THUNDERWORLD: EFFICIENT WORLD MODEL GENERATION WITH SELECTIVE FP4 REFINEMENT VIA NATIVE GEMMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study whether world models can recover high-quality generation through dense four-bit floating-point (FP4) matrix computation without sparse matrix instructions or token/channel reordering. ThunderWorld is a training-free framework that corrects four-bit weight-and-activation (W4A4) error using additional dense FP4 products formed from quantization residuals. Block-aligned supports jointly govern residual storage, online preparation, and dense execution under a compute budget, preserving every base attention block. Controlled attention diagnostics support structure-informed selection over random and window policies at matched repair computation. On Cosmos3 Edge and Nano across RBench and PAI-Bench-G, a configuration capped at twice base W4A4 matrix work keeps mean task-score gaps within 0.08 RBench points and 0.37 PAI-Bench-G percentage points of BF16. It reduces LPIPS, a perceptual distance to same-input, same-seed BF16 generations, by 3.3–7.8% versus ARCQuant (2.0), with 1.17–$1.88\times$ its throughput on RTX PRO 6000 and Jetson AGX Thor. These results provide a numerical and systems basis for world-model inference on devices with native FP4 Tensor Cores.

1 INTRODUCTION

Video world models support robot learning, simulation, and planning (Jang et al., 2025; NVIDIA, 2025b; Kim et al., 2026), but repeated denoising is costly. Cosmos 3 couples autoregressive (AR) reasoning and diffusion-based (DM) generation through shared attention (NVIDIA, 2026a). We ask whether generation quality can be recovered through dense native four-bit floating-point (FP4) general matrix multiplications (GEMMs), without sparse matrix instructions or token/channel reordering. Native W4A4 directly executes four-bit weights and activations, offering an attractive basis for resource-constrained deployment (NVIDIA, 2025a).

Direct W4A4 loses fidelity. On ten Edge scenes, Learned Perceptual Image Patch Similarity (LPIPS; lower is better) to BF16 is 0.2413 for W4A4 versus 0.0934 for native FP8 (Zhang et al., 2018), with matched inputs, seeds, and settings. For decoded components A_0, W_0, quantizing the remaining operand errors gives residual components A_1, W_1. Following multi-component arithmetic (Henry et al., 2019; Hayashi et al., 2026), bilateral *repair* adds $A_1 W_0^\mathsf{T}$ and $A_0 W_1^\mathsf{T}$ to the *base product* $A_0 W_0^\mathsf{T}$, lowering LPIPS to 0.0790. Uniform repair adds computation, storage, and online preparation.

Recent methods recover FP4 accuracy through activation compensation. ARCQuant augments a dense FP4 GEMM with activation-residual channels chosen offline (Meng et al., 2026a); SharQ combines a hardware-constrained $N:M$ sparse activation backbone with dense FP4 residual compensation (Meng et al., 2026b). These designs require online channel-residual packing or sparse-path preparation. In our world-model deployments, preparation, reduction, and layout operations contribute substantial runtime (Appendix D.3). Efficient repair therefore depends on how correction inputs and outputs are organized, alongside their matrix work.

ThunderWorld organizes bilateral repair around regular matrix tiles. Directional modality interactions and depth-dependent AR–DM attention provide block-level priorities in the existing token layout. Offline calibration fixes weight-residual supports and activation quotas; current inputs select correction tiles. The same supports govern residual packing and dense FP4 execution, reducing extra writes and data movement. The request budget caps issued matrix multiply-accumulate (MMA) work,

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

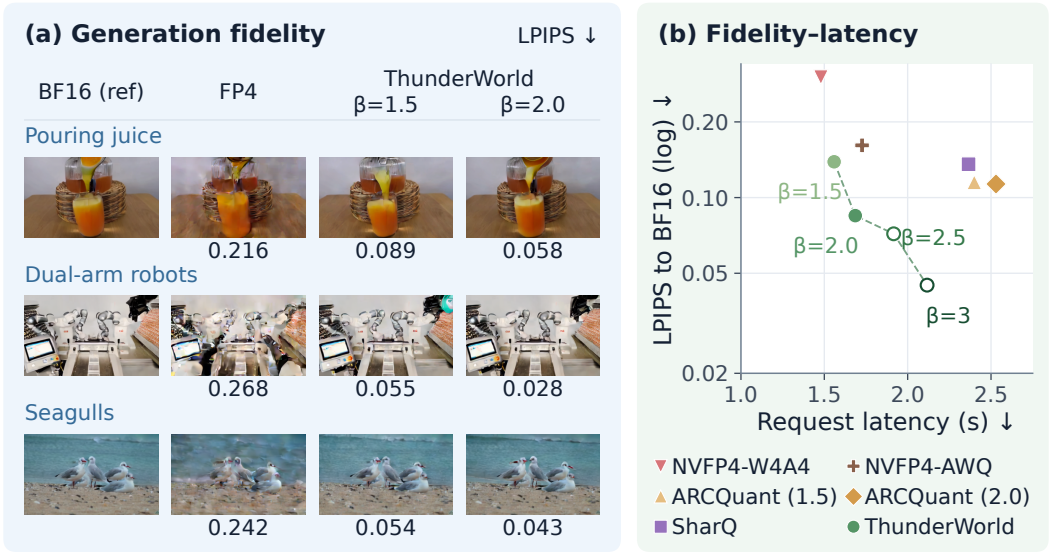


Figure 1: **Generation examples and fidelity–latency trade-offs on Cosmos3 Edge.** (a) Selected examples at frame 60, with scenes as rows and paired BF16, FP4, and ThunderWorld configurations as columns. (b) Diagnostic measurements on RTX PRO 6000 over six development scenes with two seeds each. Video LPIPS measures agreement with paired BF16 generations. Budget labels cap matrix work relative to base W4A4. NVFP4-AWQ uses W4A16 dequantization.

including padding, relative to base W4A4. All base computation is retained. Figure 1 previews the quality–latency trade-off.

Our contributions are:

- **A dense-FP4 recovery path.** We study single-sided, bilateral, and higher-order residual products in complete generation, demonstrating substantial FP4 fidelity recovery and non-monotonic higher-order gains. These findings motivate selective low-order repair.
- **Structure-aligned repair and execution.** ThunderWorld couples bilateral repair with block-aligned supports. Calibrated allocation and current-input scores determine which residual tiles to prepare and execute, linking repair priorities to compact storage, selective packing, and shared accumulation. Matched-compute diagnostics validate structure-informed selection; retained tiles use dense FP4 without token/channel reordering.
- **Server and edge validation.** Across Cosmos3 Edge and Nano on RBench and the PAI-Bench-G video-generation track (Deng et al., 2026; Zhou et al., 2025), the $2\times$ configuration lowers mean LPIPS by 3.3–7.8% versus ARCQuant (2.0) with 1.17–$1.88\times$ its throughput on RTX PRO 6000 and Jetson AGX Thor. Mean task-score gaps remain within 0.08 RBench points and 0.37 PAI-Bench-G percentage points of BF16 (Section 5.3).

2 W4A4 Fidelity Loss and Residual Repair

2.1 Residual-based precision recovery

Mixed-precision iterative refinement uses residuals to correct inexpensive approximate solves (Carson & Higham, 2018; Markidis et al., 2018). Multi-component arithmetic applies related ideas to low-precision products (Henry et al., 2019), including FP4 implementations of the Ozaki schemes (Hayashi et al., 2026). We examine how this principle affects world-model generation.

For a linear product $Y = AW^{\mathsf{T}}$, $A \in \mathbb{R}^{M\times K}$ contains input activations and $W \in \mathbb{R}^{N\times K}$ contains weights. Let $\mathcal{Q}_4$ encode and decode actual FP4 codes and scales. Quantizing once gives $A = A_0 + R_A$ and $W = W_0 + R_W$, where R_A, R_W are the remaining errors, not themselves FP4 values. For either

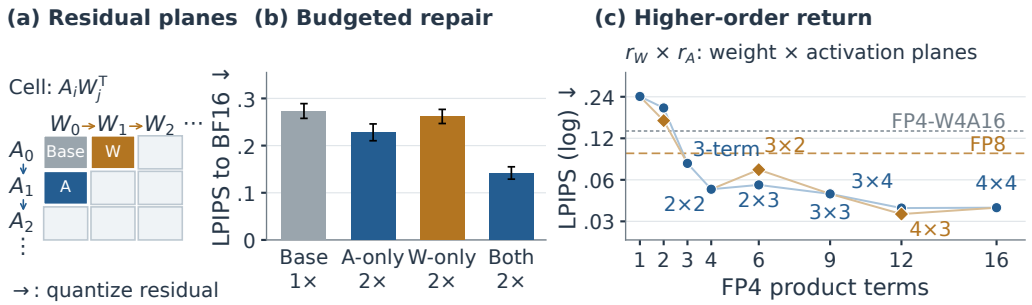


Figure 2: **FP4 residual recovery and expansion cost.** (a) Recursive residual-plane products. (b) Single-side and selected bilateral repair at a 2× request cap (base: 1×), with 95% scene-bootstrap intervals. (c) Uniform expansion versus FP8 and FP4-W4A16. Protocols and cohorts: Appendix A.

operand $X \in \{A, W\}$, recursively quantize its remaining error:

$$R_X^{(0)} = X, \qquad X_i = \mathcal{Q}_4(R_X^{(i)}), \qquad R_X^{(i+1)} = R_X^{(i)} - X_i. \tag{1}$$

Each new *component* (or stored plane) quantizes the preceding residual (Figure 2a); A_1 and W_1 thus approximate R_A and R_W. An $r_W \times r_A$ expansion pairs weight and activation planes into $r_W r_A$ products $A_i W_j^\top$. *Repair* denotes additional low-precision products that correct the base result. Bilateral first-order repair retains

$$\widehat{Y}_1 = \underbrace{A_0 W_0^\top}_{\text{base}} + \underbrace{A_1 W_0^\top}_{\text{activation repair}} + \underbrace{A_0 W_1^\top}_{\text{weight repair}}. \tag{2}$$

Activation-only and weight-only repair retain their respective additional product; bilateral repair retains both. Adding the *cross product* $A_1 W_1^\top$ gives the four-product 2×2 expansion. Weight components are prepared offline; activation components use the current input. For the two attention products, both operands are runtime activations (Section 3.1). These iterations refine representations; they do not add denoising steps. Appendix A gives encoding and residual semantics.

2.2 Measured recovery and the cost of uniform expansion

Figure 2c reports LPIPS for uniform expansion on ten Edge scenes; complete metrics appear in Appendix A.2. Three-term repair lowers mean LPIPS to BF16 from 0.2413 to 0.0790 (67.3%); adding the cross product reaches 0.0513, and 3×3 expansion reaches 0.0475 versus 0.0934 for native FP8. Gains are configuration-dependent and non-monotonic. Full three- and four-term repair store two weight planes, exceeding the same-scope FP8 footprint. Appendix A.2 reports paired uncertainty.

Under a common 2× request matrix-compute cap on 72 scenes, selected bilateral repair reaches LPIPS 0.142 versus 0.228 for A-only and 0.262 for W-only (Figure 2b). This policy includes calibrated weights and attention repair (Appendix A.3); a fixed linear-only control instead favors A-only (Appendix A.3), motivating joint selection of repair supports and representations.

Finding 1: Precision recovery through FP4 residuals

Successive FP4 residual products can exceed native-FP8 mean fidelity in the controlled expansion study. Recovery depends on operand coverage and expansion order; non-monotonic higher-order gains motivate selecting useful products and supports under compute and storage budgets.

Uniform repair exposes recoverable information but spends resources wherever an extra product has little output benefit. We next examine whether useful repair priorities align with blocks in the existing token layout, enabling selective correction without token/channel reordering.

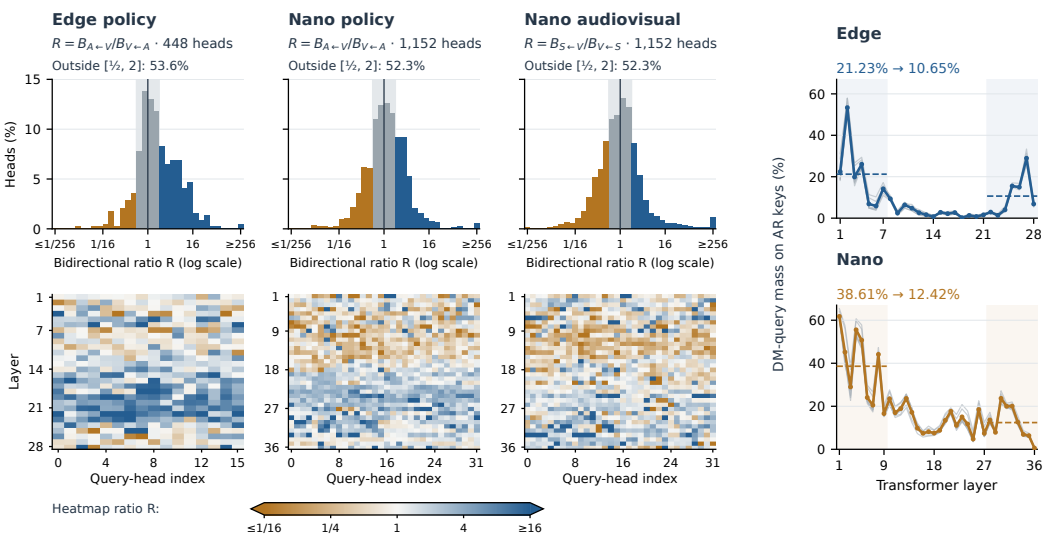


Figure 3: **Head-level directionality and depth-dependent AR attention.** Left: opposing-relation block-sum ratios, one per layer/query-head after averaging steps, seeds, and cases; histogram shading marks $[1/2, 2]$. Right: DM-query probability on AR keys by layer; gray lines show case means, colored lines model means, and dashed lines quarter means. These profiles motivate direction/head- and layer-specific repair scores, tested in Section 3.4; estimators appear in Appendix B.

3 From AR–DM Structure to Effective Repair Selection

3.1 Architecture and Measurement Units

Cosmos 3 uses separate AR/DM projections and feed-forward networks (FFNs) with shared attention (NVIDIA, 2026a;b;c). Each attention head forms queries Q, keys K, and values V; QK^{T} produces logits, softmax gives probabilities P, and PV gives the attention output. We abbreviate these products as QK and PV. Writing relations as query role ← key/value role gives

$$P = \begin{bmatrix} P_{\mathrm{AR}\leftarrow\mathrm{AR}} & 0 \\ P_{\mathrm{DM}\leftarrow\mathrm{AR}} & P_{\mathrm{DM}\leftarrow\mathrm{DM}} \end{bmatrix}, \qquad P_{\mathrm{AR}\leftarrow\mathrm{AR}} \text{ is causal.} \tag{3}$$

DM queries read AR and DM keys through one softmax. Task-dependent DM roles include conditioning vision, generated video, action, and sound. A *layer/head unit* fixes a layer and query head; *attention mass* is the sum of probabilities over a specified block. Appendix B defines the estimators and Edge/Nano measurement cohorts.

3.2 Directional Modality Blocks

Role-grouped tokens form contiguous rectangular blocks. Opposing modality blocks contain equal numbers of entries but can carry different attention mass because their queries normalize over different rows. Their layer/head block-sum ratios fall outside $[1/2, 2]$ for 53.6% of Edge action/video units and 52.3% of both Nano action/video and sound/video units (Figure 3). Nano's nearly balanced aggregate sums conceal strongly directional heads. These differences motivate retaining direction and head identity when ranking repair candidates.

Finding 2: Directional modality blocks

More than half of measured layer/head units show at least 2× attention-mass imbalance between opposing modality blocks. These block-level differences motivate direction- and head-specific correction-tile selection, retaining dense FP4 computation within tiles without token reordering.

3.3 Depth-dependent AR Context

Mean DM←AR probability mass falls from 21.23% to 10.65% between shallow and deep layer quarters on Edge, and from 38.61% to 12.42% on Nano (Figure 3). AR context therefore has a different average contribution across layers. Local rebounds remain, so we preserve layer-specific measurements instead of imposing a monotonic cutoff. Both findings concern blocks in the existing layout: they can guide *repair-tile skipping*, which omits additional correction products without removing any base attention block.

Finding 3: Depth-dependent AR attention

DM←AR attention is stronger in shallow layers and weaker in deep layers on average, with local rebounds. This block-level pattern motivates layer-specific repair priorities, preserving all base attention blocks while selectively skipping correction tiles according to measured structure.

3.4 Does Structure Improve Repair Selection?

Attention mass is a candidate signal, not a direct measure of repair benefit: QK corrections change the common softmax and hence the complete output. We test this link with a frozen structure-informed selector. A development prior records probability density for each directed role pair, layer, head, and denoising phase (step interval). Current query/key role counts map that prior onto 128×128 tiles; the highest-scoring tiles receive bilateral repair. Directionality supplies distinct role/head scores, while depth enters through layer-specific priors without a hard cutoff. Appendix B.2 gives the scoring rule.

We evaluate 320 complete attention outputs along 16 unchanged BF16 trajectories: two fixed scenes and four seeds each for Edge image-to-video (I2V) and policy. Normalized mean-squared error (NMSE) is squared output error divided by squared BF16 output norm, before the output projection.

Inspired by uniform random selection in Random Attention (Wang et al., 2026), we include a random repair-tile baseline (eight masks). We compare against a bidirectional window along global token order and a *current-attention-mass oracle*. The oracle ranks exact current probability mass; it does not minimize output error. All policies select the same number of head/key tiles per 128-query row, use the same mask for QK and PV, and retain every base block. Issued FP4 MMA work, including padding, is identical. The attention-only ratio $\beta_{\text{attn}} = 1.5$ or 2.0 compares base plus repair work to base QK/PV work; it differs from the whole-request budget in Section 4. Selector costs are separate.

Table 1: **Structure-guided repair selection.** Attention-output NMSE and absolute gaps ($\times 10^{-3}$) on fixed BF16 trajectories, with matched padded FP4 repair work and all base blocks retained. $\delta_{\text{oracle}} = |E_{\text{structure}} - E_{\text{oracle}}|$; $\Delta_{\text{win}} = E_{\text{window}} - E_{\text{structure}}$. Gray marks our selection; bold marks the best non-oracle NMSE. The oracle maximizes current attention mass.

Edge task	β_{attn}	Random	Window	Structure	Mass oracle	$\delta_{\text{oracle}} \downarrow$	$\Delta_{\text{win}} \uparrow$
I2V	1.5	22.068	13.293	**11.193**	6.252	4.941	2.100
I2V	2.0	15.542	9.583	**5.708**	3.754	1.954	3.875
Policy	1.5	23.322	15.914	**12.919**	10.183	2.735	2.996
Policy	2.0	16.251	11.325	**5.468**	5.194	0.275	5.857

Structure-informed selection lowers mean NMSE by 44.6–66.4% versus random and 15.8–51.7% versus window (Table 1). On policy at $\beta_{\text{attn}} = 2$, its NMSE is 0.005468 versus 0.005194 for the mass oracle, an absolute gap of 0.000275; the other settings have larger gaps. These local gains support structure-informed selection. The complete generation policies in Section 5 use calibrated quotas and current-input scores, separately from this prior-only diagnostic.

4 ThunderWorld: Budgeted Repair with Dense FP4 Execution

ThunderWorld connects repair priorities to execution through a common tile support: it determines which residuals to store, prepare, and multiply. Attention uses the existing role blocks; linear/FFN operators use their own residual and sensitivity scores (Figure 4).

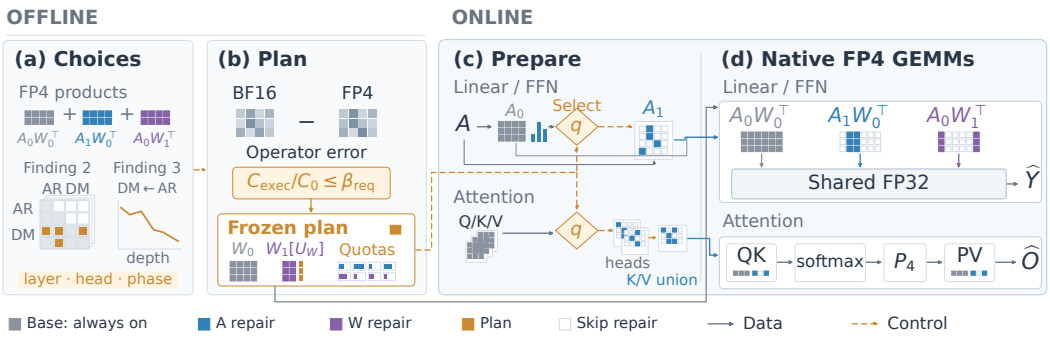


Figure 4: **From repair allocation to dense FP4 execution.** (a) Findings 2–3 guide candidate grouping for repair-tile selection and skipping. (b) Operator-output calibration determines weight supports and activation quotas under the request budget β_{req}. (c) Current inputs select residuals to pack, including shared K/V support unions. (d) Retained products use dense FP4 MMA tiles with shared Linear/FFN accumulation or streaming QK–softmax–PV; unselected correction tiles are skipped while the base is always active. Tiles and quotas are schematic; Appendix B.3 gives the evaluated policies.

4.1 Offline Allocation and Current-input Selection

A *repair tile* is a matrix block on which a correction product executes. Its *candidate set* contains legal blocks; a *support* is the subset actually selected. A *quota* caps the number selected within a group. Offline calibration fixes weight supports and activation quotas by same-input BF16 operator-output error reduction. Groups follow operators, layers, heads, denoising phases (step intervals), and classifier-free guidance (CFG) branches, with separate QK/PV quotas.

For request shape x, preparation verifies every legal online selection S against the normalized request budget β_{req}:

$$C_{\text{exec}}(S, x)/C_0(x) \le \beta_{\text{req}}. \tag{4}$$

C_0 counts base W4A4 projection, FFN, and attention MMA at identical shapes and execution tiles. The ledger counts two operations per multiply-add, including all denoising steps, CFG branches, tail padding, configured cross products, and selector MMA, with data types itemized. Packing, sorting, and softmax costs are measured separately; this bound is on matrix work, not latency.

At runtime, linear operators rank current activation-residual energy weighted by offline channel sensitivity. Attention ranks sampled probability mass times current value-tile root-mean-square magnitude, subject to the frozen QK/PV quotas. The $1.5\times$ deployment policy uses partial head-specific attention repair; $2\times$ uses full three-term attention and larger linear quotas. Appendix B.3 specifies these frozen policies. Each call uses its current input, and selected supports govern both preparation and execution.

4.2 From Selected Tiles to Reduced Execution Work

Compact residual weights. Each linear operator stores complete W_0 and only the union of W_1 supports needed across the request. Codes, scales, and block indices are prepared offline. Calls address subsets of this asset, so persistent residual storage follows the union rather than a second complete plane.

Selective activation preparation. *Packing* converts values into FP4 codes and scales consumed by the kernel. Base packing computes full-input statistics, A_0, and residual summaries; after selection, a second pass writes A_1 codes and scales only for the support consumed by correction GEMMs. In grouped-query attention (GQA), multiple query heads share a key/value head (Ainslie et al., 2023); K/V preparation therefore uses their support union. Selective packing reduces writes, although scratch buffers retain their full allocated extents.

Dense correction execution. Each selected product executes as a regular dense FP4 GEMM tile. Linear/FFN base and repair products feed one FP32 accumulator: activation repair reuses W_0, while weight repair loads compact W_1. Unselected repair tiles issue no corresponding products, and shared accumulation avoids full-sized correction outputs and their combine traffic. Attention uses the same FP4 matrix primitives in a streaming kernel: corrected QK feeds the joint softmax, whose current probability tile is quantized for PV and its selected corrections. FlashAttention-style running normalization (Dao et al., 2022) avoids storing the full probability matrix. All base blocks remain; softmax and selection retain their non-GEMM roles. Prepared CUDA Graphs replay the dependent kernels with current inputs, reducing host-launch overhead. Appendix D.2 gives packing compatibility and backend details.

5 EXPERIMENTS

We evaluate fidelity to paired BF16 generations, task adherence under vision-language model (VLM) evaluation, and deployment costs on server and edge hardware.

Table 2: Category-averaged fidelity to paired BF16 (PSNR in dB). Native FP8 and NVFP4-AWQ are higher-precision compute references above each separator. Lower rows use native FP4 GEMMs in the quantized generation path. Bold marks this group's best values; gray denotes ThunderWorld.

	RBench			PAI-Bench-G		
Method	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑
Cosmos3 Edge						
Native FP8	0.1188	20.61	0.7992	0.0990	23.96	0.8071
NVFP4-AWQ	0.1768	17.87	0.7247	0.1749	20.13	0.6964
NVFP4 W4A4	0.3001	15.61	0.5687	0.2824	17.76	0.5648
ARCQuant (1.5)	0.1935	17.32	0.6919	0.1829	19.94	0.6775
ARCQuant (2.0)	0.1776	17.75	0.7214	0.1670	20.46	0.7047
SharQ	0.1918	17.31	0.6997	0.1812	19.86	0.6794
ThunderWorld (1.5)	0.1928	17.37	0.6908	0.1834	19.79	0.6676
ThunderWorld (2.0)	**0.1717**	**18.10**	**0.7248**	**0.1567**	**20.85**	**0.7157**
Cosmos3 Nano						
Native FP8	0.0931	22.13	0.8241	0.0960	24.26	0.8042
NVFP4-AWQ	0.1201	20.47	0.7881	0.1214	22.77	0.7661
NVFP4 W4A4	0.3584	16.02	0.5253	0.3979	16.71	0.4849
ARCQuant (1.5)	0.1339	19.84	0.7592	0.1408	21.76	0.7287
ARCQuant (2.0)	0.1250	20.20	0.7765	0.1322	22.15	0.7446
SharQ	0.1301	19.95	0.7677	0.1346	22.03	0.7396
ThunderWorld (1.5)	0.1381	19.86	0.7520	0.1519	21.42	0.7093
ThunderWorld (2.0)	**0.1181**	**20.67**	**0.7848**	**0.1219**	**22.71**	**0.7577**

5.1 EXPERIMENTAL SETUP

Models and workloads. We evaluate Cosmos3 Edge and Nano (NVIDIA, 2026b;c) on 650 RBench scenes (Deng et al., 2026) and the 1,044-scene video-generation track of PAI-Bench (PAI-Bench-G) (Zhou et al., 2025) in the image-to-video (I2V) setting. RBench covers robot manipulation and planning across embodiments, while PAI-Bench-G assesses physical plausibility across diverse domains. We use two paired seeds per scene, 320×192 resolution, 61 frames, and 35 denoising steps. ThunderWorld uses request MMA caps $\beta_{\text{req}} = 1.5/2.0$ (β in full-generation figures and tables).

Baselines and repair scope. Higher-precision compute references are native W8A8 FP8 and ModelOpt NVFP4-AWQ (NVIDIA, 2026d), which decodes FP4 weights for BF16 GEMM. Native-FP4 baselines are NVFP4 W4A4, activation-only ARCQuant, and SharQ's sparse/dense activation compensation (Meng et al., 2026a;b), adapted to the same generation projection/FFN scope. ARCQuant (1.5)/(2.0) label our tested channel budgets: nominal linear MMA work is $1 + S/K = 1.5/2.0$, where S residual channels extend a GEMM's reduction dimension K. Both baselines retain BF16 attention; ThunderWorld caps request-wide projection, FFN, and attention MMA (Appendix D.2).

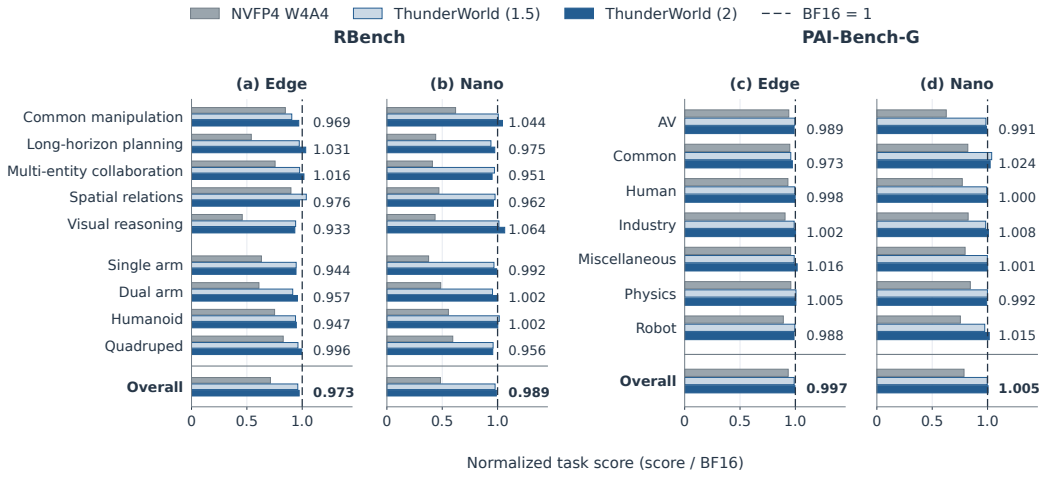


Figure 5: **Task-category scores normalized to BF16.** Bars show NVFP4 W4A4 and ThunderWorld at $\beta_{\text{req}} = 1.5/2.0$; dashed lines mark BF16$= 1$ and labels report $\beta_{\text{req}} = 2.0$. Overall uses category-macro means before normalization. Raw scores and uncertainty are in Appendix C.3.

Evaluation protocol. We conduct quality and performance experiments on an NVIDIA RTX PRO 6000 Server Edition (600 W), with additional performance experiments on the NVIDIA Jetson AGX Thor edge device. LPIPS (Zhang et al., 2018), peak signal-to-noise ratio (PSNR; higher is better), and structural similarity (SSIM; higher is better) (Wang et al., 2004) compare all 61 raw decoded frames with same-input, same-seed BF16 outputs. We average seeds within scenes and weight categories equally. The benchmark-coverage cohort includes prior development scenes; Appendix C gives cohort roles, settings, metrics, and paired uncertainty.

5.2 Full-Generation Fidelity

Fidelity at $\beta_{\text{req}} = 2.0$. ThunderWorld lowers mean LPIPS by 3.3–7.8% relative to ARCQuant (2.0) across both models and benchmarks (Table 2). Mean PSNR increases by 0.35–0.56 dB and SSIM by 0.0034–0.0131. LPIPS is also 9.3–13.6% lower than SharQ. Bilateral repair achieves these gains with FP4 attention, while both activation-compensation baselines retain BF16 attention.

Budget trade-off. Increasing β_{req} from 1.5 to 2.0 reduces mean LPIPS by 11.0–19.8%. The $1.5\times$ policy does not consistently outperform ARCQuant (1.5), while native FP8 retains better fidelity than both policies. Paired statistics and visual examples appear in Appendices C and E.

5.3 Task Preservation under VLM Evaluation

Qwen3-VL-8B-Instruct (Bai et al., 2025) evaluates task adherence with method labels hidden. At $\beta_{\text{req}} = 2.0$, ThunderWorld improves mean RBench scores by 0.74–1.46 points (36.4–104.2%) and PAI-Bench-G accuracy by 4.42–16.28 percentage points over NVFP4 W4A4. Mean differences from BF16 remain within 0.08 RBench points and 0.37 PAI-Bench-G percentage points, with category-level variation in Figure 5. Raw scores, paired uncertainty, and evaluator analyses appear in Appendix C.3.

5.4 Weight Storage and Runtime Efficiency

Weight storage. Table 3 summarizes static model weight: ThunderWorld uses 31–34% less storage than FP8. Appendix D.2 gives the accounting scope and breakdown.

Generation throughput. ThunderWorld achieves 1.17–$1.88\times$ ARCQuant (2.0) throughput at $\beta_{\text{req}} = 2.0$ and 1.18–$1.94\times$ ARCQuant (1.5) at $\beta_{\text{req}} = 1.5$ across the tested model/device settings. At $2\times$, throughput is also 1.17–$1.75\times$ SharQ's. Both policies remain slower than unrepaired W4A4 (Figure 6). Appendix D.1 reports request latency and W4A4-relative throughput.

Table 3: **Static Model Weight (GiB).** TW denotes our method, ThunderWorld.

Model (params)	BF16	FP8	FP4	ARC (1.5)	SharQ	TW (1.5)	TW (2)
Edge (4B)	5.25	2.95	1.95	2.33	2.04	1.99	2.02
Nano (16B)	25.88	14.34	9.31	11.14	9.73	9.50	9.86

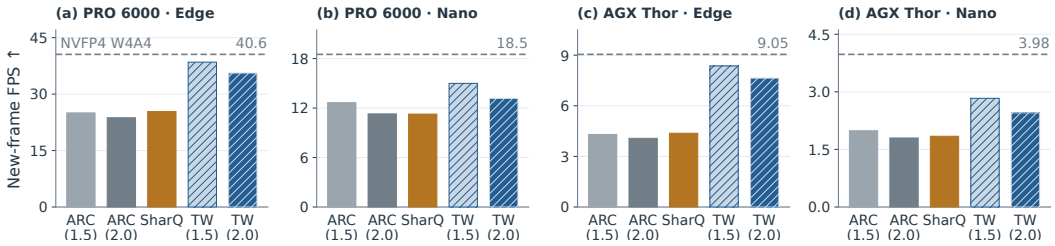


Figure 6: **Generation throughput of native FP4 GEMM implementations.** FPS is 60 generated frames divided by median complete-request time. Dashed lines show NVFP4 W4A4; hatched bars show ThunderWorld (TW). ARC denotes ARCQuant; panels use independent vertical scales. Latency and timing protocols are in Appendix D.

Execution costs. Selected supports couple residual packing and shared accumulation to dense repair execution. Figure D.2 shows lower combined preparation and peripheral costs than ARCQuant (2.0) and SharQ. At $\beta_{\text{req}} = 2.0$, quantization/preparation, reduction, and layout savings outweigh longer projection/FFN and attention time: the observed advantage comes chiefly from repair organization and data movement.

6 Related Work

Numerical decomposition and compensation. Multi-component arithmetic reconstructs products from low-precision components (Henry et al., 2019; Markidis et al., 2018; Hayashi et al., 2026). AWQ, Four Over Six, BRECQ, and SVDQuant mitigate low-bit error through scaling, reconstruction, or higher-precision correction (Lin et al., 2024; Cook et al., 2025; Li et al., 2021; 2025). ARCQuant duplicates weight columns for offline-selected residual channels, extending a dense FP4 GEMM (Meng et al., 2026a). SharQ combines hardware-valid N:M sparse activations with a dense FP4 residual path and shared weights (Meng et al., 2026b). Both fuse preparation for activation-side recovery. ThunderWorld couples bilateral repair to block-aligned selection, compact residual storage, and selective packing; retained tiles use dense FP4 without channel reordering or $N:M$ masks.

Diffusion and attention. Q-Diffusion and ViDiT-Q study diffusion quantization for image and video models; SageAttention3 provides FP4 attention (Li et al., 2023; Zhao et al., 2025; Zhang et al., 2025). Sparse VideoGen, Sparse VideoGen2, and SVG-EAR reduce base attention work (Xi et al., 2025; Yang et al., 2025; Zhou et al., 2026); Sparse VideoGen2 clusters and permutes tokens. ThunderWorld instead keeps all base blocks and token/channel order, selecting only repair products. Random Attention motivates our random selector (Wang et al., 2026).

7 Conclusion

ThunderWorld demonstrates substantial world-model fidelity recovery and near-BF16 task-score means using dense native-FP4 base and correction products, without sparse matrix instructions or token/channel reordering. Recursive residual quantization recovers lost information through additional FP4 products; budgeted bilateral repair selects which products and tiles merit their cost. The same supports govern residual storage, activation packing, and dense execution, reducing preparation and data movement around GEMMs. Surrounding operations remain mixed precision, and FP4 attention is a key remaining performance bottleneck. Together, these results provide a numerical and systems basis for world models to exploit abundant low-precision Tensor Core compute on resource-constrained edge devices.

Reproducibility statement

The appendix specifies numerical semantics, evaluation protocols, and support selection. The anonymous repository provides implementation source, source hashes, aggregate result tables, paired statistics, and verification scripts: https://anonymous.4open.science/r/ThunderWorld-submit-1DC3. Full per-video records, frozen benchmark assets, and machine logs are retained separately and are not included in the review artifact.

AI use statement

We used generative AI tools to provide feedback on research methodology and experimental design, and to assist with coding and experiment execution. We did not use generative AI tools to generate synthetic datasets, formulate or prove mathematical claims, perform translation, or conduct qualitative data analysis. We also used generative AI tools for literature search, manuscript drafting and editing, and scientific figure preparation. All AI-assisted outputs were reviewed and verified by the authors, who made the final decisions on the methodology, experiments, interpretation, and scientific claims, and take responsibility for the final content.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Erin Carson and Nicholas J. Higham. Accelerating the solution of linear systems by iterative refinement in three precisions. *SIAM Journal on Scientific Computing*, 40(2):A817–A847, 2018. doi: 10.1137/17M1140819. URL https://epubs.siam.org/doi/10.1137/17M1140819.

Jack Cook, Junxian Guo, Guangxuan Xiao, Yujun Lin, Keith Wyss, Mahdi Nazemi, Asit Mishra, Carlo del Mundo, Tijmen Blankevoort, and Song Han. Four over six: More accurate NVFP4 quantization with adaptive block scaling. *arXiv preprint arXiv:2512.02010*, 2025. URL https://arxiv.org/abs/2512.02010.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022. doi: 10.52202/068431-1189. URL https://proceedings.neurips.cc/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

Yufan Deng, Zilin Pan, Hongyu Zhang, Xiaojie Li, Ruoqing Hu, Yufei Ding, Yiming Zou, Yan Zeng, and Daquan Zhou. Rethinking video generation model for the embodied world. *arXiv preprint arXiv:2601.15282*, 2026. URL https://arxiv.org/abs/2601.15282.

Shun-ichiro Hayashi, Daichi Mukunoki, Tetsuya Hoshino, and Takahiro Katagiri. DGEMM with Ozaki scheme I/II on FP4 tensor cores: A base-13 E2M1 limb representation. *arXiv preprint arXiv:2608.06812*, 2026. URL https://arxiv.org/abs/2608.06812v1.

Greg Henry, Ping Tak Peter Tang, and Alexander Heinecke. Leveraging the bfloat16 artificial intelligence datatype for higher-precision computations. In *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*, pp. 69–76, 2019. doi: 10.1109/ARITH.2019.00019. URL https://doi.org/10.1109/ARITH.2019.00019.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. DreamGen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025. URL https://arxiv.org/abs/2505.12705v2. Version 2.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos Policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026. URL https://arxiv.org/abs/2601.16163.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. SVDQuant: Absorbing outliers by low-rank component for 4-bit diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f34f0630c33be15b8c89426bb8056798-Abstract-Conference.html.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-Diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17535–17545, October 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Li_Q-Diffusion_Quantizing_Diffusion_Models_ICCV_2023_paper.html.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations*, 2021. URL https://iclr.cc/virtual/2021/poster/2596.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *MLSys*, volume 6, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Stefano Markidis, Steven Wei Der Chien, Erwin Laure, Ivy Bo Peng, and Jeffrey S. Vetter. NVIDIA tensor core programmability, performance & precision. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 522–531, 2018. doi: 10.1109/IPDPSW.2018.00091. URL https://doi.org/10.1109/IPDPSW.2018.00091.

Haoqian Meng, Yilun Luo, Yafei Zhao, Wenyuan Liu, Peng Zhang, and Xindian Ma. ARCQuant: Boosting NVFP4 quantization with augmented residual channels for LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8609–8623, 2026a. doi: 10.18653/v1/2026.acl-long.388. URL https://aclanthology.org/2026.acl-long.388/.

Haoqian Meng, Yilun Luo, Yafei Zhao, Wenyuan Liu, Huaqing Zheng, Xindian Ma, and Peng Zhang. SharQ: Bridging activation sparsity and FP4 quantization for LLM inference. *arXiv preprint arXiv:2606.26587*, 2026b. URL https://arxiv.org/abs/2606.26587.

NVIDIA. NVIDIA RTX Blackwell GPU Architecture, 2025a. URL https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/quadro-product-literature/pdf/NVIDIA-RTX-Blackwell-PRO-GPU-Architecture-v1_1.pdf. White paper, version 1.1, Appendix A.

NVIDIA. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025b. URL https://arxiv.org/abs/2501.03575.

NVIDIA. Introducing NVFP4 for efficient and accurate low-precision inference. NVIDIA Technical Blog, 2025c. URL https://developer.nvidia.com/blog/introducing-nvfp4-for-efficient-and-accurate-low-precision-inference/. Accessed 2026-09-04.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026a. URL https://research.nvidia.com/labs/cosmos-lab/cosmos3/technical-report.pdf.

NVIDIA. Cosmos3-Edge: Official model card, 2026b. URL https://huggingface.co/nvidia/Cosmos3-Edge. Model-card reference; experimental checkpoint revisions are frozen separately.

NVIDIA. Cosmos3-Nano: Official model card, 2026c. URL https://huggingface.co/nvidia/Cosmos3-Nano. Model-card reference; experimental checkpoint revisions are frozen separately.

NVIDIA. TensorRT Model Optimizer: Quantization configurations, 2026d. URL https://github.com/NVIDIA/TensorRT-Model-Optimizer/blob/main/modelopt/torch/quantization/config.py. Official source, including NVFP4 AWQ; accessed September 21, 2026. Experimental software versions are pinned separately.

Heng Wang, Jielin Qiu, Wenting Zhao, Cheng Qian, Liangwei Yang, Jiawei Han, Heng Ji, Silvio Savarese, Shelby Heinecke, and Huan Wang. Random attention: Rethinking KV cache eviction for efficient reasoning. *arXiv preprint arXiv:2609.03430*, 2026. URL https://arxiv.org/abs/2609.03430.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861. URL https://ece.uwaterloo.ca/~z70wang/publications/ssim.pdf.

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

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, Jianfei Chen, Song Han, Kurt Keutzer, and Ion Stoica. Sparse VideoGen2: Accelerate video generation with sparse attention via semantic-aware permutation. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3243. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/8bd5148caced2d73cea7b6961a874a49-Abstract-Conference.html.

Jintao Zhang, Jia Wei, Haoxu Wang, Pengle Zhang, Xiaoming Xu, Haofeng Huang, Kai Jiang, Jianfei Chen, and Jun Zhu. SageAttention3: Microscaling FP4 attention for inference and an exploration of 8-bit training. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-1799. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4db397e0f760cc573c681e81a01a3dba-Abstract-Conference.html.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Tianchen Zhao, Tongcheng Fang, Haofeng Huang, Rui Wan, Widyadewi Soedarmadji, Enshu Liu, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang.

ViDiT-Q: Efficient and accurate quantization of diffusion transformers for image and video generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a4a1ee071ce0fe63b83bce507c9dc4d7-Abstract-Conference.html.

Fengzhe Zhou, Jiannan Huang, Jialuo Li, Deva Ramanan, and Humphrey Shi. PAI-Bench: A comprehensive benchmark for physical AI. *arXiv preprint arXiv:2512.01989*, 2025. URL https://arxiv.org/abs/2512.01989.

Xuanyi Zhou, Qiuyang Mang, Shuo Yang, Haocheng Xi, Jintao Zhang, Huanzhi Mao, Joseph E. Gonzalez, Kurt Keutzer, Ion Stoica, and Alvin Cheung. SVG-EAR: Parameter-free linear compensation for sparse video generation via error-aware routing. *arXiv preprint arXiv:2603.08982*, 2026. URL https://arxiv.org/abs/2603.08982.

A Numerical Semantics and Refinement Evidence

We use native NVFP4 E2M1 components with K16 E4M3 block scaling (NVIDIA, 2025c). Equation 1 quantizes the decoded residual left by preceding planes; products accumulate in FP32 before BF16 output. Channel compensation uses A/s and Ws, with packing prescale $1/s$ and BF16 rounding of scaled inputs. With r_A activation planes and r_W weight planes, complete expansion is

$$Y_{r_W \times r_A} = \sum_{i=0}^{r_A-1} \sum_{j=0}^{r_W-1} A_i W_j^{\mathsf{T}}. \tag{5}$$

The first-order form in Equation 2 retains the base and both operand corrections; adding $A_1 W_1^{\mathsf{T}}$ gives the 2×2 expansion. For $E_X = X - X_0$ and $\delta_X = E_X - X_1$, its error before accumulation and output rounding is

$$Y - \widehat{Y}_1 = \delta_A W_0^{\mathsf{T}} + A_0 \delta_W^{\mathsf{T}} + E_A E_W^{\mathsf{T}}. \tag{6}$$

The first two terms contain unrepresented residuals; the last is the omitted product of operand errors. Rectangles count components, not polynomial degree.

A.1 Residual-aware weight encoding

We additionally optimize already selected W1 blocks by jointly tuning component scales and optionally moving at most one element per K16 group from the base quantization input into its residual. W1 is recomputed from the original weight and decoded W0; affected blocks remain mandatory. On sampled Edge attention projections, joint search reduces weight NMSE by 22.3% relative to plain residual quantization, compared with 21.4% for scale search alone. Its additional full-operator improvement over scale search is only 0.0013% for three-term repair. We treat this as an offline encoding optimization; the main generation gains rely on numerical recovery and support allocation.

A.2 Complete-generation expansion

Table A.1 and Figure A.1 evaluate ten Edge scenes, five per benchmark, using the generation settings in Section 5.1 and same-seed BF16 references. Metrics average scenes equally. FP4 variants share W0 and projection/FFN scope. In attention, A-/W-side repair refines current Q/P and K/V activations, respectively; QK corrections precede the common softmax. Base blocks remain complete. Product counts describe each controlled GEMM rather than request compute.

Table A.1: Edge fidelity to paired BF16 on ten scenes, one seed each. $Wk \times Aj$ denotes k weight-side and j activation-side components (current K/V and Q/P in attention). Counts are products per controlled GEMM.

Configuration	PSNR ↑	SSIM ↑	LPIPS ↓	Rel. L_2 ↓
Native FP8	21.8377	0.832031	0.093386	0.168254
NVFP4-AWQ W4A16	19.4537	0.762059	0.135545	0.217643
1 term (W4A4)	17.4678	0.616533	0.241311	0.277014
2 terms (A-side)	17.7701	0.659644	0.199144	0.265985
2 terms (W-side)	19.3148	0.706061	0.161150	0.223087
3 terms (first order)	23.4849	0.861584	0.078965	0.146610
4 terms (2×2)	26.1406	0.908237	0.051284	0.108360
6 terms ($W2 \times A3$)	25.8244	0.908841	0.055134	0.113263
6 terms ($W3 \times A2$)	27.0174	0.882810	0.071042	0.121706
9 terms (3×3)	28.9291	0.927862	0.047455	0.092832
12 terms ($W3 \times A4$)	27.8159	0.933359	0.037453	0.094704
12 terms ($W4 \times A3$)	28.8223	0.940925	0.033883	0.087088
16 terms (4×4)	28.6418	0.926100	0.037688	0.095008

Three-term bilateral repair reduces mean LPIPS by 67.3% relative to W4A4 and improves the mean over FP8 on this panel. Adding the cross product reduces LPIPS further, from 0.0790 to 0.0513, and improves over FP8 on all ten scenes. Higher-order returns are metric-dependent and nonmonotonic: twelve products ($W4 \times A3$) outperform sixteen in all three main metrics. These results motivate low-order recovery with selective allocation rather than uniformly increasing the product count.

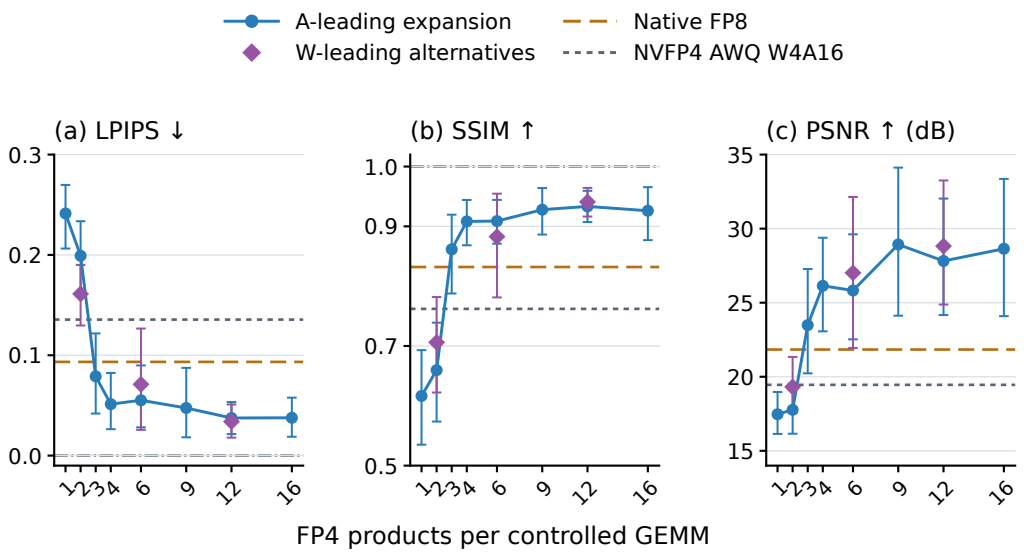


Figure A.1: Ten-scene expansion curves with 95% scene-bootstrap intervals. Blue uses A-only, 2×3, and 3×4 at 2/6/12 products; purple uses W-only, 3×2, and 4×3. Dimensions count W-side $\times$ A-side planes. The horizontal axis counts products per controlled GEMM.

A.3 Fixed and selected bilateral repair

The extended cohort has 36 RBench and 36 PAI-Bench-G Edge scenes, with two seeds and paired BF16 references. Table A.2 fixes W0/W1 and the quantizer, varying only projection/FFN repair. Bilateral controls use equal operand quotas and residual-based block rankings without calibrated allocation; all attention corrections are disabled.

Table A.2: Fixed linear-repair allocation on 72 Edge scenes, without calibrated support allocation or attention repair. Bilateral rows split the available linear-repair budget evenly between operands. Metrics average paired-BF16 comparisons over seeds and scenes.

Linear repair	Linear cap	LPIPS ↓	SSIM ↑	PSNR ↑	Rel. L_2 ↓
Base	1×	0.273221	0.606003	17.4063	0.278846
A-only	2×	**0.228083**	**0.647131**	**17.9208**	**0.268441**
W-only	2×	0.261767	0.616308	17.6784	0.270808
Both, 25% each	1.5×	0.246427	0.630586	17.7248	0.272022
Both, 50% each	2×	0.239845	0.636381	17.8153	0.271282

Fixed versus selected allocation. At the 2× linear cap, the fixed 50/50 policy reaches 0.2398 LPIPS, compared with 0.2281 for A-only and 0.2618 for W-only. Uniformly dividing repair work therefore does not by itself improve on the stronger single side. The frozen ThunderWorld policies reach 0.1721 and 0.1420 at the $1.5 \times /2\times$ request caps on the same 72 scenes (Figure 2b shows 2×). These compare complete policies: selected runs use their own calibrated weight assets, attention repairs, and seed pairs. Together, the controls support selecting both repair supports and allocation rather than assuming a uniform split is optimal.

B Structural Measurements and Support Selection

B.1 Directed relations and head distributions

For role i querying role j, define the block mass under the original row-wise softmax as

$$B_{i\leftarrow j} = \sum_{u\in i}\sum_{v\in j} P_{uv}, \qquad R^{i,j}_{\ell h} = \frac{\overline{B}_{\ell h, i\leftarrow j}}{\overline{B}_{\ell h, j\leftarrow i}}. \tag{7}$$

Opposing blocks contain the same $n_i n_j$ entries. When queries are sampled, their sums are scaled by the full-to-sampled query count; every valid key remains in the softmax. The bars average steps, seeds, and cases before division. A head unit is a fixed layer and query head, including separate GQA query heads. Roles are conditioning vision (C), generated video (V), action (A), and sound (S).

Table B.1: Fractions of layer/query-head units outside each factor-t band $[1/t, t]$. *Mass,* $2\times$ gives their summed pair-probability share; the last column is the observed range of per-trajectory head fractions.

Model/task	Pair	$1.5\times$	$2\times$	$4\times$	Mass, $2\times$	Range, $2\times$
Edge I2V	C/V	82.6%	75.0%	42.4%	86.1%	69.6–78.6%
Edge Policy	C/V	85.9%	75.4%	47.3%	81.9%	71.9–76.8%
Edge Policy	C/A	80.4%	68.1%	40.8%	75.1%	66.5–74.6%
Edge Policy	V/A	70.3%	53.6%	32.4%	49.5%	52.2–56.9%
Nano I2V	C/V	81.2%	65.1%	33.2%	76.6%	63.6–68.0%
Nano Policy	C/V	78.6%	63.8%	27.1%	70.2%	61.8–66.0%
Nano Policy	C/A	74.7%	56.1%	26.5%	52.2%	53.5–62.4%
Nano Policy	V/A	70.1%	52.3%	21.7%	48.1%	51.0–56.9%
Nano AV	V/S	70.2%	52.3%	23.5%	48.0%	53.1–60.0%

Table B.1 includes all 448 Edge and 1,152 Nano layer/head units. More than half show greater-than-twofold imbalance for generated action/video and sound/video relations: 53.6% on Edge policy, 52.3% on Nano policy, and 52.3% on Nano AV. These units carry approximately 48–50% of the corresponding pair mass. Directional structure identifies candidate repair locations; its output benefit is tested below.

Depth profiles. The profiles in Figure 3 aggregate the same 84 BF16 trajectories, weighting DM query roles by token count before averaging seeds and cases. From the shallowest to deepest layer quarters, mean DM←AR mass decreases from 0.2123 to 0.1065 on Edge and 0.3861 to 0.1242 on Nano. The overall decline coexists with local increases: 15/27 and 21/35 adjacent transitions decrease. This motivates layer-specific selection rather than a fixed depth cutoff.

B.2 Matched-compute repair-tile selection

We evaluate selection on frozen BF16 trajectories from Edge I2V and policy workloads: two scenes per task, four seeds per scene, and 320 attention panels. Every policy retains all base blocks and applies the same first-order bilateral FP4 corrections to selected QK/PV tiles. We measure the complete QK–softmax–PV output before output projection; diagnostic outputs do not alter the trajectory.

Structure maps a frozen directed-role/layer/head/phase attention prior to the current token layout. Random samples the same number of tiles uniformly; Window uses sequence locality; Mass oracle ranks tiles by current BF16 attention mass. The prior is fitted on development trajectories, while the oracle uses current probabilities. This control evaluates the prior alone; deployment also scores current inputs (Appendix B.3).

We compare 25% and 50% repair coverage, corresponding to $\beta_{\mathrm{attn}} = 1.5/2.0$. All selectors issue the same padded FP4 repair MMA work at each budget. Masks are prepared outside the diagnostic operator; oracle scoring work is separate from this repair budget. Panel NMSE is squared output error divided by squared BF16 output norm, averaged over panels and requests. Paired intervals resample seeds within scenes.

Table B.2: Paired local NMSE differences (Structure minus comparator, $\times 10^{-3}$). Intervals resample seeds within scenes; adjusted intervals retain the frozen six-comparison family per cohort.

Edge task	β_{attn}	Comparator	Mean	95% CI	Familywise 95% CI
I2V	1.5	Random	-10.875	[-10.976, -10.789]	[-11.013, -10.770]
I2V	1.5	Window	-2.100	[-2.920, -1.330]	[-3.190, -1.095]
I2V	2.0	Random	-9.834	[-9.939, -9.733]	[-9.973, -9.707]
I2V	2.0	Window	-3.875	[-4.466, -3.326]	[-4.668, -3.170]
Policy	1.5	Random	-10.403	[-10.637, -10.209]	[-10.715, -10.174]
Policy	1.5	Window	-2.996	[-3.217, -2.791]	[-3.294, -2.729]
Policy	2.0	Random	-10.783	[-11.073, -10.502]	[-11.196, -10.422]
Policy	2.0	Window	-5.857	[-6.023, -5.727]	[-6.079, -5.707]

Structure reduces mean local attention-output NMSE relative to Random and Window in all four settings under matched repair work (Tables 1 and B.2). Complete-generation quality is evaluated separately in Section 5.2.

B.3 Calibration and current-input selection

Offline preparation. The deployment policies are calibrated offline and frozen before benchmark evaluation. Calibration determines compact weight-residual supports and phase-/CFG-specific quotas by evaluating actual repaired outputs against same-input BF16 references. W0 and W1 supports remain fixed throughout each request. Current inputs rank admissible activation and attention repair blocks within these quotas; cross products are disabled in both policies.

The $1.5\times$ budget. The limited budget retains compact linear bilateral repair and partial, head-specific attention repair. Linear selection scores the current activation residual using offline weight sensitivity. Attention follows calibrated per-head QK/PV quotas and current-input scores based on sampled attention mass and V-tile magnitude.

The $2\times$ budget. The larger budget expands linear repair support and enables full first-order bilateral repair for denoising attention. Offline search refines weight prescaling, component scales, and W1 locations using actual operator error. Weight supports stay fixed; activation blocks use the current residual. QK repair precedes softmax, and PV uses its resulting probabilities.

In both policies, selection controls which residual tiles are prepared and which dense FP4 correction products execute, while retaining the entire base W4A4 path. Budget accounting includes selector MMA, GQA support unions, and tail padding; no prior-step output is reused.

C Evaluation Protocols and Additional Results

C.1 Dataset coverage and fidelity metrics

The evaluation covers 650 RBench and 1,044 PAI-Bench-G scenes per model, with two paired seeds per scene. All methods share the scene/seed inventory and generation settings in Section 5.1, with CFG 6 and flow shift 3. Table C.1 summarizes coverage. This full-benchmark inventory retains previously used development scenes.

Table C.1: Completed generation coverage. Each scene has two paired seeds; candidate pairs exclude BF16 self-reference. Task scoring reuses the videos.

Model	Benchmark	Scenes	BF16 videos	All videos	Candidate pairs
Edge	RBench	650	1,300	11,700	10,400
Edge	PAI-Bench-G	1,044	2,088	18,792	16,704
Nano	RBench	650	1,300	11,700	10,400
Nano	PAI-Bench-G	1,044	2,088	18,792	16,704

Metrics use raw decoded float32 RGB before video compression and include all 61 frames, including the conditioned first frame. LPIPS uses AlexNet v0.1; SSIM averages RGB channels and frames with a uniform 7×7 window, sample covariance, $K_1 = 0.01$, and $K_2 = 0.03$. PSNR uses clip-global float64 MSE at range one, then averages per-video dB values. Summaries average seeds, scenes within category, and categories equally (nine RBench, seven PAI-Bench-G). Table C.2 adds relative L_2 and equal-scene LPIPS to the main metrics.

Table C.2: Additional fidelity statistics: category-macro relative L_2 and equal-scene LPIPS. Both average the two seeds within scenes and include all 61 frames; lower is better.

	RBench		PAI-Bench-G	
Method	Rel. L_2	LPIPS (scene)	Rel. L_2	LPIPS (scene)
Cosmos3 Edge				
Native FP8	0.1920	0.1187	0.1657	0.1035
NVFP4-AWQ	0.2559	0.1766	0.2469	0.1810
NVFP4 W4A4	0.3169	0.3008	0.3048	0.2865
ARCQuant (1.5)	0.2687	0.1934	0.2487	0.1888
ARCQuant (2.0)	0.2577	0.1767	0.2390	0.1734
SharQ	0.2704	0.1912	0.2517	0.1880
ThunderWorld (1.5)	0.2669	0.1939	0.2502	0.1893
ThunderWorld (2.0)	0.2479	0.1711	0.2279	0.1633
Cosmos3 Nano				
Native FP8	0.1661	0.0940	0.1620	0.1020
NVFP4-AWQ	0.2007	0.1198	0.1928	0.1285
NVFP4 W4A4	0.3108	0.3568	0.3464	0.4021
ARCQuant (1.5)	0.2116	0.1340	0.2096	0.1487
ARCQuant (2.0)	0.2042	0.1251	0.2036	0.1405
SharQ	0.2097	0.1310	0.2048	0.1415
ThunderWorld (1.5)	0.2105	0.1386	0.2149	0.1593
ThunderWorld (2.0)	0.1944	0.1179	0.1906	0.1281

C.2 Paired fidelity comparisons

Table C.3 reports paired LPIPS differences from ARCQuant (2.0), using category-stratified scene bootstrap intervals and the frozen Bonferroni correction. At $\beta_{\text{req}} = 2.0$, mean LPIPS improves in all four model/benchmark settings, with three adjusted upper bounds below zero. The $1.5\times$ policy provides a lower-cost operating point with reduced fidelity.

Table C.3: Paired LPIPS differences versus ARCQuant (2.0) (negative is better). Scene-bootstrap intervals preserve category weighting; adjusted upper bounds use the frozen Bonferroni family of 12 comparisons.

Model / benchmark	β	ΔLPIPS	95% CI	Adjusted upper
Edge / RBench	1.5	+0.0152	$[+0.0109, +0.0195]$	+0.0214
Edge / RBench	2.0	-0.0059	$[-0.0104, -0.0015]$	+0.0006
Edge / PAI-Bench-G	1.5	+0.0164	$[+0.0131, +0.0196]$	+0.0211
Edge / PAI-Bench-G	2.0	-0.0103	$[-0.0138, -0.0069]$	-0.0053
Nano / RBench	1.5	+0.0130	$[+0.0091, +0.0169]$	+0.0188
Nano / RBench	2.0	-0.0070	$[-0.0108, -0.0032]$	-0.0014
Nano / PAI-Bench-G	1.5	+0.0197	$[+0.0168, +0.0226]$	+0.0239
Nano / PAI-Bench-G	2.0	-0.0103	$[-0.0133, -0.0074]$	-0.0061

C.3 Task-judge protocol and uncertainty

We use Qwen3-VL-8B-Instruct (Bai et al., 2025), served with its FP8 checkpoint, BF16 KV cache, and temperature zero. Opaque media IDs hide method and model labels. The judge receives the task and candidate video, without the BF16 reference or answer key.

RBench follows ReVidgen's task-category and embodiment task-adherence rubrics. Frames $[0, 12, 24, 36, 48, 60]$ form a chronological 3×2 grid; visual-reasoning chains are frozen across methods. Rubric arithmetic and the final clamp produce the 1–5 task-VQA score. PAI-Bench-G uses 5,636 official binary questions from 1,036 annotated scenes, with constrained YES/NO decoding and 16 uniformly sampled video frames. The eight scenes without questions receive separate prompt-adherence scores.

PAI-Bench-G questions are averaged within each video, then both benchmarks average seeds, scenes within category, and categories. Table C.4 gives category and overall means; Table C.5 gives paired 95% intervals from scene-stratified resampling. Both repair budgets improve over W4A4 in all four settings, also under the frozen multiple-comparison correction. The $2\times$ means exceed $1.5\times$ throughout, while their paired differences remain statistically inconclusive.

Table C.4: Task-category Qwen3-VL-8B-Instruct scores for all five methods (higher is better). FP4 denotes NVFP4 W4A4; gray columns are ThunderWorld (TW). Macro averages weight categories equally. Paired intervals appear in Table C.5.

RBench task-VQA (1–5)

	Edge					Nano				
Category	BF16	FP8	FP4	TW 1.5	TW 2.0	BF16	FP8	FP4	TW 1.5	TW 2.0
Manipulation	2.1730	2.1480	1.8420	1.9680	2.1050	2.0300	2.0760	1.2600	2.0430	2.1200
Long-horizon	2.1933	2.1504	1.1842	2.1352	2.2623	2.2676	2.2035	1.0000	2.1315	2.2099
Multi-entity	2.6000	2.5780	1.9630	2.5390	2.6410	2.9740	2.9600	1.2230	2.8900	2.8270
Spatial relations	4.3260	4.4650	3.8850	4.4860	4.2210	4.7180	4.7100	2.2170	4.6110	4.5410
Visual reasoning	2.6665	2.7605	1.2217	2.5076	2.4884	2.3277	2.3626	1.0134	2.3561	2.4774
Single arm	3.2150	3.1400	2.0300	3.0400	3.0350	3.2100	3.2300	1.2100	3.1050	3.1850
Dual arm	3.0300	2.9650	1.8450	2.7700	2.9000	2.4050	2.3300	1.1700	2.2950	2.4100
Humanoid	2.7250	2.7400	2.0450	2.5600	2.5800	2.9600	2.9450	1.6450	3.0050	2.9650
Quadruped	2.8550	2.8450	2.3650	2.7450	2.8450	3.1000	2.9600	1.8450	2.9750	2.9650
Macro average	2.8649	2.8658	2.0423	2.7501	2.7864	2.8880	2.8641	1.3982	2.8235	2.8556

PAI-Bench-G Domain-VQA (%)

	Edge					Nano				
Category	BF16	FP8	FP4	TW 1.5	TW 2.0	BF16	FP8	FP4	TW 1.5	TW 2.0
AV	47.91	47.49	45.06	47.69	47.36	50.83	52.20	31.87	50.08	50.36
Common	81.64	79.73	77.53	78.27	79.45	86.15	88.34	70.78	89.36	88.18
Human	69.72	70.35	65.02	69.39	69.58	72.30	72.53	55.79	71.73	72.29
Industry	73.70	72.81	66.83	72.90	73.82	75.54	76.18	62.29	74.24	76.14
Misc.	83.74	83.54	80.18	82.84	85.12	84.99	85.36	67.76	84.84	85.05
Physics	83.59	84.63	80.16	84.05	84.01	87.41	86.26	73.80	87.32	86.73
Robot	65.89	64.84	58.72	65.56	65.09	66.99	66.69	50.53	65.29	68.01
Macro average	72.31	71.91	67.64	71.53	72.06	74.89	75.37	58.97	74.69	75.25

Table C.5: Paired ThunderWorld task-score differences from BF16 and NVFP4 W4A4, with scene-stratified 95% bootstrap intervals. RBench uses score units; PAI-Bench-G uses percentage points.

Model	Benchmark	β	Δ	95% CI
Reference: BF16				
Edge	RBench	1.5	-0.1148	$[-0.1882, -0.0419]$
Edge	RBench	2.0	-0.0785	$[-0.1504, -0.0064]$
Edge	PAI-Bench-G	1.5	-0.78	$[-1.62, +0.05]$
Edge	PAI-Bench-G	2.0	-0.25	$[-1.11, +0.59]$
Nano	RBench	1.5	-0.0645	$[-0.1269, -0.0022]$
Nano	RBench	2.0	-0.0325	$[-0.0971, +0.0332]$
Nano	PAI-Bench-G	1.5	-0.19	$[-0.93, +0.58]$
Nano	PAI-Bench-G	2.0	$+0.37$	$[-0.37, +1.13]$
Reference: NVFP4 W4A4				
Edge	RBench	1.5	$+0.7078$	$[+0.6301, +0.7859]$
Edge	RBench	2.0	$+0.7441$	$[+0.6571, +0.8305]$
Edge	PAI-Bench-G	1.5	$+3.89$	$[+2.89, +4.88]$
Edge	PAI-Bench-G	2.0	$+4.42$	$[+3.39, +5.44]$
Nano	RBench	1.5	$+1.4254$	$[+1.3404, +1.5124]$
Nano	RBench	2.0	$+1.4574$	$[+1.3710, +1.5476]$
Nano	PAI-Bench-G	1.5	$+15.72$	$[+14.22, +17.24]$
Nano	PAI-Bench-G	2.0	$+16.28$	$[+14.78, +17.78]$

D RUNTIME, STORAGE, AND EXECUTION DETAILS

D.1 REQUEST TIMING AND THROUGHPUT

Table D.1 reports warm batch-one complete-request latency with debug disabled and the generation settings in Section 5.1. Timing includes input processing, AR prefill, denoising, decoding, and CPU output; loading, offline preparation, warmup, Graph capture, and file serialization are excluded. Figure 6 converts median latency to throughput as 60 newly generated frames per request. Thor uses MAXN mode. The measurements retain device-specific diagnostic cohorts, including shared-GPU PRO 6000 Edge records.

Table D.1: Complete-request latency (median seconds) of native FP4 GEMM implementations. Device-specific timing cohorts and boundaries appear in Appendix D.

	PRO 6000		AGX Thor	
Method	Edge	Nano	Edge	Nano
NVFP4 W4A4	1.479	3.241	6.631	15.090
ARCQuant (1.5)	2.399	4.741	13.950	30.194
ARCQuant (2.0)	2.530	5.312	14.734	33.362
SharQ	2.365	5.323	13.721	32.544
ThunderWorld (1.5)	1.559	4.001	7.176	21.187
ThunderWorld (2.0)	1.686	4.559	7.857	24.465

The $1.5\times/2\times$ policies retain 71.2–94.9%/61.7–87.8% of W4A4 throughput, respectively, exchanging additional repair work for the fidelity gains in Section 5.2.

D.2 PACKING, EXECUTION, AND PERSISTENT WEIGHT STORAGE

Baseline operators and budget scope. ARCQuant fixes channel reordering indices and S residual channels offline (Meng et al., 2026a). Online preparation fuses reordering, normalization, quantization, and residual augmentation. Duplicating the corresponding W_0 columns permits one dense FP4 GEMM of reduction dimension $K + S$ to compute the base and activation correction with shared weight codes. ARCQuant controls repair work through layer-specific S. Our Cosmos adapters set $S/K = 0.5$ or 1, giving the nominal labels $(1.5)/(2.0)$ before padding. ThunderWorld's labels cap padded MMA across the complete projection/FFN and attention request.

SharQ selects two adjacent-element pairs from each group of eight for its NVFP4 sparse backbone, then quantizes the activation residual relative to that represented backbone (Meng et al., 2026b). Sparse FP4 and dense residual GEMMs share weight codes with path-specific scale layouts. Online selection, compression, and residual construction are fused; residual execution accumulates into the existing output. This recovers activation sparsification and quantization error using shared quantized weights. Our comparison uses these native kernels in the recorded Cosmos adapters.

Selected dense repair. Selected residual tiles preserve their encoded values during packing. W1 storage contains only the required support union; base and correction products share FP32 accumulation before output. Selective activation packing reduces writes, while the current implementation retains full activation-buffer extents. Selection skips correction tiles without sparsifying the base matrix or imposing a 4:8 pattern within executed tiles.

Table 3 counts AR/DM attention Q/K/V/O and FFN weights, including codes, scales, indices, and backend layouts. AR attention remains BF16, adding 0.656 GiB on Edge and 2.813 GiB on Nano to the quantized subset in Figure D.1. That subset comprises DM projections/FFNs and base-only W4A4 AR FFNs. ThunderWorld $(1.5)/(2)$ uses 1.33/1.37 GiB on Edge and 6.68/7.05 GiB on Nano, 39–42% below FP8 over the same weights. Higher budgets add selected W1 blocks rather than a full residual plane. These weight totals exclude embeddings, auxiliary modules, VAE, and runtime activation buffers.

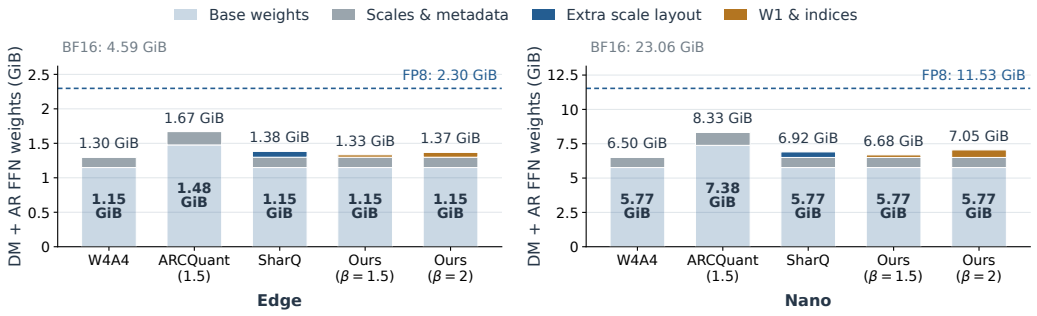


Figure D.1: **Persistent weight storage for DM projections/FFNs and AR FFNs (GiB).** Stacks separate base weights, scales and metadata, extra scale layouts, and W_1 with indices. Dashed lines mark FP8; annotations give BF16. FP4 methods use base-only W4A4 AR FFNs. Runtime buffers are excluded.

D.3 GPU EXECUTION BREAKDOWN AND FP4 ATTENTION COVERAGE

Figure D.2 groups cumulative GPU kernel time into projection/FFN, attention, quantization/preparation, and peripheral operation families. Fused preparation and attention remain indivisible; *Others* identifies standalone selection/support kernels. CPU scheduling is outside this GPU profile. Request latency is measured separately without profiling.

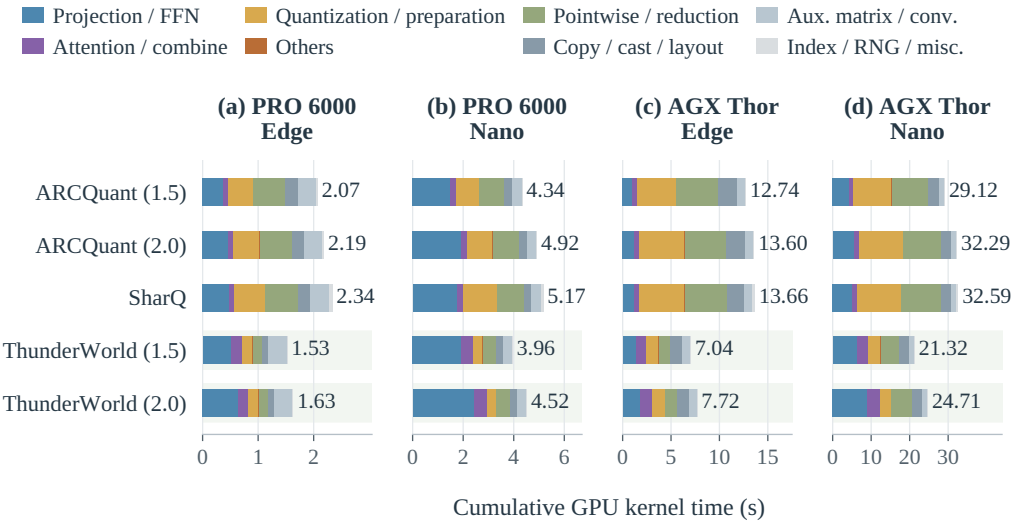


Figure D.2: Cumulative GPU kernel time with independent horizontal scales. Quantization / preparation includes fused operators; Others denotes standalone selection/support kernels. Request latency is reported separately in Table D.1.

At 2×, ThunderWorld's lower combined preparation and peripheral costs offset longer projection/FFN and attention kernel time than ARCQuant (2.0) and SharQ across all four panels. The baselines use native FP4 projection/FFN and BF16 QK/PV attention. ThunderWorld uses FP4 base/residual products for denoising QK/PV, with higher-precision softmax, accumulation, and causal AR attention. Its current FP4 attention is slower than baseline BF16 attention; preparation and surrounding operations account for the observed request gains, leaving attention as the next kernel optimization target.

E Additional Qualitative Comparisons

Figures E.1–E.5 complement Table 2 with ten illustrative scenes across robot manipulation, physical interactions, and everyday settings. Methods share the input and seed for each scene; columns show frames 30 and 60 of the saved raw generations. All methods use the frozen configurations in Section 5.1. Gray rows identify ThunderWorld.

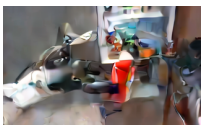
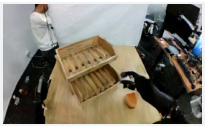
	Juice carton retrieval		Orange placement	
Method	Frame 30	Frame 60	Frame 30	Frame 60
BF16				
Native FP8				
NVFP4-AWQ				
NVFP4 W4A4				
ARCQuant (1.5)				
ARCQuant (2.0)				
SharQ				
ThunderWorld β = 1.5				
ThunderWorld β = 2.0				

Figure E.1: Additional RBench generations from Cosmos3 Edge: juice carton retrieval (left) and orange placement on a lower tray (right). Each scene is shown at frames 30 and 60.

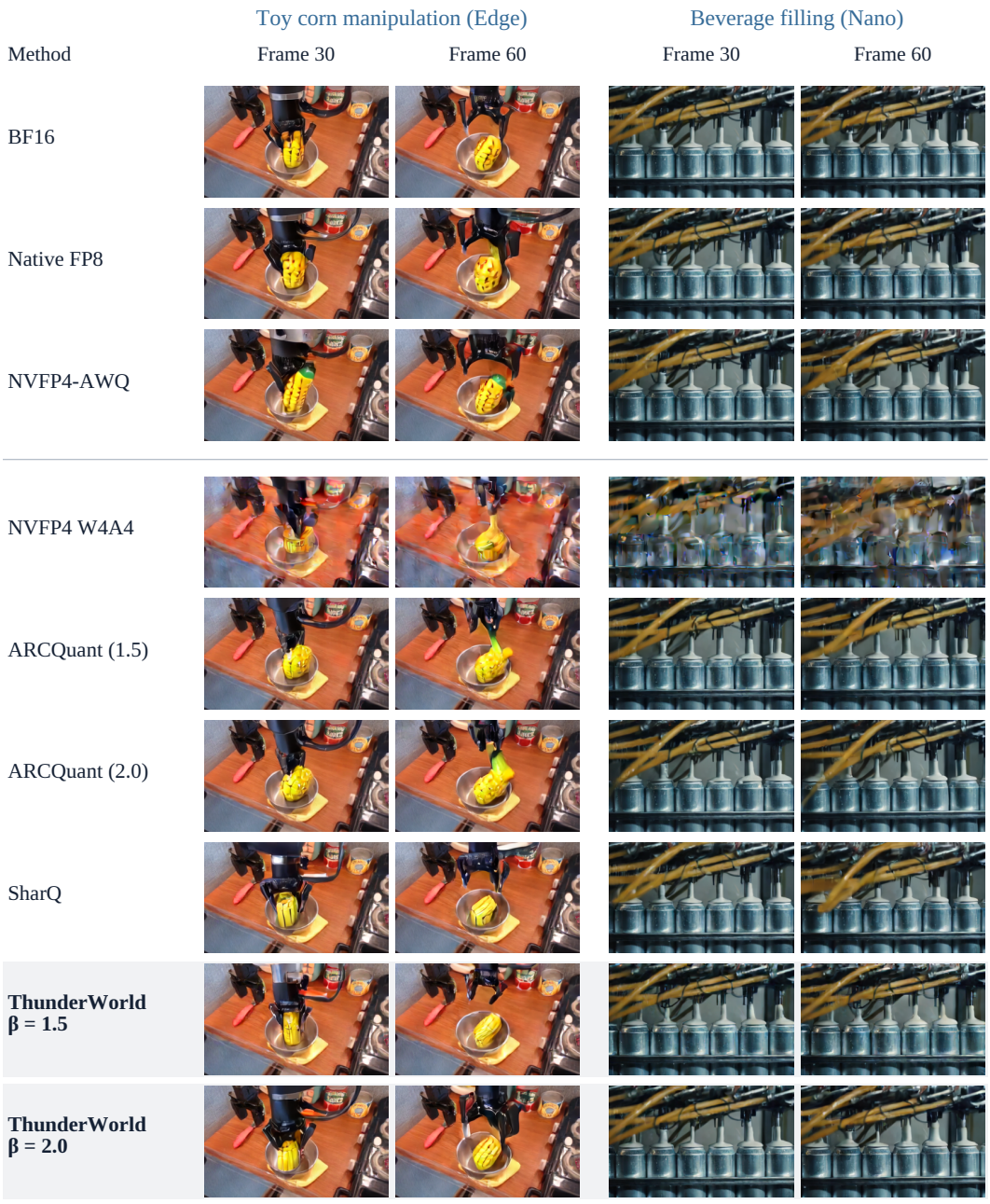


Figure E.2: Additional PAI-Bench-G generations: toy corn manipulation from Cosmos3 Edge (left) and beverage filling from Cosmos3 Nano (right). Each scene is shown at frames 30 and 60.



Figure E.3: Additional RBench generations from Cosmos3 Nano: ball transfer (left) and box lifting (right). Each scene is shown at frames 30 and 60.

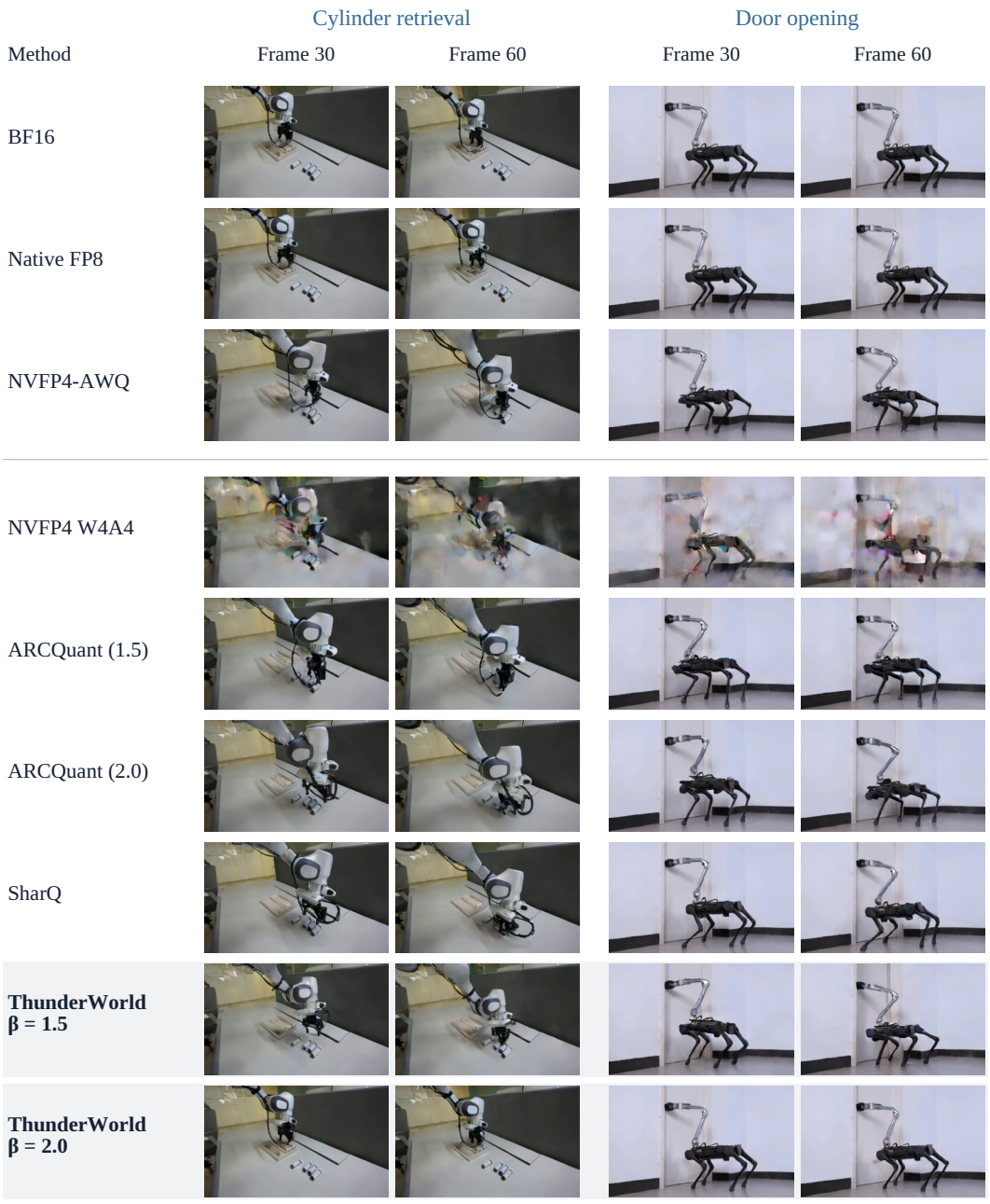


Figure E.4: Additional RBench generations from Cosmos3 Nano: retrieving a cylinder from a container (left) and a quadruped robot pushing a door open (right). Each scene is shown at frames 30 and 60.

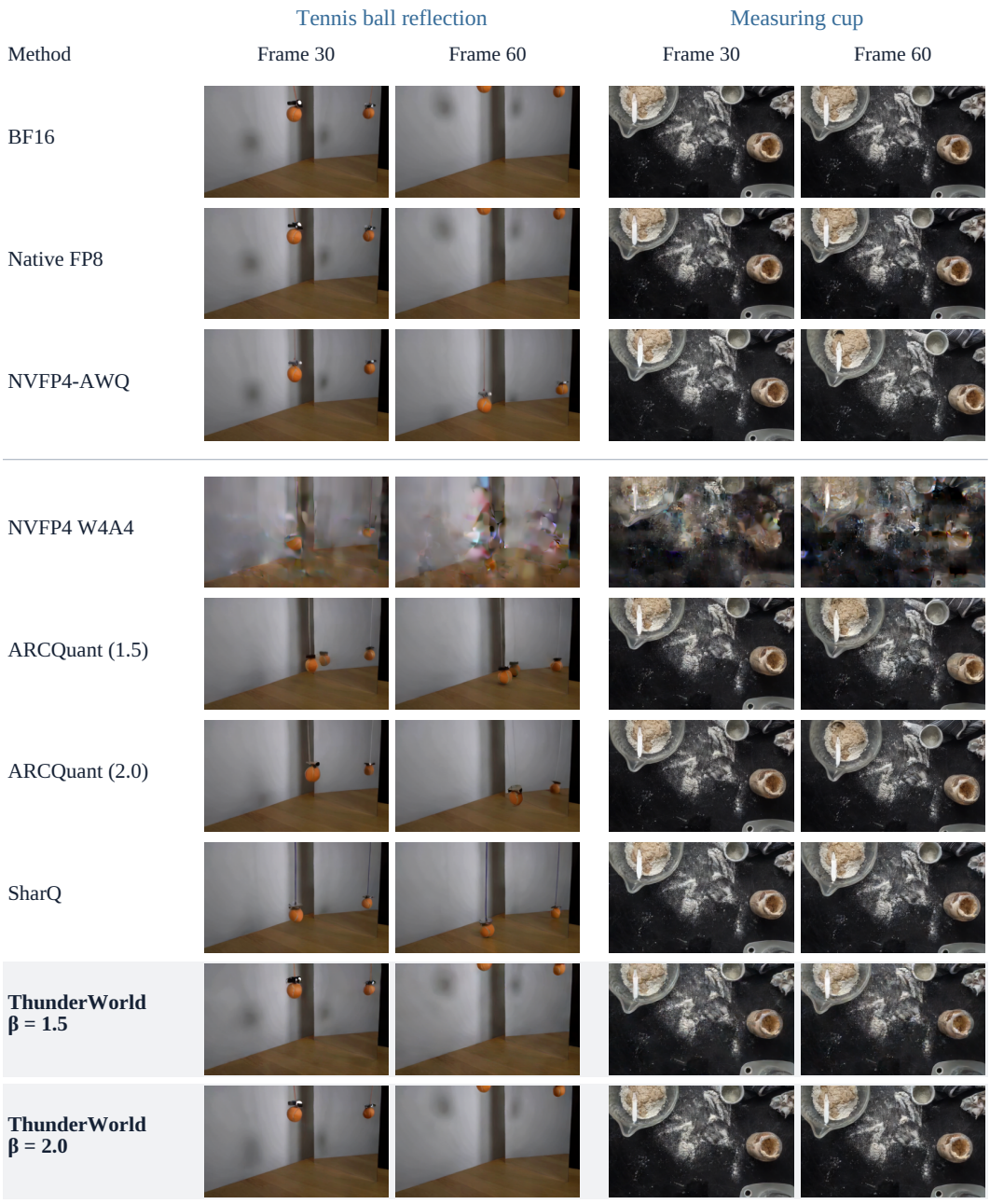


Figure E.5: Additional PAI-Bench-G generations from Cosmos3 Nano: a suspended tennis ball and its mirror reflection (left) and a measuring cup on a flour-covered surface (right). Each scene is shown at frames 30 and 60.