [+0.092, +0.127] 32/0	−0.141 [−0.165, −0.117] 32/0	−0.062 [−0.078, −0.046] 29/3	win
XAttention 1.3B 720p×81 (ALLOT at the default’s wall-clock time)	HEART (EBC)	32	+5.16 [+4.40, +5.92] 32/0	+0.079 [+0.065, +0.093] 32/0	−0.095 [−0.114, −0.076] 31/1	−0.032 [−0.045, −0.019] 27/5	win
XAttention 1.3B 720p×81 (TMR on both sides)	HEART (EBC + TMR)	32	+5.21 [+4.44, +5.98] 32/0	+0.083 [+0.070, +0.097] 32/0	−0.100 [−0.120, −0.079] 31/1	−0.035 [−0.048, −0.022] 28/4	win

budget.

DiTFastAttn-style rule. Following Algorithm 1 of DiTFastAttn [28], each step after the warm-up visits the layers in order and gives layer ℓ the sparsest density for which the sum of the cell probes chosen so far in that step stays below $\delta_{\text{DFA}}(\ell+1)/L$, and dense otherwise. The probes and latency table are ALLOT’s, without the trajectory gain, and δ_{DFA} is set by bisection to meet the budget.

VBench. The VBench column of Table 1 is the weighted composite of the 6 dimensions used by HEART, with VBench’s official normalization and weights. ALLOT is +0.30 [+0.004, +0.59] 22/10 over the original FAST and +0.13 [−0.11, +0.38] 20/12 over the original PASA, +0.18 [+0.09, +0.27] 24/8 over the DFSAttn-style rule, and ALLOT + TMR is −0.28 [−1.19, +0.62] relative to HEART (EBC + TMR); all remaining intervals contain zero except DiTFastAttn-style (−0.89 [−1.76, −0.02]).

Held-out set. Rerunning these settings except TMR on 32 held-out prompts leaves every verdict unchanged. ALLOT’s PSNR gains over EBC, Efficient-vDiT, the default and the CalibAtt-style rule range from 5.81 to 7.33 dB (all

32/0). Its PSNR gains in dB are +3.76 [+2.96, +4.55] 31/1 over FAST, +4.57 [+4.11, +5.03] 32/0 over PASA, and +1.48 [+1.20, +1.75] 31/1 over the DFSAttn-style rule. LPIPS and DISTS are better in every row (W/L 28/4–32/0).

G.3. XAttention on HunyuanVideo

Setting. This block follows the HunyuanVideo setting of HEART’s Table 2, also used by ScalingAttention (HunyuanVideo-T2V, 480p×129 with frames of 720×480, 50 steps, $n=32$). Selection and kernel are those of §G.2, applied to the concatenated [video | text] sequence, and all settings keep the first 5 steps dense. EBC is calibrated as in §G.2 on 10 HunyuanVideo prompts that do not overlap with the evaluation set. FAST follows §G.2 on the same 10 calibration trajectories (selected $\gamma=4$, $\omega=2$) and, as on Wan, uses XAttention’s masks rather than ScalingAttention’s static topology and kernel. ALLOT’s table is solved from HunyuanVideo’s mean-pooled probes, the same trajectory gain as §J.1 and the latency table of this XAttention path, targeting the default’s wall-clock time.

Results (Table G.3). ALLOT runs at the default’s wall-clock time (481.4 / 481.3 s) and is 0.3% and 2.6% faster than EBC and FAST. In PSNR it is +11.84 [+10.85, +12.83] 32/0, +10.88 [+9.87, +11.88]

Table G.2. XAttention allocation settings ($n=32$; shared latency table and budget). Top: construction and absolute metrics. Bottom: paired differences, $\Delta = \text{ALLOT} - \text{row}$. Default and CalibAtt-style use ALLOT at the default’s time; HEART (EBC + TMR) uses ALLOT + TMR.

Setting	Construction	Dense cells	Time (s)	PSNR	SSIM	LPIPS	DISTS
Dense	–	1500	183.8	–	–	–	–
Default	5 dense steps; threshold 0.90	150	148.7	18.14	0.659	0.268	0.154
Default, density cutoff	Default per-cell densities; density cutoff	150	151.5	20.41	0.745	0.181	0.111
HEART (EBC)	Per-head thresholds; integer programming	150	151.7	19.43	0.699	0.221	0.130
HEART (EBC + TMR)	EBC + cross-step mask reuse	150	145.0	19.54	0.706	0.214	0.128
Efficient-vDiT	Per-layer density; layers 0–2 dense	285	151.1	19.08	0.702	0.225	0.133
DiTFastAttn-style [28]	Greedy per step; error cap $\delta_{\text{DFA}}(\ell+1)/L$, else dense	208	152.2	19.38	0.708	0.219	0.130
ScalingAttention (FAST)	Hellinger profile; three-parameter surface; per-head density	159	152.9	22.75	0.790	0.141	0.117
PASA	10 dense steps; per-step density $\propto \nu_s$	360	151.8	21.91	0.768	0.163	0.117
CalibAtt-style	Exponential threshold (§G.2); $\beta=2.24$	150	149.6	19.67	0.700	0.222	0.140
DFSAttn-style [9]	12 dense steps; decrease every 12 steps	360	151.3	24.86	0.834	0.107	0.096
ALLOT	Solved	587	151.3	26.39	0.854	0.089	0.085
ALLOT + TMR	ALLOT’s table + mask reuse	587	146.0	26.13	0.847	0.094	0.089
ALLOT (default time)	Solved at default time	521	148.6	25.37	0.835	0.106	0.096

Setting	Δ PSNR	LPIPS W/L	DISTS W/L	Verdict
Dense	–	–	–	–
Default	+7.23 [+6.60, +7.85] 32/0	32/0	31/1	win
Default, density cutoff	+5.98 [+5.46, +6.50] 32/0	32/0	30/2	win
HEART (EBC)	+6.96 [+6.35, +7.56] 32/0	32/0	31/1	win
HEART (EBC + TMR)	+6.59 [+5.99, +7.19] 32/0	32/0	31/1	win
Efficient-vDiT	+7.31 [+6.74, +7.88] 32/0	32/0	32/0	win
DiTFastAttn-style [28]	+7.01 [+6.55, +7.47] 32/0	32/0	32/0	win
ScalingAttention (FAST)	+3.63 [+3.04, +4.23] 31/1	31/1	30/2	win
PASA	+4.47 [+4.14, +4.81] 32/0	32/0	32/0	win
CalibAtt-style	+5.69 [+5.17, +6.21] 32/0	32/0	30/2	win
DFSAttn-style [9]	+1.53 [+1.26, +1.80] 31/1	30/2	28/4	win
ALLOT	–	–	–	–
ALLOT + TMR	–	–	–	–
ALLOT (default time)	–	–	–	–

Table G.3. XAttention on HunyuanVideo 480p×129 (720×480; $n=32$; speedup = dense wall-clock / row wall-clock). Paired differences in §G.3.

Allocation	PSNR↑	SSIM↑	LPIPS↓	VBench↑	Wall-clock (s)	Speedup
Dense	–	–	–	70.99	595.4	1.00×
Default (threshold 0.90)	15.65	0.568	0.476	65.62	481.3	1.24×
HEART (EBC)	16.61	0.608	0.428	67.20	483.0	1.23×
ScalingAttention (FAST)	20.33	0.705	0.328	67.72	494.4	1.20×
ALLOT	27.49	0.875	0.122	69.96	481.4	1.24×

32/0 and +7.16 [+6.20, +8.12] 32/0 above the default, EBC and FAST. SSIM, LPIPS and DISTS are better on all 32 prompts in all three comparisons. On VBench (same accounting as §G.2) it is +4.34 [+2.40, +6.28], +2.75 [+1.15, +4.36] and +2.24 [+1.44, +3.03] above the three.

G.4. XAttention on 1.3B 720p×81

Setting. This block follows the Wan2.1-1.3B 1280×720×81 setting of HEART’s Table 2 (50 steps, CFG 5, UniPC, $n=32$). We use the model’s default flow shift of 3.0, as in the Radial and SVOO 1.3B 720p rows of Table 2. Selection, kernel, the 5 dense leading steps and TMR are those of §G.2. EBC’s per-head thresholds and ALLOT’s cell probes and trajectory gain all reuse the 480p×81 calibration. The latency table is measured at 720p. ALLOT’s table is solved targeting the default’s wall-clock time (421 dense cells), and ALLOT + TMR applies TMR to the same table.

Results (Table G.4; paired intervals in Table G.1). ALLOT is 1.2% and 1.1% faster than the default and EBC, and ALLOT + TMR runs at the wall-clock time of HEART (EBC + TMR) (§A.2). In PSNR it is 6.35, 5.16 and 5.21 dB above

Table G.4. XAttention on Wan2.1-1.3B 720p×81 ($n=32$).

Allocation	PSNR↑	SSIM↑	LPIPS↓	VBench↑	Wall-clock (s)
Dense	—	—	—	71.12	715.9
Default (threshold 0.90)	21.16	0.731	0.287	71.08	499.3
HEART (EBC)	22.36	0.762	0.241	71.05	498.5
ALLOT	27.51	0.841	0.146	72.07	493.2
HEART (EBC + TMR)	22.17	0.751	0.252	70.73	441.5
ALLOT + TMR	27.38	0.835	0.152	72.02	442.3

Table H.1. Dense wall-clock times (s) used for the speedups of Table 2. Wan models at 720p×81 except SpargeAttn (480p×81); HunyuanVideo at 720p×129.

Pipeline	Wan2.1-1.3B	Wan2.1-14B	Wan2.2-A14B	HunyuanVideo
SpargeAttn	183.8	—	—	—
Radial	728.1	3576.1	—	3192.3
SVOO	700.1	3406.0	3355.9	3225.5
SVG2	682.9	—	—	—

the default, EBC and HEART (EBC + TMR). SSIM is better on all 32 prompts in all three comparisons and LPIPS wins 32/0, 31/1 and 31/1. On VBench (same accounting as §G.2) it is +0.99 [−0.20, +2.18], +1.02 [+0.17, +1.86] and +1.30 [+0.21, +2.38] above the three.

H. Published Pipelines and Fidelity–Speedup Curves

Speedups in Table 2 are relative to the dense wall-clock times in Table H.1: the dense path of the same kernel for SpargeAttn and Radial, and the dense runs of their own pipelines for SVOO and SVG2 (§H.1 and §H.4).

H.1. SVOO

Protocol. SVOO’s official repository [14] is run unchanged, with hyperparameters copied from its 720p scripts.

14B 720p row. The table is solved from probes under SVOO’s selection strategy and the latency table of the SVOO path (§F.3), reaching equal wall-clock time with the default (§A.2; 2586.6 / 2603.5 s): PSNR 31.70 dB, +3.55 [+2.95, +4.15] 32/0, and SSIM, LPIPS and DISTs are all better as well (Table H.2).

1.3B 720p row. The table is solved with the latency table of the SVOO path (§F.3), reaching equal wall-clock time with the default (447.7 / 451.6 s): PSNR 30.73 dB, +0.66 [+0.19, +1.13] 21/11, but LPIPS (9/23) and DISTs (4/28) favor the default (Table H.2).

HunyuanVideo 720p×129 row. The setting follows the main table of the SVOO paper (HunyuanVideo-T2V, 129 frames at a resolution of 720×1280, 50 steps), and the default copies SVOO’s official HunyuanVideo configuration.

The table reuses the cell probes measured under SVOO’s selection strategy on HunyuanVideo 480p and the trajectory gain of §J.1. Only the latency table changes; as for 14B, it is fitted to the end-to-end wall-clock time of four uniform densities on the SVOO path. It targets the default’s wall-clock time. ALLOT is 7.9% faster than the default (1697.0 / 1842.7 s). The dense wall-clock time is the median over prompts 0 and 1 (3225.5 s). PSNR 30.93 dB, +3.38 [+2.51, +4.25] 31/1, and SSIM, LPIPS and DISTs are all better as well (Table H.2).

VBench. With the same accounting as §G.2, the per-prompt paired differences ALLOT − default are: 14B 720p −0.08 [−0.29, +0.12], Wan2.2 720p −0.32 [−0.94, +0.29], 1.3B 720p −0.66 [−1.55, +0.22], HunyuanVideo 720p×129 −0.25 [−1.04, +0.55]; none of the four is significant.

H.2. SpargeAttn

SpargeAttn’s released tuned hyperparameters [29] are for Wan2.1-1.3B at 480p×81, the setting of Table 2. At 720p×81 we run both the official checkpoint and one retuned on our grid with its built-in tuner: PSNR 28.64 → 30.88 and 29.30 → 31.29 dB (520.6 → 514.2 and 574.4 → 527.5 s; speedup 1.40× → 1.42× and 1.27× → 1.38×). All its default settings add a 30% warm-up, and warm-up steps use native SDPA, the same as the dense cells of ALLOT’s tables. In ALLOT’s SpargeAttn tables 40–55% of the cells are dense and use SDPA.

H.3. Radial

Radial’s mask code and kernel (FlashInfer fixed 128-block BSR) [12] are used as is. Radial’s official Wan setting uses decay 0.2 (Table 2). A second operating point with decay 1.0 gives 28.45 → 30.45 dB at equal wall-clock time (500.9 / 499.8 s; Table H.4).

At 480p×33 the minimum density of the mask is 0.525, so on Wan, Radial is run only at 720p×81, and the two groups differ only in each cell’s `decay_factor`. The table of the 14B row is solved from mean-pooled probes.

HunyuanVideo 720p×129 row. The default is Radial’s official HunyuanVideo configuration (first 12 steps dense, no dense layer, decay 0.95). We generate 129 frames at a resolution of 720×1280. ALLOT’s table is solved from probes measured under Radial (§F.1), with −0.55 [−1.17, +0.07] 16/16 at equal wall-clock time (1805.2 / 1814.2 s). The SSIM, LPIPS and DISTs intervals all contain 0.

VBench of the two pipelines. For VBench on these settings (same accounting as §G.2), ALLOT is significantly higher on the two SpargeAttn official-configuration rows

Table H.2. Paired differences (ALLOT – default) for the SVOO rows of Table 2; verdicts follow §A.1.

Setting	Baseline	n	Δ PSNR	Δ SSIM	Δ LPIPS	Δ DISTS	Verdict
SVOO 14B 720p	default	32	+3.55	+0.027	−0.024	−0.011	win
			[+2.95, +4.15]	[+0.019, +0.035]	[−0.031, −0.017]	[−0.013, −0.008]	
			32/0	31/1	31/1	30/2	
SVOO Wan2.2 720p	default	32	+4.87	+0.057	−0.045	−0.014	win
			[+4.37, +5.38]	[+0.042, +0.072]	[−0.055, −0.035]	[−0.016, −0.012]	
			32/0	32/0	32/0	32/0	
SVOO 1.3B 720p	default	32	+0.66	−0.003	+0.012	+0.017	tie
			[+0.19, +1.13]	[−0.009, +0.004]	[+0.004, +0.021]	[+0.011, +0.023]	
			21/11	14/18	9/23	4/28	
SVOO HunyuanVideo 720p×129	default	32	+3.38	+0.039	−0.047	−0.019	win
			[+2.51, +4.25]	[+0.029, +0.050]	[−0.060, −0.035]	[−0.023, −0.014]	
			31/1	30/2	29/3	29/3	

Table H.3. Paired differences (ALLOT – default) for the SpargeAttn rows; verdicts follow §A.1. The first row is from Table 2, the other two are the 720p settings above.

Setting	Baseline	n	Δ PSNR	Δ SSIM	Δ LPIPS	Δ DISTS	Verdict
SpargeAttn 1.3B 480p×81, official configuration	default	32	+5.03	+0.108	−0.080	−0.041	win
			[+4.49, +5.57]	[+0.083, +0.133]	[−0.100, −0.061]	[−0.051, −0.030]	
			32/0	32/0	32/0	32/0	
SpargeAttn 1.3B 720p, official configuration	default	32	+2.24	+0.035	−0.032	−0.015	win
			[+1.82, +2.65]	[+0.022, +0.047]	[−0.044, −0.019]	[−0.025, −0.005]	
			31/1	29/3	28/4	23/9	
SpargeAttn 1.3B 720p, retuned	default	32	+1.99	+0.028	−0.029	−0.014	win
			[+1.63, +2.36]	[+0.016, +0.041]	[−0.041, −0.018]	[−0.022, −0.006]	
			31/1	30/2	28/4	27/5	

Table H.4. Paired differences (ALLOT – default) for the Radial rows of Table 2 and the decay 1.0 setting; verdicts follow §A.1.

Setting	Baseline	n	Δ PSNR	Δ SSIM	Δ LPIPS	Δ DISTS	Verdict
Radial 1.3B 720p, decay 0.2	default	32	+0.40	+0.005	−0.005	−0.004	win
			[+0.11, +0.69]	[+0.000, +0.010]	[−0.013, +0.002]	[−0.010, +0.003]	
			26/6	24/8	23/9	21/11	
Radial 1.3B 720p, decay 1.0	default	32	+2.00	+0.016	−0.012	−0.007	win
			[+1.46, +2.54]	[+0.009, +0.024]	[−0.021, −0.004]	[−0.013, +0.000]	
			30/2	26/6	25/7	23/9	
Radial 14B 720p	default	32	+0.53	+0.011	−0.013	−0.007	win
			[+0.37, +0.70]	[+0.006, +0.015]	[−0.017, −0.009]	[−0.011, −0.003]	
			28/4	28/4	28/4	29/3	
Radial HunyuanVideo 720p×129	default	32	−0.55	−0.000	+0.010	+0.003	tie
			[−1.17, +0.07]	[−0.008, +0.007]	[−0.003, +0.023]	[−0.004, +0.010]	
			16/16	18/14	12/20	16/16	

(+1.55 [+0.78, +2.31] 27/5 and +1.19 [+0.14, +2.24] 26/6); the intervals of the other five rows all contain 0.

H.4. SVG2

Protocol. SVG2’s official Wan pipeline [25] is run (Wan2.1-1.3B, 1280×720×81, 50 steps, CFG 5, UniPC). SVOO, which builds on SVG2’s code, covers the other three

models (§H.1). Hyperparameters are copied from its 720p script (top- p 0.90, minimum key-cluster ratio 0.10). Following HEART’s protocol, the warm-up keeps the first 5 steps dense. The reference is the same pipeline with full attention at every step.

Table. Cell probes are re-measured under SVG2’s selection strategy, and the latency table is measured on the SVG2 path at this shape. The table is solved at the default’s wall-clock time (410.5 / 410.9 s). The dense wall-clock time, 682.9 s, is obtained from the dense-to-default time ratio, timed alternately on 6 prompts.

Results. PSNR 25.60 dB, $+1.34 [+0.64, +2.05]$ 26/6; the mean SSIM is slightly better ($+0.012 [-0.001, +0.024]$ 17/15), while the mean LPIPS is slightly worse ($+0.005 [-0.015, +0.024]$ 13/19) and DISTS favors the default ($+0.021 [+0.004, +0.037]$ 6/26).

VBench. The per-prompt paired difference ALLOT – default is $-0.34 [-1.48, +0.80]$, not significant.

I.5. Fidelity–speedup curves at 720p×81

The budget is swept, one table is solved per budget and paired with a uniform baseline at matched wall-clock time. The wall-clock time of the two groups differs by $\leq 2\%$.

I. Ablations

I.1. Ablation on XAttention

On XAttention, the parametric shape rule of Table 4 takes its shape from the table solved on XAttention: that table’s first 10 steps are fully dense, layer 0 is dense in 74% of the steps, and every other layer in less than 50%. The rule therefore keeps the first 10 steps and layer 0 dense over all steps, and lowers every other layer linearly with the step from 0.59 at step 10 to 0.05 at step 49, with the coefficient set by bisection on the latency table to match ALLOT’s cost.

I.2. Density-proportional cost

On the setting of Table 4 (XAttention, 1.3B 480p×81, $n=32$), we keep everything else in ALLOT fixed, only change the cost in the solver to $r(b) = b$, and then align the budget to ALLOT’s cost using the latency table. The resulting table has mean density 0.48, yet its latency-table cost is 0.73 of dense. It has 446 dense cells (ALLOT 587), and its PSNR, SSIM and LPIPS do not differ significantly from ALLOT’s (Table I.1).

I.3. Calibration metric of the trajectory gain

Setting. 1.3B 720p×81, SVOO pipeline, $n=32$. The pixel-MSE calibration uses the table at a budget of 0.62 of dense: probes under SVOO’s selection strategy and the latency table of the SVOO path (§F.3). The LPIPS and DISTS calibrations differ only in the metric used to aggregate M_k .

Results. The pixel-MSE table gives 31.12 dB. Calibrating with LPIPS instead gives 30.28 dB ($-0.84 [-1.20, -0.49]$

4/28); calibrating with DISTS gives 29.06 dB ($-2.06 [-2.71, -1.41]$ 1/31). Both alternatives also worsen LPIPS itself. Perceptual metrics saturate at large errors (LPIPS already reaches 0.52 in the first window, while MSE differs by about $460\times$), so they underweight early steps: the ratio of w over the first and last 10 steps is only 1/9 (LPIPS) and 1/29 (DISTS) of that under MSE calibration, and the densities of the early steps drop accordingly (dense cells 617 $\rightarrow$ 590 / 532).

I.4. Ablation on mean-pooled top- k

Every row is on 1.3B 480p×33 with the same selection strategy (mean-pooled top- k), kernel (VSA) and budget, $n=32$, replacing one module at a time with the rest unchanged.

As on XAttention (Table I.1), the largest contributor is trajectory calibration.

J. Cross-Model Comparison

All Wan settings of this section use mean-pooled top- k selection and the VSA kernel (the 1.3B curve of §H.5 uses FlashInfer BSR), and settings differ only in the density table.

- **Uniform baseline:** first 15 steps dense, all other cells at one density.
- **Strengthened baseline:** layer 0 dense over all steps + first 20 steps dense + uniform elsewhere (20 steps is the best of $\{0, 5, 10, 15, 20, 25\}$ on 1.3B; 14B uses 20).
- **Parametric shape rule** (Table I.2): first 20 steps and layers 0/1 dense over all steps, other layers decreasing linearly with the step, shape taken from the solved table.

J.1. Summary of four models

HunyuanVideo uses FlexAttention [5] and is compared at equal density ($r(b) = b$): at this shape the FlexAttention path offers little or no speedup over dense attention, so an equal-latency budget would leave almost nothing to allocate. On 32 held-out prompts, 1.3B is $+2.18 [+1.84, +2.51]$ 32/0 over the strengthened baseline and $+6.49 [+5.62, +7.35]$ 32/0 over the uniform baseline; 14B 480p is $+0.44 [+0.22, +0.66]$ 26/6 over the strengthened baseline (DISTS 12/20, not significant) and $+8.82 [+8.07, +9.57]$ 32/0 over the uniform baseline.

J.2. Wan2.2-A14B

The two 14B experts split the work at $t=0.875$ and share one table, and the probe noise levels t are placed on each expert’s own schedule steps. The table is solved with 14B’s latency table and Wan2.2’s own measured trajectory gain, and the three groups use the same layer-step budget.

J.3. Solved tables

The solved warm-up (the number of consecutive fully dense steps from step 0) grows as the budget increases: 10 / 15 / 20 steps at the three 14B 720p budgets (15 dense steps +

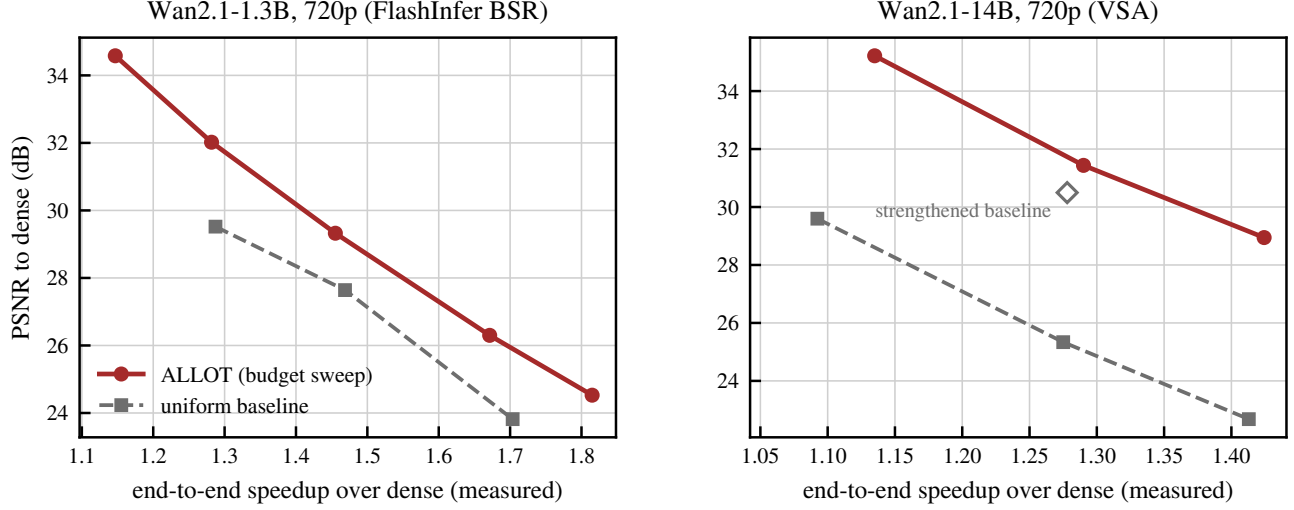


Figure H.1. Fidelity–speedup curves at 720p×81 (left Wan2.1-1.3B, right Wan2.1-14B). Red: ALLOT; gray: uniform baseline at equal wall-clock time; diamonds: strengthened baseline.

Table I.1. Paired differences (row – ALLOT) for the rows of Table 4 (ablation, no verdict).

Setting	Baseline	n	Δ PSNR	Δ SSIM	Δ LPIPS	Δ DISTS
XAttention 1.3B 480p×81	same density for all layers (trajectory gain only)	32	−1.07	−0.023	+0.022	+0.021
			[−1.34, −0.80] 2/30	[−0.029, −0.018] 1/31	[+0.017, +0.027] 2/30	[+0.016, +0.026] 3/29
XAttention 1.3B 480p×81	trajectory gain all 1	32	−7.05	−0.148	+0.131	+0.042
			[−7.52, −6.58] 0/32	[−0.173, −0.123] 0/32	[+0.110, +0.152] 0/32	[+0.033, +0.051] 1/31
XAttention 1.3B 480p×81	parametric shape rule	32	−1.76	−0.028	+0.023	+0.014
			[−2.02, −1.50] 1/31	[−0.034, −0.023] 1/31	[+0.018, +0.028] 1/31	[+0.010, +0.018] 3/29
XAttention 1.3B 480p×81	density-proportional cost	32	−0.12	+0.002	−0.001	−0.002
			[−0.26, +0.03] 14/18	[−0.000, +0.005] 20/12	[−0.003, +0.001] 16/16	[−0.004, −0.000] 20/12

Table I.2. Ablation on mean-pooled top- k (Δ = row – ALLOT).

Setting	PSNR / LPIPS / DISTS	Δ PSNR	LPIPS W/L	DISTS W/L
ALLOT	25.99 / 0.098 / 0.107	–	–	–
Same density for all layers (trajectory gain only)	22.77 / 0.189 / 0.177	−3.22 [−3.78, −2.67] 0/32	0/32	0/32
Trajectory gain all 1	14.33 / 0.389 / 0.187	−11.66 [−13.17, −10.16] 0/32	0/32	2/30
Parametric shape rule	24.86 / 0.121 / 0.129	−1.13 [−1.27, −0.99] 0/32	0/32	0/32

uniform 0.20 / 0.30 / 0.45), and 5–20 steps at the five 1.3B 720p budgets. Under the same budget, the solved warm-up is 15 steps on the Wan models, while on HunyuanVideo no step is fully dense: one of the 60 layers is at 0.30 from step 0 and four are sparse in steps 5–9. Layers 0 and 1 are dense in all 50 steps on HunyuanVideo and in 47 steps or more on Wan.

J.4. Strengthened baseline at other budgets

On 1.3B 480p×33, we apply the form of the strengthened baseline (layer 0 dense over all steps + the first k steps dense + one density for the other layers) at two other budgets and re-search k . Each budget is the cost under the same latency table of “first 15 steps dense + uniform b ”, with $b = 0.20 / 0.45$ (the same as §C.4). For each k the density of the other layers is set via the latency table to reach the same cost. At

Table J.1. PSNR (dB) of four models under the same budget. *: the uniform baseline with layer 0 additionally dense over all steps (the number of warm-up steps was not searched).

Model and setting	Speedup	n	Uniform baseline	Strengthened baseline	ALLOT	ALLOT — strengthened baseline	ALLOT — uniform baseline
Wan2.1-14B, 720p×81	1.29×	16	24.87	30.49	31.25	+0.76 [+0.46, +1.06] 15/1	+6.38 [+5.55, +7.21] 16/0
Wan2.1-1.3B, 480p×33	1.14×	32	19.59	23.44	25.99	+2.55 [+2.16, +2.94] 32/0	+6.40 [+5.63, +7.17] 32/0
Wan2.1-14B, 480p×33	1.09–1.11×	32	19.20	27.26	27.85	+0.60 [+0.41, +0.79] 28/4	+8.65 [+7.84, +9.46] 32/0
Wan2.2-A14B, 480p×33	1.09×	32	19.36	24.20*	25.86	+1.67 [+1.37, +1.96] 32/0*	+6.50 [+5.92, +7.07] 32/0
HunyuanVideo, 480p×33	equal density	32	27.82	27.46	28.42	+0.96 [+0.02, +1.90] 24/8	+0.60 [−0.34, +1.53] 22/10

Table J.2. PSNR (dB, $n=32$) of the strengthened baseline as a function of k on Wan2.1-1.3B 480p×33, and ALLOT.

Budget	$k=5$	10	15	20	25	30	35	ALLOT	ALLOT — best k	ALLOT — $k=20$
0.20	16.92	19.73	20.23	19.13	15.84	—	—	22.79	+2.56 [+2.18, +2.94] 32/0	+3.66 [+3.26, +4.06] 32/0
0.30	17.85	21.00	22.71	23.44	23.14	—	—	25.99	+2.55 [+2.16, +2.94] 32/0	+2.55 [+2.16, +2.94] 32/0
0.45	—	—	25.15	26.74	27.77	28.60	29.05	31.15	+2.09 [+1.73, +2.45] 32/0	+4.41 [+3.94, +4.88] 32/0

Table K.1. Robustness of the measurement setup. Agreement: share of cells whose dense/sparse status matches the original table (Wan2.1-1.3B unless noted).

Variable	Change	Agreement (%)	Correlation of w 's shape
Probing videos	4 → 1	XAttention 96.1–98.5; mean-pooled: 1.3B 95.2–98.0, 14B 97.9–98.4, HunyuanVideo 93.4–96.3	—
Probing videos	4 → 2	XAttention 95.7–99.3; mean-pooled: 1.3B 96.7–97.5, 14B 98.5–99.4, HunyuanVideo 96.7–97.7	—
Calibration prompts	8 → 1	XAttention 92.3–98.5; mean-pooled: 1.3B 89.1–98.5, HunyuanVideo 86.1–97.9	—
Calibration prompts	8 → 4	96.7–99.4	—
Window density	0.30 → 0.15 / 0.50	XAttention 98.9 / 98.5; mean-pooled 99.6 / 97.1	0.995 / 0.995
Window selection strategy	mean-pooled → SparseAttn	96.5 / 96.0 (two budgets)	0.993

each budget ALLOT is re-solved with the same cell probes and trajectory gain.

The best k changes with the budget: the $k=20$ selected at 0.30, moved to 0.20, is 1.10 dB lower than the best k of that budget ([0.61, 1.59], 29/3), and moved to 0.45 it is 2.32 dB lower ([1.79, 2.85], 28/4). At 0.45 PSNR rises monotonically with k ; at $k=40$ the density of the other layers would be only 0.01, so the search stops at $k=35$. After re-searching k , ALLOT is still a win at both new budgets (LPIPS 31/1 and 32/0, DISTS 21/11 and 32/0). All k are selected on the evaluation set. The warm-up solved by ALLOT is 10 / 15 / 20 steps at the three budgets.

K. Robustness of the Measurement Setup

K.1. Number of calibration inputs

Each variant re-solves the table and compares it with the 4-video, 8-prompt baseline (Table K.1). With 2 probing videos, 96–99% of cells keep the same dense/sparse status across models. On XAttention (1.3B), 1 or 2 videos lose less than 0.6 dB, about 8% of ALLOT's gain; 8 videos bring no significant improvement. Step-wise rankings are more sensitive to calibration inputs than layer-wise rankings: the single-prompt ratio of w over the first and last 10 steps ranges from 162 to 1507 on 1.3B (mean of 8: 333).

K.2. Robustness of the trajectory gain

All selection strategies reuse the trajectory gain measured with mean-pooled top- k (Table K.1). Window density changes the magnitude of w but barely its shape; calibrating separately at three densities changes 0.6% of the cells and PSNR by -0.02 [−0.10, +0.06] 15/17. Re-calibrating w on 8 held-out prompts gives a log correlation of 0.999 with the original. When the 12 tables are re-solved with this w , 98.4–99.3% of the cells retain their dense/sparse status.

K.3. Probe resolution

All 81-frame tables (720p and 480p×81) are solved from 480p×33 probes, except Radial 1.3B which uses 720p latent probes (§H.3). SVOO's co-clustering changes with resolution, so on the SVOO pipeline (1.3B 720p×81) the probes were re-measured from the dense 720p generations of 4 evaluation prompts and the table re-solved with the recipe of that row of Table 2. The mean final-video PSNR ($n=16$) increases from 30.43 to 30.48 dB, +0.05 [−0.14, +0.25] 9/7, and the LPIPS and DISTS intervals both contain 0.

K.4. Inference settings

The probes, trajectory gain and latency table are measured with 50 UniPC steps and CFG 5.0. On the first group of Table 1, we change one inference setting at a time without re-measuring, and compare ALLOT with the default against a dense reference generated under the same set-

Table K.2. Changing one inference setting after calibration (XAttention, Wan2.1-1.3B 480p $\times$ 81, $n=32$; other settings as in Table 1). Fidelity is measured against the dense output under the same setting; wall-clock time follows Appendix A.

Setting	Allocation	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	Wall-clock (s)
30 steps	Default	19.87	0.696	0.232	92.5
	ALLOT	25.80	0.830	0.110	91.9
CFG 7.5	Default	17.68	0.668	0.255	150.1
	ALLOT	24.48	0.827	0.101	147.7
Euler sampler	Default	18.79	0.664	0.278	147.8
	ALLOT	25.97	0.832	0.113	147.1

ting (Table K.2). With CFG 7.5 and the Euler sampler, ALLOT uses the table of Table 1 unchanged. With 30 steps, the table is re-solved on the 30-step schedule, taking each step’s trajectory gain from the 50-step schedule at the nearest noise level. ALLOT wins in all three settings at equal wall-clock time: +5.93 [+5.26, +6.61] 32/0 with 30 steps, +6.80 [+6.26, +7.34] 32/0 with CFG 7.5 and +7.18 [+6.41, +7.95] 32/0 with the Euler sampler, compared with 7.23 dB in the original setting.