SAMPLE MIDWAY, CONDITION DEEPER: ONE-PASS AUTOREGRESSIVE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete-token video generators sample a clip as a grid of thousands of codes, and raster decoding runs one full transformer pass per code (174 s per UCF101 clip on an A100). Parallel decoders sample a group of codes per pass, but the codes in a group are drawn independently, so a choice that the condition leaves open, such as the phase of a pedaling motion, can be drawn differently at different positions. Before a code is drawn, every hidden state is a deterministic function of the condition, so only the drawn code itself can tell another position which option was taken, and this code can re-enter the network at the layer where it was drawn. We propose IntraAR, which samples successive token groups at successively deeper layers of one forward pass. After a group is sampled at layer ℓ_s, the embedding of each drawn code is added to its hidden state, the row is frozen as key–value memory, and the remaining layers predict the next groups conditioned on it. One pass thus performs S sampling steps with an exact likelihood. We show that sampling a group earlier pays off when the information it passes to later groups exceeds the fitting error caused by its shallower prediction. On five video benchmarks, IntraAR lowers batch-1 latency by 24.5–41.5$\times$ relative to matched raster decoding while I3D-FVD changes by -2.26 to $+1.83$; on three ImageNet backbones, latency falls by 11.2–29.8$\times$ at FID increases of 0.14–0.28. Removing only the embedding of the drawn code, with identical groups and computation, raises UCF101 I3D-FVD from 195.6 to 269.6.

1 INTRODUCTION

Discrete-token video generators such as OmniLM, VideoGPT, and TATS encode a clip into a spatiotemporal grid of codes and generate that grid with a transformer (Wang et al., 2024; Yan et al., 2021; Ge et al., 2022). A class-conditioned UCF101 clip becomes a $5 \times 32 \times 32$ grid of 5,120 codes. Raster decoding samples one code per forward pass through all layers; key–value caching removes recomputation of earlier codes, yet the 5,120 passes remain sequential, and a 24-layer OmniLM needs 174 s per clip on an A100 (Table 1).

Parallel decoders reduce the number of passes by sampling a group of codes in each pass. PAR and NAR group spatially distant or neighboring positions, and LPD and ZipAR choose groups by spatial locality (Wang et al., 2025; He et al., 2025b; Zhang et al., 2026; He et al., 2025a). Within a group, all categorical distributions are computed before any code of the group is drawn, so the draws are independent. This matters whenever the condition leaves a choice open. The label "Biking" does not specify where the pedal is in its rotation, yet the codes that depict the pedal, the leg, and the next frame must agree on one phase. When such codes share a group, each can draw a different phase (Figure 1b). On UCF101, raising PAR's group size from 4 to 16 reduces the latency from 54.4 to 17.2 s and increases I3D-FVD from 204.7 to 223.6. Parallel decoders thus exchange coherence for fewer passes.

What exactly is missing in a group? Before the group is drawn, the hidden state of every position is a deterministic function of the condition and of the earlier codes. More layers or bidirectional attention make these states more informative, yet no computation can tell one position which option another position will draw, because that draw has not happened yet. Two draws become coupled only when one conditions on the realized value of the other. Group autoregression provides this value at the input of the next forward pass, so coupling S groups costs S complete passes.

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.

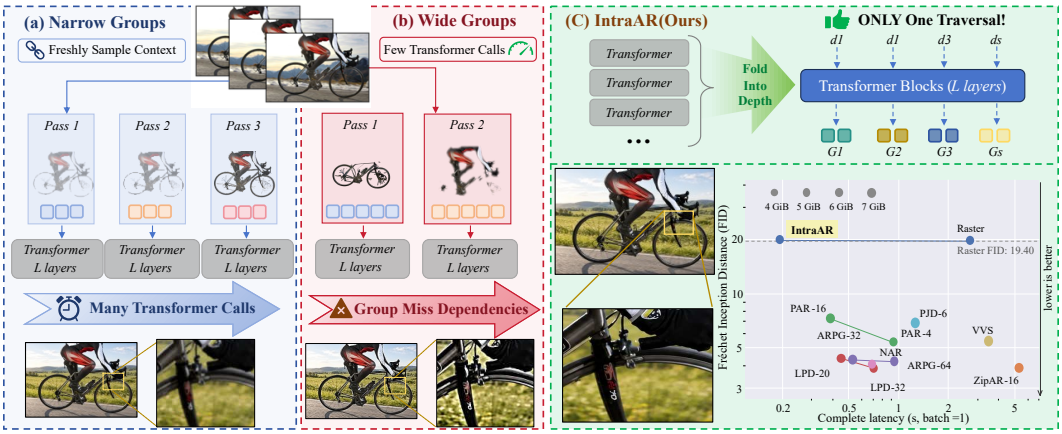


Figure 1: Where a drawn code re-enters the network. (a) With narrow groups, each drawn group re-enters at the first layer of a new forward pass, so the clip needs many passes. (b) Wide groups need few passes, but the codes of one group are drawn independently and can disagree (zoomed wheel). (c) IntraAR samples successive groups at increasing depths of one pass, so deeper layers condition on earlier draws. Right: FID and complete batch-1 latency on ImageNet/LlamaGen-XL; marker size is peak memory. External methods use their own guidance scales (Table 3).

The drawn code can instead re-enter at the layer where it was drawn. A code sampled from the hidden state at an intermediate layer ℓ can be embedded and written back at that same layer, after which layers $\ell+1, \ldots, L$ predict the other positions with the drawn code in view. The depth of one forward pass can thus host a sequence of sampling steps (Figure 1c), which we call *intra-forward autoregression* (IntraAR). After P layers that process the whole grid, IntraAR samples group G_1 at layer ℓ_1, adds the embedding of each drawn code to its hidden state, and freezes the resulting row as key–value memory. Deeper layers update the remaining rows while attending to the frozen ones, sample G_2 at layer $\ell_2 > \ell_1$, and so on until the last group is sampled at layer L. One pass thus performs S sampling steps, each conditioned on all earlier draws.

Sampling inside the pass creates a trade-off that repeated passes do not have. A group drawn at layer ℓ_s is predicted with only ℓ_s layers, while every later group gains access to its draw. We analyze this trade-off exactly: the excess negative log-likelihood (NLL) splits into dependence left within groups, information lost in memory, and fitting error (Theorem 1); an early draw pays off exactly when the information it gives to later groups exceeds the added fitting error of its shallower prediction (Equation (8)); and depth caps the number of sampling steps at $L - P$ (Theorem 2).

Experiments test these statements directly; for example, removing only the embedding of the drawn code, at identical computation, raises UCF101 I3D-FVD from 195.6 to 269.6. Our contributions are:

- IntraAR, which samples token groups at successive depths of one forward pass and writes each draw back as frozen key–value memory, adding $3d$ parameters for width d and keeping an exact, normalized likelihood.
- An analysis of the excess NLL that states exactly when an earlier draw helps and how depth bounds the number of sampling steps.
- Matched experiments on five video and three image generators: relative to raster decoding, IntraAR reduces latency by 24.5–41.5× on video (I3D-FVD -2.26 to $+1.83$) and 11.2–29.8× on ImageNet (FID $+0.14$ to $+0.28$), with the lowest latency among all compared decoders on every video benchmark.

2 Related Work

We organize prior work by where a drawn code re-enters the network.

Drawn codes re-enter at the next forward pass. Raster generators over discrete tokens, such as LlamaGen, RAR, OmniLM, VideoGPT, and TATS, feed every drawn code into the input of the next pass (Sun et al., 2024; Yu et al., 2025; Wang et al., 2024; Yan et al., 2021; Ge et al., 2022), and better

tokenizers have made this recipe competitive (Yu et al., 2024). VAR samples a whole scale per pass (Tian et al., 2024); PAR, NAR, LPD, ZipAR, and ARPG sample groups of positions chosen to be distant, neighboring, local, or random (Wang et al., 2025; He et al., 2025b; Zhang et al., 2026; He et al., 2025a; Li et al., 2026a); block diffusion denoises a block in parallel within an autoregressive block order (Arriola et al., 2025). These methods lower the number of passes by enlarging the unit drawn per pass, and each unit still waits for a complete new pass before later units can see it. IntraAR keeps many small groups and lets their draws re-enter within the same pass.

Independent drafts are corrected in further passes. MaskGIT and MAGVIT sample all masked codes in parallel and re-mask the least confident ones for the next iteration (Chang et al., 2022; Yu et al., 2023). PJD refines spatial drafts with Jacobi updates, and VVS verifies speculative drafts while skipping part of the verification (Liao et al., 2026; Dong et al., 2026). Coupling arises from the extra passes that inspect a draft. In IntraAR each code is drawn once, and its draw is coupled to earlier draws already when it is made.

Information exchange inside one pass. DnD predicts the successive residual codes of one spatial position along depth, while positions remain autoregressive across passes (Chen et al., 2025). Early-exit decoding stops the computation of one token at an intermediate layer (Schuster et al., 2022), and DMTD reuses late layers to emit several tokens per cycle and refills earlier-layer caches afterwards (Luo et al., 2026). PTP feeds random auxiliary inputs into the model, and CoDD couples output distributions in diffusion language models (Draxler et al., 2026; Li et al., 2026b); both escape the independent-draw product through shared randomness or output structure. IntraAR draws different positions of the visual grid at increasing depths of one pass, writes each draw back as frozen memory, and completes the grid without replaying layers or refilling caches.

3 Why Grouped Sampling Loses Coherence

Setup. A fixed tokenizer maps a video or an image to a grid of discrete codes. We write C for everything the generator conditions on, namely a class label, observed frames, or nothing, and $Z = (Z_1, \dots, Z_N)$ with $Z_i \in [V] = \{1, \dots, V\}$ for the codes to generate; codes of observed frames belong to C. Every position keeps its native spatial or spatiotemporal coordinate. An ordered partition $\mathcal{G} = (G_1, \dots, G_S)$ of $[N]$ splits the grid into groups, and $Z_{<s}$ denotes the codes of $G_1 \cup \cdots \cup G_{s-1}$. Group autoregression samples

$$q(Z \mid C) = \prod_{s=1}^{S} \prod_{i \in G_s} q_i\big(Z_i \mid C, Z_{<s}\big). \tag{1}$$

Raster decoding is the case $S = N$, and PAR and NAR use fewer, larger groups. The outer product is sequential: a transformer with L blocks runs one complete pass per group, that is, SL sequential block computations. The inner product is the price of parallelism, because the codes in one group are drawn independently.

What independent draws cannot represent. Let $\mathcal{H}_s = (C, Z_{<s})$ be the information available to group G_s. The joint uncertainty of the group falls short of the sum of its per-position uncertainties by the conditional total correlation

$$\mathrm{TC}(Z_{G_s} \mid \mathcal{H}_s) = \sum_{i \in G_s} H(Z_i \mid \mathcal{H}_s) - H(Z_{G_s} \mid \mathcal{H}_s) \;\geq\; 0. \tag{2}$$

Since the hidden states that produce $q_i(\cdot \mid \mathcal{H}_s)$ are deterministic functions of $\mathcal{H}_s$, the product in (1) has expected NLL at least $\sum_{i \in G_s} H(Z_i \mid \mathcal{H}_s)$ on this group, which exceeds the true joint entropy by exactly $\mathrm{TC}(Z_{G_s} \mid \mathcal{H}_s)$. This gap is independent of the depth, width, and attention pattern of the network. In the biking example, let the pedal phase be a fair bit U that k codes in one group must share. Then $\mathrm{TC} = (k-1)\log 2$, and the k independent draws agree on one phase with probability only 2^{1-k}.

What removes the gap. The gap vanishes only if some draws condition on the realized values of others: if one code in the example is drawn first and the other $k-1$ positions see it, they copy its phase and TC drops to zero. Group autoregression achieves this by starting a new pass. The next section keeps the realized value inside the pass: a code drawn at layer ℓ is written back at layer ℓ, and the remaining layers predict later groups conditioned on it.

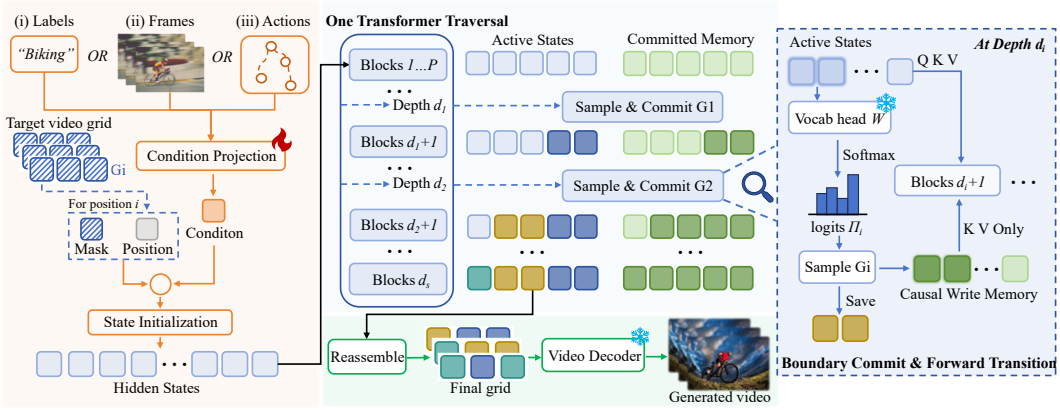


Figure 2: One IntraAR forward pass. Left: a class label, observed frames, or actions enter through the backbone's native conditioning pathway, and every target position starts from a learned mask embedding at its native coordinate. Middle: blocks $1, \dots, P$ process the whole grid; at depth ℓ_s (written d_s in the diagram) group G_s is sampled and its rows are frozen as memory (green), while the remaining rows keep being updated. Right: at a commitment layer, the shared vocabulary head gives the distribution π_i, the drawn code is embedded and written, and deeper blocks read the written row only through keys and values. After layer L, the grid is reassembled and decoded by the fixed tokenizer decoder.

4 IntraAR: Sampling at Successive Depths of One Forward Pass

IntraAR generates the whole grid in one forward pass of an L-block transformer. A schedule fixes a prefix depth P and commitment layers $P < \ell_1 < \cdots < \ell_S = L$; at layer ℓ_s the pass samples group G_s and *commits* it, meaning that its codes are written into the network and never revised (Figure 2). By default the groups are equal contiguous chunks of the backbone's raster order and $\ell_s = P + s$. Every sampling step inside the pass must meet two requirements: **(R1)** the group is predicted from enough context to make a good draw, although it has fewer layers than a full pass, and **(R2)** its realized codes reach every later prediction intact. Shared prefix layers serve R1; the sample write and the frozen memory serve R2. Algorithm 1 (Appendix A) lists the complete procedure.

4.1 Shared Prefix Layers

Each target position i starts from a learned mask embedding $\boldsymbol{e}_{[\mathrm{M}]}$ placed at its native coordinate, and C enters through the backbone's native conditioning pathway. In blocks $1, \dots, P$ all target positions attend to each other and to the condition. This computation is shared by every group, so the first group, drawn at layer $\ell_1 = P + 1$, is predicted from states that already reflect the layout C implies for the whole clip. Removing the prefix hurts mainly the earliest groups (Section 5.3), as R1 predicts.

4.2 Sampling and Writing a Group

Let $\boldsymbol{h}_i^{(\ell)} \in \mathbb{R}^d$ be the hidden state of position i after block ℓ. At layer ℓ_s, the backbone's own vocabulary head turns each state of G_s into a categorical distribution, the codes are drawn independently, and each draw is written back:

$$\pi_i = \mathrm{softmax}\big(\boldsymbol{W}_{\mathrm{out}}\,\mathrm{RMSNorm}_{\mathrm{out}}(\boldsymbol{h}_i^{(\ell_s)})\big), \qquad \widehat{Z}_i \sim \mathrm{Cat}(\pi_i), \qquad i \in G_s. \tag{3}$$
$$\boldsymbol{m}_i = \mathrm{RMSNorm}_{\mathrm{mem}}\big(\boldsymbol{h}_i^{(\ell_s)} + \boldsymbol{e}_{\widehat{Z}_i}\big),$$

Here $\boldsymbol{e}_v$ is the backbone's input embedding of code v. The two summands of the written row $\boldsymbol{m}_i$ carry different information. The hidden state records the context in which the draw was made, and the code embedding records which option was drawn, which is exactly the information that no deterministic state contains (Section 3). The memory normalization puts the sum of a deep hidden state and an input embedding on a common scale. With nonzero normalization scales and distinct code embeddings, the write is injective in the drawn code, so the memory keeps every earlier draw recoverable (Proposition 2, Appendix B). The head matrix $\boldsymbol{W}_{\mathrm{out}}$ is shared by all S commitments, and the only new parameters are $\boldsymbol{e}_{[\mathrm{M}]}$ and the two normalization scales, $3d$ in total.

4.3 COMMITTED ROWS AS READ-ONLY MEMORY

A committed row has to keep representing the same draw for every deeper layer. IntraAR therefore freezes it. At block ℓ, let $\boldsymbol{H}_\ell$ stack the rows that are still active, that is, the positions of groups with $\ell_s \geq \ell$, and let $\boldsymbol{B}_\ell$ stack the condition rows and the memory rows $\boldsymbol{m}_j$ committed before ℓ. With $\boldsymbol{X}_\ell = [\boldsymbol{B}_\ell; \boldsymbol{H}_\ell]$, head width d_k, and the native normalizations and positional operations omitted, block ℓ computes

$$\widetilde{\boldsymbol{H}}_\ell = \boldsymbol{H}_\ell + \mathrm{softmax}\big(\boldsymbol{H}_\ell \boldsymbol{W}_\ell^Q (\boldsymbol{X}_\ell \boldsymbol{W}_\ell^K)^\top / \sqrt{d_k}\big)\, \boldsymbol{X}_\ell \boldsymbol{W}_\ell^V, \qquad \boldsymbol{H}_\ell \leftarrow \widetilde{\boldsymbol{H}}_\ell + \mathrm{MLP}_\ell(\widetilde{\boldsymbol{H}}_\ell), \quad (4)$$

Only active rows form queries and receive the attention and MLP updates; memory rows enter as keys and values, projected by each layer's own $\boldsymbol{W}_\ell^K, \boldsymbol{W}_\ell^V$ and placed at their original coordinates. Freezing thus preserves the injective record of (3) at every depth, and it also makes the pass cheaper: the number of query rows shrinks from N to $|G_S|$ as groups are committed. Since every state entering layer ℓ_s is a deterministic function of $(C, M_{<s})$, where $M_{<s} = (M_1, \ldots, M_{s-1})$ with $M_r = (\boldsymbol{m}_i)_{i \in G_r}$, the pass defines

$$q_\theta(Z \mid C) = \prod_{s=1}^{S} \prod_{i \in G_s} q_{\theta,i}\big(Z_i \mid C, M_{<s}\big), \qquad q_{\theta,i}(\cdot \mid C, M_{<s}) = \pi_i, \quad (5)$$

a normalized autoregressive distribution whose likelihood is evaluated exactly in one teacher-forced pass (Lemma 1).

Training. We continue training a pretrained raster generator with the same pass. At layer ℓ_s the logits of G_s are computed first, and the ground-truth codes are written afterwards through (3); hence no target enters its own prediction. With $M^\star_{<s}$ denoting the memory built from ground-truth codes, the loss is

$$\mathcal{L}(\theta) = -\,\mathbb{E}_{(C,Z)\sim p} \sum_{s=1}^{S} \sum_{i \in G_s} \log q_{\theta,i}\big(Z_i \mid C, M^\star_{<s}\big). \quad (6)$$

Written codes are data, so gradients reach earlier layers through the written hidden states and embeddings without a discrete gradient estimator. All S groups are trained in one forward and backward pass, as in raster training.

4.4 ANALYSIS: WHAT SAMPLING INSIDE THE PASS COSTS

We assume a finite vocabulary, nonempty groups, independent draws within a group, and deterministic writes $M_s = \phi_s(C, M_{<s}, Z_{G_s})$, with finite expected NLL. Let $\mathcal{E}(q) = \mathbb{E}_p[-\log q(Z \mid C)] - H_p(Z \mid C)$ denote the excess NLL of the grid. Information quantities are taken under the data distribution with teacher-forced writes, so $M_{<s}$ below is $M^\star_{<s}$; proofs are in Appendix B.

Theorem 1 (Dependence, memory, and fitting)**.** *The excess NLL of IntraAR splits into three nonnegative terms,*

$$\mathcal{E}(q_\theta) = \underbrace{\sum_s \mathrm{TC}(Z_{G_s} \mid C, Z_{<s})}_{\Gamma(\mathcal{G})} + \underbrace{\sum_{s,\, i \in G_s} I(Z_i; Z_{<s} \mid C, M_{<s})}_{\Delta_{\mathrm{mem}}} + \underbrace{\sum_{s,\, i \in G_s} \mathbb{E}\,\mathrm{KL}(p_i \,\|\, q_{\theta,i})}_{\mathcal{F}_\theta}, \quad (7)$$

where $p_i = p(Z_i \mid C, M_{<s})$ is the true conditional given the memory.

$\Gamma(\mathcal{G})$ is the within-group dependence of Equation (2), summed over groups; it shrinks as groups become finer. Δ_{mem} is the information about earlier codes that the memory drops, and it is zero for the injective write of Section 4.2. $\mathcal{F}_\theta$ is the error of fitting each conditional with the layers available to it, and this is where a shallow early commitment pays. Removing the code embedding therefore inflates Δ_{mem}, and removing the prefix inflates $\mathcal{F}_\theta$ of early groups.

When is an earlier draw worth its depth? Take two adjacent groups A, B with the same incoming history $\mathcal{H}$. The early schedule samples A, writes M_A, and then samples B; the merged schedule samples $A \cup B$ together at B's layer. Their losses on $A \cup B$ satisfy

$$\mathcal{L}_{\mathrm{early}} - \mathcal{L}_{\mathrm{merge}} = \big(\mathcal{F}_{\mathrm{early}} - \mathcal{F}_{\mathrm{merge}}\big) - \sum_{j \in B} I\big(Z_j; M_A \mid \mathcal{H}\big), \quad (8)$$

where $\mathcal{F}_{\mathrm{early}}, \mathcal{F}_{\mathrm{merge}}$ are the fitting terms of the two schedules on $A \cup B$. The early draw lowers the loss exactly when the information it passes to B exceeds the extra fitting error of predicting A with fewer layers. Section 5.4 measures the net effect.

Theorem 2 (Depth bounds the number of sampling steps)**.** *Let $\Gamma^*(s)$ be the smallest Γ over ordered partitions into s nonempty groups; Γ^* is nonincreasing in s. Since commitment layers are distinct and lie in $\{P+1, \ldots, L\}$, IntraAR has $S \le K = \min(N, L-P)$ and $\mathcal{E}(q_\theta) \ge \Gamma^*(K) + \Delta_{\mathrm{mem}} + \mathcal{F}_\theta$.*

More steps lower the dependence floor but shorten the depth of the earliest predictions; Section 5.4 varies S at fixed depth to measure both effects.

5 Experiments

5.1 Setup

Benchmarks and backbones. Five video settings cover class-conditioned UCF101 and future prediction on Kinetics-600 (K600) with OmniLM, first-frame-conditioned BAIR with VideoGPT, and unconditional Sky and Taichi with TATS-base (Wang et al., 2024; Yan et al., 2021; Ge et al., 2022); class-conditioned ImageNet with LlamaGen-XL, OmniLM, and RAR-L tests images (Sun et al., 2024; Yu et al., 2025) (Table 7). The matched reference is each backbone's native raster decoder (Raster), trained from the same weights on the same tokens as IntraAR; both sample at temperature 1 without guidance. IntraAR uses $S = 16$ groups ($S = 8$ on BAIR) with $\ell_s = P + s$. External decoders (PAR and NAR on all tasks; LPD, ARPG, ZipAR, PJD, and VVS on ImageNet) use their own sampling recipes and guidance (Appendix C.3).

Metrics. We report teacher-forced NLL; FID, sFID, Inception Score, precision, and recall for images; and FVD with I3D and VideoMAE-v2 features for videos (Wang et al., 2023; Ge et al., 2024). Latency is the complete batch-1 time on one A100, including tokenizer decoding; entries are three-seed means (Appendix C).

5.2 One Pass Versus Many Passes

Video. Table 1 compares the five matched pairs with PAR and NAR. IntraAR lowers batch-1 latency by 24.5–41.5× relative to Raster, most on UCF101, whose 5,120-code grid needs the most raster passes, while I3D-FVD moves by $+1.83$, $+0.96$, $+1.71$, -1.07, and -2.26 on UCF101, K600, BAIR, Sky, and Taichi. PAR shows the motivating trade-off: on UCF101, 16-token groups run 3.2× faster than 4-token groups and lose 18.9 I3D-FVD. IntraAR is faster than PAR-16 and has lower FVD than both PAR settings on all five tasks; NAR reaches lower I3D-FVD on UCF101 (with guidance), BAIR, and Sky at 1.8–1.9× IntraAR's latency. The tokenizer decoder becomes the main remaining cost: on UCF101, generation drops from 171.97 to 2.12 s while decoding stays at about 2.1 s (Table 11). IntraAR also has the highest batch-4 throughput on four of five tasks (Table 8).

Images. On ImageNet (Table 2), FID rises by 0.14, 0.28, and 0.23 for LlamaGen-XL, OmniLM, and RAR-L, while latency falls by 14.0, 29.8, and 11.2×, again following grid size. NLL stays within 0.04 nats/token of Raster (Table 9) although IntraAR draws 16 groups instead of 256 or 1,024 single codes, so the dependence left inside a group, $\Gamma(\mathcal{G})$, appears small. Among seven guided external decoders (Table 3), IntraAR has the lowest latency (0.19 s, 2.0× below PAR-16), the highest batch-4 throughput, and the lowest batch-4 memory.

Samples. Figures 3–5 show the problem of Section 3 solved. The first BAIR frame leaves open where the arm moves; each rollout picks one trajectory and keeps it in all later frames, and two futures differ around the arm and the objects it pushes while agreeing on the table and background

Table 1: Video generation across five datasets. Three-seed means; I3D/MAE are FVD, Lat. is batch-1 s/video. Backbones: OmniLM (Wang et al., 2024), VideoGPT (Yan et al., 2021), TATS-base (Ge et al., 2022). Raster and IntraAR are matched and unguided; PAR (Wang et al., 2025) and NAR (He et al., 2025b) use their own recipes (Appendix C.3). Bold: best per column.

Method	UCF101			K600			BAIR			Sky			Taichi		
	OmniLM			OmniLM			VideoGPT			TATS-base			TATS-base		
	I3D↓	MAE↓	Lat.↓	I3D↓	MAE↓	Lat.↓	I3D↓	MAE↓	Lat.↓	I3D↓	MAE↓	Lat.↓	I3D↓	MAE↓	Lat.↓
Raster	193.76	145.43	174.04	**35.76**	**54.97**	111.16	**111.16**	92.57	71.92	160.81	**136.76**	11.94	108.58	121.07	13.28
PAR-4 (Wang et al., 2025)	204.74	160.11	54.35	39.79	63.62	35.77	117.18	101.42	20.69	165.10	150.98	3.56	112.45	129.90	3.91
PAR-16 (Wang et al., 2025)	223.61	184.56	17.19	47.49	80.43	12.16	130.75	120.29	6.46	181.13	171.76	1.11	133.74	156.97	1.28
NAR (He et al., 2025b)	**184.53**	**138.19**	7.80	37.16	56.74	6.62	112.03	98.06	3.40	**151.07**	140.12	0.80	112.55	122.42	0.92
IntraAR	195.59	144.52	**4.19**	36.72	55.15	**3.81**	112.87	**91.56**	**1.94**	159.74	138.19	**0.42**	**106.32**	**119.09**	**0.54**



Figure 3: Class-conditioned UCF101 samples from IntraAR (FloorGymnastics, Diving, PoleVault; frames 0–16 of each clip). Pose and contact with the floor, water, or pole evolve consistently within each clip.

Table 2: Matched ImageNet decoding across three backbones (Sun et al., 2024; Wang et al., 2024; Yu et al., 2025). Unguided, temperature 1; three-seed means. Precision/recall in %, latency in batch-1 s/image, rate in batch-4 images/s.

Method	LlamaGen-XL 36L, $d = 1280$					OmniLM 24L, $d = 1536$					RAR-L 24L, $d = 1024$				
	FID↓	Prec.↑	Rec.↑	Lat.↓	Rate↑	FID↓	Prec.↑	Rec.↑	Lat.↓	Rate↑	FID↓	Prec.↑	Rec.↑	Lat.↓	Rate↑
Raster	**19.40**	**61.43**	66.19	2.72	1.15	**10.28**	**69.14**	**64.82**	17.75	0.20	**9.24**	**70.93**	67.96	1.67	1.87
IntraAR	19.54	61.40	**66.45**	**0.19**	**17.46**	10.56	69.00	64.54	**0.60**	**4.97**	9.47	70.72	**68.02**	**0.15**	**24.25**

(Figure 5). The randomness of a sample thus lies in the choices the first frame leaves open, and every frame follows the same choice.

5.3 What Couples the Groups?

Table 4: Image mechanism controls. Three-seed means. NLL: nats/token; latency: batch-1 s/image; peak: batch-4 GiB. sFID complements FID. Bold: best per model/metric. Models: Table 7.

Variant	NLL↓	FID↓	sFID↓	Lat.↓	Peak↓
ImageNet / LlamaGen-XL (36L, $d = 1280$)					
IntraAR	**5.63**	19.54	**8.78**	0.19	5.63
No sample write	6.03	29.70	11.45	0.19	5.63
No prefix	5.97	34.14	12.71	**0.12**	**5.52**
Dense queries	**5.63**	**19.53**	**8.78**	0.22	6.13
ImageNet / RAR-L (24L, $d = 1024$)					
IntraAR	**4.25**	9.47	**5.73**	0.15	4.04
No sample write	4.67	13.84	6.44	0.15	4.03
No prefix	4.44	11.15	6.07	**0.12**	**3.99**
Dense queries	**4.25**	**9.44**	**5.73**	0.17	4.32

Table 5: Video mechanism controls. Three-seed means. NLL: nats/token; I3D/MAE: FVD; latency: batch-1 s/video; peak: batch-4 GiB. Bold: best per task/metric. Models: Table 7.

Variant	NLL↓	I3D↓	MAE↓	Lat.↓	Peak↓
UCF101 / OmniLM (24L, $d = 1536$)					
IntraAR	**5.78**	195.59	144.52	4.19	43.55
No sample write	6.60	269.57	223.20	4.15	43.50
No prefix	6.12	249.26	196.81	**3.43**	**43.39**
Dense queries	**5.78**	**195.37**	**144.43**	5.13	50.71
K600 / OmniLM (24L, $d = 1536$)					
IntraAR	**4.33**	36.72	**55.15**	3.81	44.22
No sample write	4.67	42.91	70.59	3.74	44.18
No prefix	4.56	44.18	69.30	**3.11**	**43.98**
Dense queries	**4.33**	**36.71**	55.17	4.28	48.81

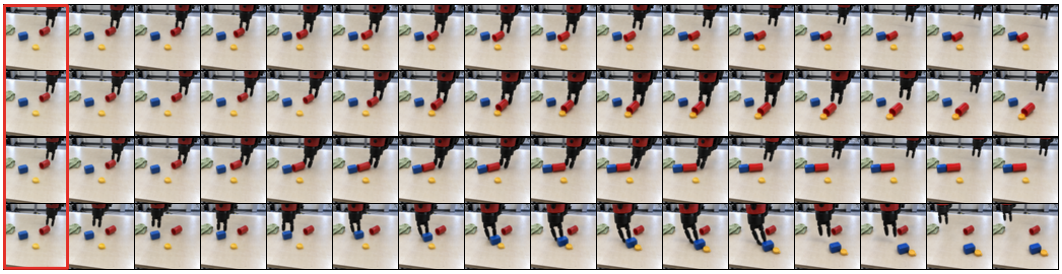


Figure 4: Four futures from one BAIR frame (red border). Each row is an IntraAR rollout; the rows choose different arm trajectories, and each row keeps its trajectory across all 15 frames.

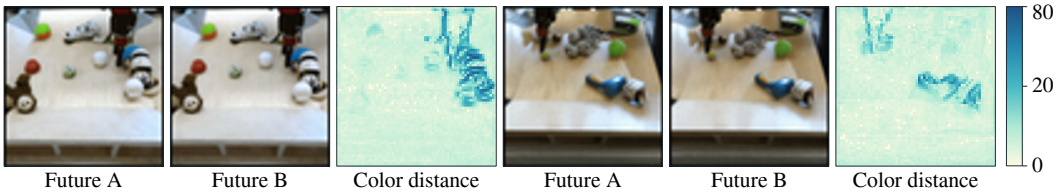


Figure 5: Where two BAIR futures differ. Each triplet shows the last frame of two IntraAR candidates A and B drawn from the same first frame, followed by their pixelwise CIE76 color distance $\|\mathrm{Lab}(A) - \mathrm{Lab}(B)\|_2$ on a shared 0–80 scale (square-root display). The two triplets use two different first frames.

Table 3: ImageNet/LlamaGen-XL: matched pair and seven external decoders. Three-seed means on the same grid and training exposure. CFG: guidance scale (1: unguided); latency in batch-1 s/image, rate in batch-4 images/s, B1/B4 peak GiB. Bold: best within each block. Recipes: Appendix C.3.

Method	CFG	Image quality					End-to-end cost		Peak memory	
		FID↓	sFID↓	IS↑	Prec.↑	Rec.↑	Lat.↓	Rate↑	B1↓	B4↓
Matched reference / LlamaGen-XL, 36 layers, width 1,280; no guidance or truncation										
Raster	1	**19.40**	8.87	**66.71**	**61.43**	66.19	2.72	1.15	**4.30**	5.70
IntraAR	1	19.54	**8.78**	66.59	61.40	**66.45**	**0.19**	**17.46**	4.37	**5.63**
Complete external recipes / LlamaGen-XL target, 36 layers, width 1,280; sampling varies by method										
PAR-4 (Wang et al., 2025)	1.5	5.32	7.33	166.37	75.43	57.95	0.94	3.89	**4.45**	6.15
PAR-16 (Wang et al., 2025)	1.5	7.15	8.23	162.86	77.14	54.63	**0.39**	**9.79**	4.77	7.06
NAR (He et al., 2025b)	2	4.01	6.83	239.76	82.05	54.56	0.70	5.58	4.72	6.55
LPD-20 (Zhang et al., 2026)	2	4.31	**6.29**	260.67	78.84	57.87	0.45	9.15	4.67	6.11
LPD-32 (Zhang et al., 2026)	2	**3.81**	6.60	255.71	77.84	**60.33**	0.71	5.88	4.63	**6.05**
ARPG-32 (Li et al., 2026a)	6.6	4.25	6.62	**269.40**	80.86	56.60	0.53	7.38	4.87	6.47
ARPG-64 (Li et al., 2026a)	5.3	4.15	6.41	265.74	79.83	58.82	0.95	4.02	4.81	6.42
ZipAR-16 (He et al., 2025a)	2	3.83	7.25	268.25	84.30	50.77	5.33	0.62	**4.45**	6.46
PJD-6 (Liao et al., 2026)	3.5	6.78	7.47	260.02	**87.86**	43.62	1.27	2.75	4.92	7.21
VVS (Dong et al., 2026)	1.75	5.38	7.48	177.27	79.63	55.33	3.50	0.99	4.77	6.84

The drawn code couples the groups. Tables 4 and 5 change one part of IntraAR at a time. *No sample write* replaces $\boldsymbol{e}_{\widehat{Z}_i}$ in (3) by $\boldsymbol{e}_{[\mathrm{M}]}$, so every written row is a deterministic function of C and, by Section 3, all codes are drawn independently at IntraAR's computation. The drawn code alone accounts for the difference: UCF101 NLL rises by 0.82 nats/token and I3D-FVD from 195.59 to 269.57 at unchanged latency, and FID rises from 19.54 to 29.70 on LlamaGen-XL and from 9.47 to 13.84 on RAR-L. Per-group losses (Figure 9) coincide at the first group, which has no draw to read, and separate afterwards; on video the gap widens up to the last group, because later groups are later frames whose content depends on the motion drawn earlier. In Figure 6 the two top-1 maps look alike, yet the full distributions differ by a total variation of up to 0.45, most in the rows committed last.

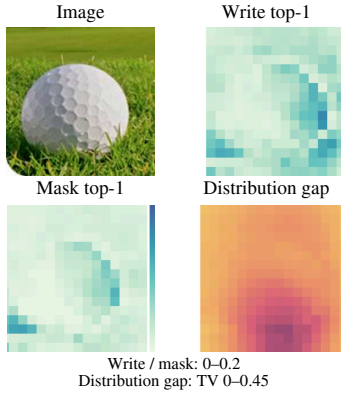


Figure 6: What the write changes. Top-1 probability under code and mask writes; bottom right: total variation of the full distributions.

The prefix serves the earliest groups. *No prefix* deletes blocks $1, \ldots, P$, so G_1 is drawn after a single block. Quality drops in all four settings (UCF101 I3D-FVD 249.26, LlamaGen-XL FID 34.14), and the loss rises most for the first groups (Figure 10). Per nat of NLL it costs more FID or FVD than removing the write on three of four settings (43 versus 25 FID points per nat on LlamaGen-XL). Teacher-forced NLL scores each group given ground-truth history, whereas at sampling time a poor early draw is inherited by every later group, so an error in the first groups weighs more on sample quality than its NLL suggests.

Freezing saves computation. *Dense queries* also updates committed rows and restores them: same samples, higher cost (UCF101: 5.13 versus 4.19 s and 50.71 versus 43.55 GiB at batch 4). The saving grows with the grid length, since frozen rows leave the query and MLP computation of every deeper layer; on RAR-L's 256-code grid the two executions are within 0.02 s. Each component thus acts where its requirement predicts: the prefix on the first groups (R1), the write on all later ones (R2), and freezing only on cost.

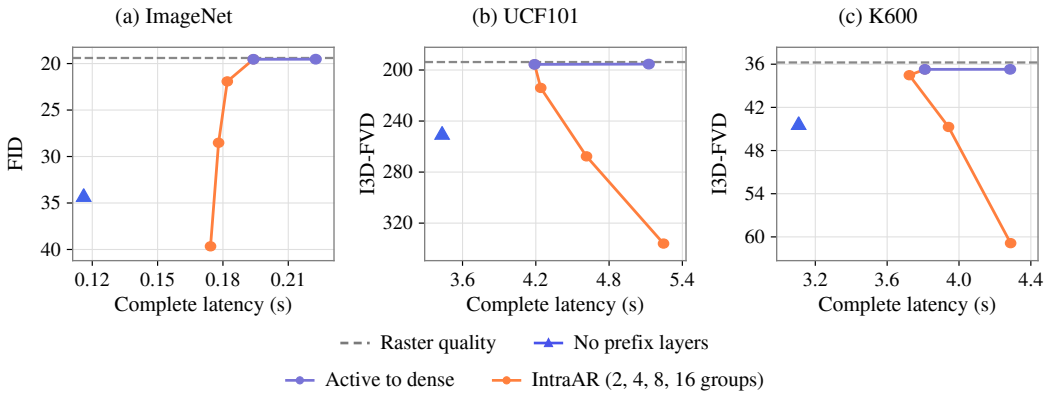


Figure 7: More sampling steps in one pass. Orange: $S = 2, 4, 8, 16$ at fixed prefix and total depth (ImageNet/LlamaGen-XL; UCF101 and K600 with OmniLM); lower FID or I3D-FVD is plotted higher. Purple: active versus dense execution. Triangles: no prefix. Dashed: Raster. Three-seed means; complete batch-1 latency.

5.4 When Should a Group Be Sampled?

An earlier draw outweighs its lost depth. The merge intervention predicts $G_1 \cup G_2$ together at ℓ_2 and retrains, which is Equation (8) with $A = G_1$, $B = G_2$ (Table 6; all 16 groups in Figure 12). Merging helps G_1, which gains one layer, by 0.09–0.25 nats/token, and hurts G_2, which loses G_1's draw, by 0.29–0.62, a 2.5–3.4 times larger effect in all three settings, so the net effect of (8) favors the early draw even though G_1 is predicted from only $P + 1$ layers. Even $G_{3:16}$ loses 0.12–0.23, as G_2 is then written from states that never saw G_1. K600 shows the smallest effects in all three terms, because its five observed frames already fix much of what G_1 would decide.

Table 6: Delaying the first draw: absolute group NLL (nats/token, three-seed means). Merging predicts original G_1 at the G_2 layer; group identities stay fixed and $G_{3:16}$ is token weighted. Bold: lower NLL per model/task.

Variant	$G_1 \downarrow$	$G_2 \downarrow$	$G_{3:16} \downarrow$
ImageNet / LlamaGen-XL (36L, $d = 1280$)			
IntraAR	6.410	**6.026**	**5.550**
Merge first two	**6.164**	6.646	5.705
UCF101 / OmniLM (24L, $d = 1536$)			
IntraAR	6.844	**6.546**	**5.648**
Merge first two	**6.610**	7.171	5.875
K600 / OmniLM (24L, $d = 1536$)			
IntraAR	4.624	**4.486**	**4.300**
Merge first two	**4.539**	4.774	4.418

More sampling steps per pass. Figure 7 varies $S \in \{2, 4, 8, 16\}$ with $\ell_s = P + s(L - P)/S$. Quality improves with S on all three tasks (UCF101 I3D-FVD from 336.02 to 195.59; ImageNet FID halves): finer groups lower the dependence floor (Theorem 2) by more than the earliest groups lose in depth, and most of the gain comes from the first splits: at $S = 8$, FID or FVD is already within 2–12% of its 16-group value (Figure 14). More groups also freeze rows earlier, while each commitment adds a head evaluation; on UCF101's 5,120 codes the first effect dominates and latency falls from 5.25 to 4.19 s, whereas on K600's 3,072 codes latency rises by 2% from 8 to 16 groups and on ImageNet's 256 codes $S = 2$ is about 10% faster.

6 Conclusion

Codes drawn in parallel from deterministic states cannot agree on choices that the condition leaves open; only a realized draw can couple them. IntraAR writes this draw back at the layer where it was made, so the remaining depth of the same pass predicts later groups conditioned on it. The experiments confirm the analysis: the drawn code accounts for most of the quality, an earlier draw outweighs its lost depth, and more steps per pass lower the dependence floor. Across five video and three image generators, one pass reaches raster-level quality at 11.2–41.5× lower latency, and the tokenizer decoder becomes the main remaining cost, so faster decoders translate directly into further end-to-end gains. The change concerns only where a drawn code re-enters the network, so it carries over to discrete-token generators whose layers can read a written code, as the five backbones used here show.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block Diffusion: Interpolating Between Autoregressive and Diffusion Language Models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2503.09573.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked Generative Image Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. URL https://arxiv.org/abs/2202.04200.

Liang Chen, Sinan Tan, Zefan Cai, Weichu Xie, Haozhe Zhao, Yichi Zhang, Junyang Lin, Jinze Bai, Tianyu Liu, and Baobao Chang. A Spark of Vision-Language Intelligence: 2-Dimensional Autoregressive Transformer for Efficient Finegrained Image Generation. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.01912.

Haotian Dong, Ye Li, Rongwei Lu, Chen Tang, Shu-Tao Xia, and Zhi Wang. VVS: Accelerating Speculative Decoding for Visual Autoregressive Generation via Partial Verification Skipping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. URL https://arxiv.org/abs/2511.13587.

Felix Draxler, Justus Will, Farrin Marouf Sofian, Theofanis Karaletsos, Sameer Singh, and Stephan Mandt. Parallel Token Prediction for Language Models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2512.21323.

Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long Video Generation with Time-Agnostic VQGAN and Time-Sensitive Transformer. In *European Conference on Computer Vision*, 2022. URL https://arxiv.org/abs/2204.03638.

Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the Content Bias in Fréchet Video Distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7277–7288, 2024. URL https://arxiv.org/abs/2404.12391.

Yefei He, Feng Chen, Yuanyu He, Shaoxuan He, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. ZipAR: Parallel Autoregressive Image Generation through Spatial Locality. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 22368–22378, 2025a. URL https://arxiv.org/abs/2412.04062.

Yefei He, Yuanyu He, Shaoxuan He, Feng Chen, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Neighboring Autoregressive Modeling for Efficient Visual Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19000–19010, 2025b. URL https://arxiv.org/abs/2503.10696.

Haopeng Li, Jinyue Yang, Guoqi Li, and Huan Wang. Autoregressive Image Generation with Randomized Parallel Decoding. In *International Conference on Learning Representations*, 2026a. URL https://arxiv.org/abs/2503.10568.

Ian Li, Zilei Shao, Benjie Wang, Rose Yu, Guy Van den Broeck, and Anji Liu. Breaking the Factorization Barrier in Diffusion Language Models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://arxiv.org/abs/2603.00045.

Boya Liao, Ying Li, Siyong Jian, and Huan Wang. Parallel Jacobi Decoding for Fast Autoregressive Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. URL https://arxiv.org/abs/2606.05703.

Xuan Luo, Weizhi Wang, and Xifeng Yan. Direct Multi-Token Decoding. In *Conference on Language Modeling*, 2026. URL https://arxiv.org/abs/2510.11958.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. Confident Adaptive Language Modeling. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2207.07061.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive Model Beats Diffusion: Llama for Scalable Image Generation, 2024. URL https://arxiv.org/abs/2406.06525. arXiv preprint arXiv:2406.06525.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual Autoregressive Modeling: Scalable Image Generation via Next-Scale Prediction. In *Advances in Neural Information Processing Systems*, volume 37, pp. 84839–84865, 2024. URL https://arxiv.org/abs/2404.02905.

Junke Wang, Yi Jiang, Zehuan Yuan, Binyue Peng, Zuxuan Wu, and Yu-Gang Jiang. OmniTokenizer: A Joint Image-Video Tokenizer for Visual Generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 28281–28295, 2024. URL https://arxiv.org/abs/2406.09399.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. VideoMAE V2: Scaling Video Masked Autoencoders With Dual Masking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14549–14560, 2023. URL https://arxiv.org/abs/2303.16727.

Yuqing Wang, Shuhuai Ren, Zhijie Lin, Yujin Han, Haoyuan Guo, Zhenheng Yang, Difan Zou, Jiashi Feng, and Xihui Liu. Parallelized Autoregressive Visual Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12955–12965, 2025. URL https://arxiv.org/abs/2412.15119.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: Video Generation using VQ-VAE and Transformers, 2021. URL https://arxiv.org/abs/2104.10157. arXiv preprint arXiv:2104.10157.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. MAGVIT: Masked Generative Video Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10459–10469, 2023. URL https://arxiv.org/abs/2212.05199.

Lijun Yu, José Lezama, Nitesh B. Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language Model Beats Diffusion - Tokenizer is key to visual generation. In *International Conference on Learning Representations*, 2024. URL https://iclr.cc/virtual/2024/poster/18119.

Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Randomized Autoregressive Visual Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18431–18441, 2025. URL https://arxiv.org/abs/2411.00776.

Zhuoyang Zhang, Luke J. Huang, Chengyue Wu, Shang Yang, Kelly Peng, Yao Lu, and Song Han. Locality-aware Parallel Decoding for Efficient Autoregressive Image Generation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2507.01957.

AI USE STATEMENT

Generative AI assisted with literature search, drafting and polishing manuscript sections, mathematical derivation checks, and table, figure, and layout preparation. The authors take full responsibility for all content, including results, proofs, references, and conclusions.

Organization of the appendix. The appendix follows the argument of the paper. Appendix A turns the central observation, that a drawn code can re-enter the network at the layer where it was drawn, into a complete procedure and explains each design decision against the alternative it replaces. Appendix B proves the statements of Section 4.4. Appendix C specifies what was trained and measured. Appendix D scores every group separately to show where in the grid the written draws help and where the lost depth hurts. Appendix E completes the comparisons of Section 5.2 with throughput, memory, training cost, conditional-prediction metrics, and samples, and Appendix F shows with three controls why coherence is judged on complete samples and whole-clip features.

A Implementation: From the Observation to One Forward Pass

The method of Section 4 rests on one observation: before a group is drawn, all hidden states are deterministic functions of the condition and the earlier draws, so the only new information a group can pass on is its drawn codes, and these codes can enter the network at the layer where they were drawn. Turning this observation into a generator raises one concrete question per step of the forward pass. What does a target row contain before anything is drawn? What may it attend to? How is a group read out at an intermediate layer, and what is written back? What happens to a written row at deeper layers? How are all S steps trained? Appendix A.1 gives the complete procedure, Appendix A.2 answers the questions in the order of the pass, and Appendix A.3 lists the numerical settings.

A.1 The Complete Procedure

Algorithm 1 states one IntraAR forward pass and its use for training and sampling, in the notation of Section 4. Training and sampling run the same pass and differ only in the code written at a commitment: the ground-truth code Z_i during training and a draw from π_i during sampling (lines 12–14). When C contains observed frames, Dec receives their codes together with $\widehat{Z}$. For a block ℓ, $\mathrm{N}_\ell^{\mathrm{a}}, \mathrm{N}_\ell^{\mathrm{f}}$ denote the backbone's native pre-attention and pre-MLP normalizations, $\rho(\cdot, \mathbf{p})$ its native positional operation at coordinates $\mathbf{p}$ ($\mathbf{p}_{[\boldsymbol{B};\boldsymbol{H}]}$ lists the coordinates of all rows in $[\boldsymbol{B};\boldsymbol{H}]$), and $\mathrm{Attn}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{softmax}(\boldsymbol{Q}\boldsymbol{K}^\top/\sqrt{d_k})\boldsymbol{V}$ with the backbone's heads and output projection. $\mathcal{A}$ and $\mathcal{R}$ are the ordered index sets of active and committed rows, $[\cdot\,;\cdot]$ stacks rows, and the variable $\boldsymbol{h}_i$ holds $\boldsymbol{h}_i^{(\ell)}$ after block ℓ.

A.2 Design Decisions and the Alternatives They Replace

Placeholder inputs at native coordinates (line 3). A raster generator feeds the code drawn at position $i-1$ into position i. In a single pass no code has been drawn when the pass starts, so every target row must start from a placeholder. IntraAR uses one learned mask embedding $\boldsymbol{e}_{[\mathrm{M}]}$ and places it at the coordinate that the pretrained backbone already assigns to that position: a 2D grid coordinate for images and a 3D spatiotemporal coordinate for video. The positional pathway therefore still tells each row where it sits in the clip, and continued training starts from a backbone whose knowledge of spatial and temporal layout is intact; $\boldsymbol{e}_{[\mathrm{M}]}$ is the only new input parameter.

Bidirectional attention, with causality carried by the write order (lines 5–8). A raster generator enforces causality with a triangular attention mask. IntraAR lets every active row attend to all target rows and to the condition, including rows of groups that are drawn later. Causality still holds: when group G_s is read out, the rows of $G_s, \ldots, G_S$ contain no code of their own, drawn or ground truth, and every state is a function of $(C, M_{<s})$ alone (Lemma 1). Causality is thus carried entirely by the order in which codes are written, and the attention pattern is free to serve requirement R1 of Section 4: an early group can look at the layout of the entire clip before it is drawn, which a causal mask would hide. Masked generators such as MaskGIT also attend bidirectionally (Chang et al., 2022), but they draw all codes of an iteration from the same states, so their draws within an iteration stay independent; IntraAR draws the groups one after another at increasing depth.

A shared prefix before the first draw. The first group is predicted from ℓ_1 layers only. If it were drawn after the first block, its distribution would come from nearly unprocessed mask embeddings, and every later group would condition on a poorly chosen draw. The P prefix blocks let all target rows exchange information with each other and with C before anything is drawn. Their cost is paid

Algorithm 1 IntraAR: S sampling steps within one forward pass

Require: condition C; groups $\mathcal{G} = (G_1, \dots, G_S)$ partitioning $[N]$; native coordinates $(\mathbf{p}_i)_{i\in[N]}$; prefix depth P and commitment layers $P < \ell_1 < \cdots < \ell_S = L$; blocks $\{\boldsymbol{W}_\ell^Q, \boldsymbol{W}_\ell^K, \boldsymbol{W}_\ell^V, \mathrm{MLP}_\ell, \mathrm{N}_\ell^{\mathrm{a}}, \mathrm{N}_\ell^{\mathrm{f}}\}_{\ell=1}^{L}$; head $(\boldsymbol{W}_{\mathrm{out}}, \mathrm{RMSNorm}_{\mathrm{out}})$; code embeddings $\{\boldsymbol{e}_v\}_{v\in[V]}$, mask embedding $\boldsymbol{e}_{[\mathrm{M}]}$, $\mathrm{RMSNorm}_{\mathrm{mem}}$; mode $\in$ {TRAIN, SAMPLE}; target grid Z (TRAIN only)

1: **function** ONEPASS(C, Z)
2: $\boldsymbol{C} \leftarrow \mathrm{Cond}(C)$ ▷ native conditioning; $\varnothing$ if unconditional
3: $\boldsymbol{h}_i \leftarrow \boldsymbol{e}_{[\mathrm{M}]}$ for all $i \in [N]$; $\mathcal{A} \leftarrow [N]$; $\mathcal{R} \leftarrow \varnothing$; $s \leftarrow 1$; $\mathcal{L} \leftarrow 0$
4: **for** $\ell = 1, \dots, L$ **do**
5: $\boldsymbol{B} \leftarrow [\boldsymbol{C}; (\boldsymbol{m}_j)_{j\in\mathcal{R}}]$, $\boldsymbol{H} \leftarrow (\boldsymbol{h}_i)_{i\in\mathcal{A}}$ ▷ read-only rows and active rows
6: $\boldsymbol{U} \leftarrow \mathrm{N}_\ell^{\mathrm{a}}([\boldsymbol{B}; \boldsymbol{H}])$, $\boldsymbol{U}_{\mathcal{A}} \leftarrow$ rows of $\boldsymbol{U}$ belonging to $\mathcal{A}$
7: $\boldsymbol{Q} \leftarrow \rho(\boldsymbol{U}_{\mathcal{A}}\boldsymbol{W}_\ell^Q, \mathbf{p}_{\mathcal{A}})$, $\boldsymbol{K} \leftarrow \rho(\boldsymbol{U}\boldsymbol{W}_\ell^K, \mathbf{p}_{[\boldsymbol{B};\boldsymbol{H}]})$, $\boldsymbol{V} \leftarrow \boldsymbol{U}\boldsymbol{W}_\ell^V$ ▷ active queries
8: $\boldsymbol{H} \leftarrow \boldsymbol{H} + \mathrm{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})$; $\boldsymbol{H} \leftarrow \boldsymbol{H} + \mathrm{MLP}_\ell(\mathrm{N}_\ell^{\mathrm{f}}(\boldsymbol{H}))$; $(\boldsymbol{h}_i)_{i\in\mathcal{A}} \leftarrow \boldsymbol{H}$ ▷ Eq. (4)
9: **if** $s \le S$ **and** $\ell = \ell_s$ **then** ▷ commitment s
10: $\pi_i \leftarrow \mathrm{softmax}\left(\boldsymbol{W}_{\mathrm{out}}\,\mathrm{RMSNorm}_{\mathrm{out}}(\boldsymbol{h}_i)\right)$ for all $i \in G_s$ ▷ $\pi_i = q_{\theta,i}(\cdot \mid C, M_{<s})$
11: **if** mode = TRAIN **then**
12: $\mathcal{L} \leftarrow \mathcal{L} - \sum_{i\in G_s} \log \pi_i[Z_i]$; $\widehat{Z}_i \leftarrow Z_i$ for $i \in G_s$ ▷ loss before targets are written
13: **else**
14: $\widehat{Z}_i \sim \mathrm{Cat}(\pi_i)$ independently for $i \in G_s$
15: $\boldsymbol{m}_i \leftarrow \mathrm{RMSNorm}_{\mathrm{mem}}\left(\boldsymbol{h}_i + \boldsymbol{e}_{\widehat{Z}_i}\right)$ for all $i \in G_s$ ▷ sample write, Eq. (3)
16: $\mathcal{A} \leftarrow \mathcal{A} \setminus G_s$; $\mathcal{R} \leftarrow \mathcal{R} \cup G_s$; $s \leftarrow s + 1$ ▷ rows of G_s are frozen
17: **return** $\widehat{Z} = (\widehat{Z}_i)_{i\in[N]}$ in native grid order, and $\mathcal{L}$

18: **procedure** TRAIN(dataset $\mathcal{D}$, pretrained raster weights θ_0, token budget T)
19: $\theta \leftarrow \theta_0$; initialize $\boldsymbol{e}_{[\mathrm{M}]}$ and set the scales of $\mathrm{RMSNorm}_{\mathrm{mem}}, \mathrm{RMSNorm}_{\mathrm{out}}$ to $\mathbf{1}$
20: **repeat**
21: draw a minibatch $\{(C^{(a)}, Z^{(a)})\}_{a=1}^{n} \sim \mathcal{D}$ and encode each $Z^{(a)}$ with the fixed tokenizer
22: $(\cdot\,, \mathcal{L}^{(a)}) \leftarrow$ ONEPASS$(C^{(a)}, Z^{(a)})$ with mode TRAIN ▷ all S groups in one pass, Eq. (6)
23: $\theta \leftarrow \mathrm{AdamW}\left(\theta,\ \nabla_\theta \frac{1}{nN} \sum_a \mathcal{L}^{(a)}\right)$ ▷ gradients pass through $\boldsymbol{h}_i, \boldsymbol{e}_{Z_i}$
24: **until** T input tokens have been processed
25: **return** θ

26: **procedure** SAMPLE(condition C, weights θ)
27: $(\widehat{Z}, \cdot) \leftarrow$ ONEPASS$(C, \varnothing)$ with mode SAMPLE ▷ $\widehat{Z} \sim q_\theta(\cdot \mid C)$, Eq. (5)
28: **return** $x \leftarrow \mathrm{Dec}(\widehat{Z})$ ▷ fixed tokenizer decoder

once and shared by all groups, whereas a separate context computation per group would reintroduce the repeated passes that IntraAR removes. Appendix D.3 shows that removing these blocks hurts the first groups most on the image models and all groups on video.

One vocabulary head for all commitments (line 10). Every commitment predicts codes of the same codebook, so IntraAR reads all of them through the backbone's own output matrix $\boldsymbol{W}_{\mathrm{out}}$, preceded by a learned $\mathrm{RMSNorm}_{\mathrm{out}}$. A separate head per commitment would add $(S-1)Vd$ parameters, about 3.1×10^8 for LlamaGen-XL with $S = 16$, $V = 16{,}384$, and $d = 1{,}280$, and each new head would start untrained. The added normalization is needed because hidden states at intermediate depths have a different scale than those that $\boldsymbol{W}_{\mathrm{out}}$ was trained on; early-exit language models read intermediate states through a shared head in the same way (Schuster et al., 2022).

Writing context plus code (line 15). A committed row could hold (i) the hidden state alone, (ii) the code embedding alone, or (iii) their sum. Option (i) is a deterministic function of $(C, M_{<s})$ and carries no information about the draw; it is exactly the no-sample-write control of Section 5.3. Option (ii) carries the draw but discards the context in which it was made, such as which object part the position belongs to, which deeper layers would then have to reconstruct from the code alone. IntraAR writes the sum, $\boldsymbol{m}_i = \mathrm{RMSNorm}_{\mathrm{mem}}(\boldsymbol{h}_i^{(\ell_s)} + \boldsymbol{e}_{\widehat{Z}_i})$. It uses the backbone's input embedding $\boldsymbol{e}_v$, the representation through which the pretrained layers already read codes, and the normalization puts the deep hidden state and the input embedding on a common scale. Proposition 2 shows that the normalized sum still identifies the drawn code. Figure 6 in Section 5.3 shows that the write changes the full predicted distributions of later positions, well beyond their top-1 code.

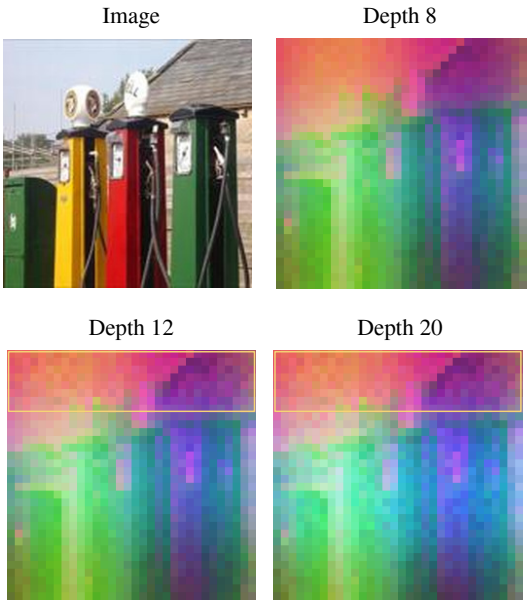


Figure 8: Committed rows keep their state across depth. Input and a common PCA projection of 32-channel states at depths 8, 12, and 20 on a 32×32 grid (Imagenette case `n03425413_16962`). The boxed rows are committed by depth 12 and are identical at depths 12 and 20, while the active rows below keep changing.

Freezing committed rows and reading them through per-layer keys and values (lines 5–8, 16). After the write, the row could keep being updated like any active row. IntraAR freezes it, which has two consequences. First, the record of the draw is exactly $\boldsymbol{m}_i$ at every deeper layer, so the injectivity of Proposition 2 holds at all of them; Figure 8 shows that the committed rows keep their state while the active rows continue to change. Second, committed rows leave the query and MLP computation, so the number of updated rows falls from N to $|G_S|$ over the pass. Each deeper layer still projects $\boldsymbol{m}_i$ with its own $\boldsymbol{W}_\ell^K, \boldsymbol{W}_\ell^V$, because each layer compares queries and keys in its own space. The dense-query control of Section 5.3 computes the omitted updates and then restores the frozen rows; it reproduces IntraAR's quality at higher latency and memory, so on long grids the omitted computation only adds cost (Appendix E.3 discusses the short-grid exception).

Training all S steps in one pass (lines 12, 19–25). Training runs the same pass with teacher forcing. At layer ℓ_s the loss of G_s is computed first and the ground-truth codes are written afterwards, so no target enters its own prediction, and one forward and backward pass trains all S conditionals, just as one causal pass trains all N conditionals of a raster model. Written codes are data, so gradients reach earlier layers through $\boldsymbol{h}_i^{(\ell_s)}$ and $\boldsymbol{e}_{Z_i}$ without an estimator for discrete sampling. At sampling time the drawn codes replace the ground truth, which is the usual gap between teacher forcing and sampling in autoregressive models. Training starts from the pretrained raster weights and adds $3d$ parameters: $\boldsymbol{e}_{[\mathrm{M}]}$ and the two normalization scales.

Schedule. Theorem 2 allows at most $K = L - P$ sampling steps when commitment layers are distinct. The default schedule $\ell_s = P + s$ uses every one of them: $S = 16 = L - P$ for LlamaGen-XL, OmniLM, RAR-L, and TATS-base, and $S = 8 = L - P$ for VideoGPT. Groups are equal contiguous chunks of the raster order in which the backbone was pretrained; Appendix D.6 compares a fixed random order, and Appendix D.5 uses fewer steps.

A.3 Native Components and Numerical Choices

Native block normalizations and positional operations are retained, and a row keeps its original coordinates after commitment or permutation. The two added RMSNorm scale vectors start at one and use $\tau = 10^{-6}$. Raster keeps its native output pathway, including RAR's final conditioning operations, while IntraAR uses the shared commitment head. Computation uses BF16, with normalization and log-softmax accumulated in FP32. The dense-query control computes queries and MLP updates for all rows and restores the committed rows after every block, which keeps the recurrence of Algorithm 1. Continued training updates all generator parameters with AdamW, learning rate 3×10^{-5}, $\beta = (0.9, 0.95)$, weight decay 0.05, gradient clipping at 1, 5% warmup, and cosine decay; the tokenizer is fixed. Each model/task pair receives at least 2×10^9 input visual tokens, rounded up to complete epochs, and evaluation uses the final checkpoint. Seeds 17, 29, and 43 pair initialization and data order across methods, and the dense-query evaluation reuses the IntraAR checkpoint. We use PyTorch 2.5.1 and CUDA 12.4 with eager execution and attention kernels that respect the visible edges of (4).

B THEORY: WHAT ONE PASS CAN AND CANNOT REPRESENT

This appendix proves the statements of Section 4.4. Appendix B.1 first works through the pedal-phase example of the introduction, because it already contains every effect that the general results quantify. Appendix B.2 shows that one pass defines a valid autoregressive distribution. Appendix B.3 proves the three-term decomposition, Appendix B.4 measures how much information a write passes on, Appendix B.5 proves the early-draw identity, and Appendix B.6 proves the depth bound. Appendix B.7 relates the likelihood to sample statistics, and Appendix B.8 counts the computation and storage of one pass against repeated passes.

B.1 A WORKED EXAMPLE: ONE SHARED BIT

Let C be constant and let $Z_1 = \cdots = Z_N = U$ for a fair bit U, the pedal phase of Section 3 shared by $N \geq 2$ positions. Each position alone has entropy $\log 2$, and so does the whole grid, because all positions carry the same bit.

One group drawn independently. For a product of per-position distributions, the expected NLL on the two admissible grids is $-\frac{1}{2}\sum_i \log\big(q_i(0)\,q_i(1)\big) \geq N \log 2$, with equality for uniform heads. The excess over the true entropy $\log 2$ is therefore at least $(N-1)\log 2$, which equals Γ for a single group. With uniform heads, which attain this bound, a sample agrees on one phase with probability 2^{1-N}. No choice of network changes this, because the N heads read deterministic states and cannot agree on a bit that has not been drawn.

A singleton drawn first. Commit $G_1 = \{1\}$, write its bit exactly, and put all other positions in G_2. Given the memory, every position of G_2 is deterministic, and Γ drops to zero. One early write of a single code thus removes the entire dependence of $N-1$ later positions; this is the mechanism that Theorem 1 and Equation (8) quantify in general.

What the write must preserve. If the write mapped both values of U to the same memory row, G_2 would be as uncertain as before, and the loss would reappear as Δ_{mem}. Proposition 2 shows that IntraAR's write keeps every drawn code recoverable. The example also fixes the scope of the analysis: a generator that feeds the shared random input U to every position reaches the correct joint law in one step, because its hidden states are themselves random. The results below cover decoders whose hidden states are deterministic given the condition and earlier draws, which includes IntraAR, raster decoding, and all parallel decoders compared in Section 5.

B.2 ONE PASS DEFINES A VALID AUTOREGRESSIVE DISTRIBUTION

We fix a condition variable C and a finite grid $Z \in [V]^N$ with data law p. Conditional probabilities and entropies are averaged over C; for continuous conditions, regular conditional laws are assumed to exist. The schedule $(P, \ell_1, \ldots, \ell_S)$, the groups $\mathcal{G} = (G_1, \ldots, G_S)$, and all position coordinates are fixed and known, so we suppress them from the conditioning. Each G_s is nonempty, and the groups are disjoint and exhaustive. Generation involves no random ordering, dropout, or other hidden stochastic state; the only fresh randomness consists of the independent categorical draws at the S commitments.

Let $M_{<s} = (M_1, \ldots, M_{s-1})$ denote the sequence of preceding writes, where $M_r = (\boldsymbol{m}_i)_{i \in G_r}$ is given by Equation (3). Under teacher forcing the writes are $M_s = \phi_s(C, M_{<s}, Z_{G_s})$, and the fitting term of Theorem 1 reads

$$\mathcal{F}_\theta = \sum_{s=1}^{S} \sum_{i \in G_s} \mathbb{E}_{C, M_{<s}}\, \mathrm{KL}\big(p(Z_i \mid C, M_{<s}) \,\big\|\, q_{\theta,i}(Z_i \mid C, M_{<s})\big). \tag{9}$$

All logarithms are natural, so $\mathcal{E}$, Γ, Δ_{mem}, and $\mathcal{F}_\theta$ are in nats per grid; dividing by N gives nats/token. Positive probabilities alone do not guarantee a finite expectation when C varies, so the statements additionally assume finite expected NLL, which keeps differences of losses well defined.

Lemma 1 (Causality and normalization)**.** *Immediately before commitment s, every active state is a deterministic function of $(C, M_{<s})$. The product of categorical factors in Equation* (5) *is a normalized conditional distribution.*

Proof. Before the first commitment, the mask embedding, the coordinates, and C determine every input and hence every state in the prefix. Suppose the claim holds before commitment s. The logits of G_s then depend only on $(C, M_{<s})$. After Z_{G_s} is drawn, the write creates M_s as a deterministic function of these variables and the new codes. Each following block takes active states determined by the preceding history together with the fixed new memory, so its outputs are determined by $(C, M_{\le s})$; iterating over the intervening blocks proves the induction step. For normalization, hold $(C, Z_{<S})$ fixed. The factors of the last group are normalized and independent, so their product sums to one over Z_{G_S}. Removing that group and repeating backwards yields $\sum_Z q_\theta(Z \mid C) = 1$; causality guarantees that an earlier factor never changes when a later group is summed out. Consequently,

$$\mathcal{E}(q_\theta) = \mathbb{E}_C \,\mathrm{KL}\big(p(Z \mid C) \,\|\, q_\theta(Z \mid C)\big) \ge 0. \tag{10}$$

The same argument covers teacher-forced likelihood evaluation, where the writes follow the grid whose probability is evaluated. □

B.3 Proof of Theorem 1: Dependence, Memory, and Fitting

Theorem 1 separates three reasons why IntraAR can assign lower likelihood to real grids than the data distribution does, so that each experiment in Appendix D can be read as a change in one term. The argument specializes standard cross-entropy and chain-rule identities to the causal memory. For a discrete variable X and a history $\mathcal{H}$, adding and subtracting $\log p(X \mid \mathcal{H})$ gives

$$\mathbb{E}\big[-\log q(X \mid \mathcal{H})\big] = H(X \mid \mathcal{H}) + \mathbb{E}_{\mathcal{H}}\,\mathrm{KL}\big(p(X \mid \mathcal{H}) \,\|\, q(X \mid \mathcal{H})\big). \tag{11}$$

KL is nonnegative because $\log u \le u - 1$ yields $-\sum_x p_x \log(q_x/p_x) \ge \sum_{x:\,p_x>0}(p_x - q_x) \ge 0$; conditional KL follows by averaging. Applying (11) to every categorical head gives

$$\mathbb{E}_p\big[-\log q_\theta(Z \mid C)\big] = \sum_{s=1}^{S} \sum_{i \in G_s} H(Z_i \mid C, M_{<s}) + \mathcal{F}_\theta. \tag{12}$$

Since $M_{<s}$ is a function of $(C, Z_{<s})$, the information that the memory omits satisfies

$$H(Z_i \mid C, M_{<s}) - H(Z_i \mid C, Z_{<s}) = I\big(Z_i; Z_{<s} \mid C, M_{<s}\big). \tag{13}$$

Add and subtract $\sum_s \sum_{i \in G_s} H(Z_i \mid C, Z_{<s})$ in (12) and subtract the chain rule $H(Z \mid C) = \sum_s H(Z_{G_s} \mid C, Z_{<s})$. The first difference is Δ_{mem} by (13), the remaining entropy difference is $\Gamma(\mathcal{G})$ by the definition (2), and $\mathcal{F}_\theta$ is unchanged. Total correlation is itself a KL divergence,

$$\mathrm{TC}(Z_G \mid \mathcal{H}) = \mathbb{E}_{\mathcal{H}}\,\mathrm{KL}\Big(p(Z_G \mid \mathcal{H}) \,\Big\|\, \textstyle\prod_{i \in G} p(Z_i \mid \mathcal{H})\Big), \tag{14}$$

so all three terms are nonnegative. □

B.4 How Much Information a Write Passes On

The no-sample-write control of Section 5.3 removes the drawn codes from memory. To predict what this removal costs, we rewrite the loss in terms of the information that each write gives to later targets. The same loss can be written relative to predicting every position independently from C. Expanding $H(Z_i \mid C, M_{<s}) = H(Z_i \mid C) - I(Z_i; M_{<s} \mid C)$, applying the chain rule of mutual information over the memory sequence, and reordering the finite sums gives

$$\mathcal{E}(q_\theta) = \mathrm{TC}(Z \mid C) - \sum_{r=1}^{S-1} \sum_{s=r+1}^{S} \sum_{i \in G_s} I\big(Z_i; M_r \mid C, M_{<r}\big) + \mathcal{F}_\theta. \tag{15}$$

Each write is thus credited for the uncertainty it removes from every later target. One shared bit that helps several later targets is counted once per target. Without a sample write, all hidden computation is deterministic given C, every credited term vanishes, and any number of layers leaves the product floor $\mathrm{TC}(Z \mid C)$; this is the regime of the no-sample-write control, and the per-group gap in Figure 9 is the empirical counterpart of the credited terms.

Proposition 1 (Information retained by one write)**.** *For the memory history $\mathcal{H}_r^{\mathrm{mem}} = (C, M_{<r})$ and a later target Z_i,*

$$I(Z_i; M_r \mid \mathcal{H}_r^{\mathrm{mem}}) = I(Z_i; Z_{G_r} \mid \mathcal{H}_r^{\mathrm{mem}}) - I(Z_i; Z_{G_r} \mid \mathcal{H}_r^{\mathrm{mem}}, M_r). \tag{16}$$

Proof. Expand $I(Z_i; Z_{G_r}, M_r \mid \mathcal{H}_r^{\mathrm{mem}})$ in two orders. Expanding first through Z_{G_r} leaves $I(Z_i; M_r \mid \mathcal{H}_r^{\mathrm{mem}}, Z_{G_r}) = 0$, since the write is deterministic. Expanding first through M_r gives the left-hand side plus the last term of (16). Equating both expansions proves the claim. □

A write can merge distinct draws. The next result gives sufficient conditions under which it preserves each drawn code exactly.

Proposition 2 (An injective write)**.** *Let the memory normalization be $f(\boldsymbol{x}) = \boldsymbol{\gamma} \odot \boldsymbol{x}/\sqrt{\|\boldsymbol{x}\|^2/d + \tau}$ with $\tau > 0$ and every component of $\boldsymbol{\gamma}$ nonzero. If distinct codes have distinct embeddings $\boldsymbol{e}_v$, the write* (3) *is injective in the drawn code given the incoming history. Hence the write history determines all earlier codes, and $\Delta_{\mathrm{mem}} = 0$ in exact arithmetic.*

Proof. For $\boldsymbol{y} = f(\boldsymbol{x})$ define $\boldsymbol{u} = \boldsymbol{y} \oslash \boldsymbol{\gamma}$. Then

$$\|\boldsymbol{u}\|^2 = \frac{\|\boldsymbol{x}\|^2}{\|\boldsymbol{x}\|^2/d + \tau} < d, \qquad \boldsymbol{x} = \sqrt{\frac{\tau}{1 - \|\boldsymbol{u}\|^2/d}}\, \boldsymbol{u}, \tag{17}$$

so f is invertible on its image. Before commitment s, the incoming history determines $\boldsymbol{h}_i^{(\ell_s)}$. Inverting $\boldsymbol{m}_i = f(\boldsymbol{h}_i^{(\ell_s)} + \boldsymbol{e}_{Z_i})$ and subtracting $\boldsymbol{h}_i^{(\ell_s)}$ recovers $\boldsymbol{e}_{Z_i}$ and hence Z_i. Repeating this recovery in commitment order shows that the earlier codes are deterministic given $(C, M_{<s})$, so every term of Δ_{mem} is zero. □

The result uses exact arithmetic, nonzero scales, and distinct embeddings; finite precision or coincident embeddings can merge draws. Preserving a distinction in memory leaves the fitting term separate: the finite-depth predictor still has to learn to use it.

B.5 When an Earlier Draw Pays Off

An earlier commitment gives later groups access to a draw but predicts that draw with fewer layers. Equation (8) states the exact balance; this subsection proves it and adds the terms that appear when whole grids of separately trained models are compared, as in Appendix D.4. Fix a common incoming history $\mathcal{H}$ and two disjoint adjacent groups A, B. Write $\mathcal{F}_A^{\mathrm{early}}, \mathcal{F}_A^{\mathrm{merge}}$ for the fitting KL terms against $p(Z_i \mid \mathcal{H})$, and $\mathcal{F}_B^{\mathrm{early}}, \mathcal{F}_B^{\mathrm{merge}}$ for those against $p(Z_j \mid \mathcal{H}, M_A)$ and $p(Z_j \mid \mathcal{H})$, respectively; the two reference conditionals for B differ by exactly the intervention. Cross-entropy decomposition gives

$$\mathcal{L}_{\mathrm{early}} = \textstyle\sum_{i \in A} H(Z_i \mid \mathcal{H}) + \sum_{j \in B} H(Z_j \mid \mathcal{H}, M_A) + \mathcal{F}_A^{\mathrm{early}} + \mathcal{F}_B^{\mathrm{early}}, \tag{18}$$

$$\mathcal{L}_{\mathrm{merge}} = \textstyle\sum_{i \in A} H(Z_i \mid \mathcal{H}) + \sum_{j \in B} H(Z_j \mid \mathcal{H}) + \mathcal{F}_A^{\mathrm{merge}} + \mathcal{F}_B^{\mathrm{merge}}. \tag{19}$$

Subtracting cancels the entropy of A, and each remaining entropy difference for B equals $-I(Z_j; M_A \mid \mathcal{H})$, which proves Equation (8) with $\mathcal{F}_{\mathrm{early}} = \mathcal{F}_A^{\mathrm{early}} + \mathcal{F}_B^{\mathrm{early}}$ and the analogous merged term. For whole-grid losses with identical preceding factors, the loss difference on all positions after $A \cup B$ adds a suffix term,

$$\mathcal{L}_{\mathrm{early}}^{\mathrm{grid}} - \mathcal{L}_{\mathrm{merge}}^{\mathrm{grid}} = \big(\mathcal{F}_{\mathrm{early}} - \mathcal{F}_{\mathrm{merge}}\big) - \sum_{j \in B} I(Z_j; M_A \mid \mathcal{H}) + R_{\mathrm{suffix}}, \tag{20}$$

where R_{suffix} is the summed loss difference, early minus merged, of all groups committed after $A \cup B$. In the first-boundary intervention of Section 5.4 there are no preceding groups and $\mathcal{H} = C$. The separately trained merge model has its own parameters, so the paired NLL difference combines the information, fitting, and suffix terms; reporting every original group (Table 6) shows each term's location.

B.6 Proof of Theorem 2: Depth Bounds the Number of Steps

A single pass has only $L - P$ layers after the prefix, and each sampling step occupies one of them. The following lemma shows that every additional step can only lower the dependence floor, which makes the number of available layers a budget.

Lemma 2 (Refining a group cannot raise the dependence floor)**.** *Split a group into nonempty sets A, B and commit A before B, holding the other groups fixed. With $\mathcal{H} = (C, Z_{\mathrm{prev}})$ containing all*

preceding codes, Γ *decreases by*

$$\mathrm{TC}(Z_{A\cup B} \mid \mathcal{H}) - \mathrm{TC}(Z_A \mid \mathcal{H}) - \mathrm{TC}(Z_B \mid \mathcal{H}, Z_A) = \sum_{j\in B} I(Z_j; Z_A \mid \mathcal{H}) \geq 0. \tag{21}$$

Proof. Write each total correlation as marginal entropies minus a joint entropy. The joint terms cancel because $H(Z_{A\cup B} \mid \mathcal{H}) = H(Z_A \mid \mathcal{H}) + H(Z_B \mid \mathcal{H}, Z_A)$, and the marginal entropies of A cancel, leaving $\sum_{j\in B}[H(Z_j \mid \mathcal{H}) - H(Z_j \mid \mathcal{H}, Z_A)]$. Later groups see the same preceding codes after both A and B are committed, and earlier groups are unchanged. □

Proof of Theorem 2. The grid is finite, so $\Gamma^*(s)$ is attained. If $s < N$, some group has at least two positions; refining it by Lemma 2 yields $s+1$ groups with no larger Γ, so $\Gamma^*(s+1) \leq \Gamma^*(s)$, and $\Gamma^*(N) = 0$. Distinct commitment layers in $\{P+1, \ldots, L\}$ and nonempty groups imply $S \leq L - P$ and $S \leq N$. Combining $S \leq K$, monotonicity, and Theorem 1 gives $\mathcal{E}(q_\theta) \geq \Gamma^*(S) + \Delta_{\mathrm{mem}} + \mathcal{F}_\theta \geq \Gamma^*(K) + \Delta_{\mathrm{mem}} + \mathcal{F}_\theta$. □

A finer identity separates the penalty of unused slots from the penalty of the chosen partition:

$$\mathcal{E}(q_\theta) = \Gamma^*(K) + \big[\Gamma^*(S) - \Gamma^*(K)\big] + \big[\Gamma(\mathcal{G}) - \Gamma^*(S)\big] + \Delta_{\mathrm{mem}} + \mathcal{F}_\theta, \tag{22}$$

where every term is nonnegative. For a target excess loss ε, define $D_{\mathrm{fix}}(\eta) = \min\{s \in [N] : \Gamma^*(s) \leq \eta\}$ with $\min \varnothing = \infty$. A fitted model with excess loss at most ε requires $D_{\mathrm{fix}}(\varepsilon - \Delta_{\mathrm{mem}} - \mathcal{F}_\theta) \leq S \leq K$. With a lossless memory and exact heads, the floor of the chosen partition is attained; a finite transformer adds the fitting error $\mathcal{F}_\theta$. The default schedule $\ell_s = P + s$ spends the whole budget, $S = K$, and Appendix D.5 measures what fewer steps cost.

B.7 From Grid Likelihood to Sample Statistics

The experiments report both likelihood and sample metrics. The following bound links them for any bounded statistic of the grid. For a fixed statistic $f(C, Z) \in [0, 1]$ and a common condition distribution,

$$\big|\mathbb{E}_p f - \mathbb{E}_{q_\theta} f\big| \leq \sqrt{\mathcal{E}(q_\theta)/2}. \tag{23}$$

Fix $C = c$ and let $A = \{z : p(z \mid c) > q_\theta(z \mid c)\}$, with total variation $\delta = p(A \mid c) - q_\theta(A \mid c)$. Merging outcomes into A and its complement cannot increase KL, and the binary KL between probabilities a, b satisfies $\mathrm{KL} \geq 2(a-b)^2$ because its second derivative in a is $1/[a(1-a)] \geq 4$. Bounded expectations differ by at most δ; averaging over C and applying Cauchy–Schwarz proves (23). A small excess NLL thus forces every bounded statistic of the samples, such as the rate at which two positions agree on a shared choice, to match the data. FVD depends on unbounded feature moments and is measured directly in Section 5.

B.8 Computation and Storage of One Pass

The latency gain of Section 5.2 comes from replacing N sequential raster passes, or S passes of group autoregression, by one. This subsection counts what changes, so that the measured costs in Appendix E.3 can be traced to their source.

For a group-replay implementation, the critical path contains SL block stages; IntraAR has L, with commitments inserted between stages. This counts sequential stages, independently of FLOPs or kernel launches. Both evaluate the output head on all N rows in total, and IntraAR initially carries all N rows as active queries. If the backbone weights occupy W_B bytes and at most $C_B \leq W_B$ bytes remain resident between complete passes, replay reads at least $W_B + (S-1)(W_B - C_B)$ bytes of weights, whereas one pass reads the backbone weights once at this level of accounting. Physical HBM traffic additionally depends on residency, tiling, and kernel fusion.

For a FLOP comparison, consider a reference that uses the same sample-write rule, replays all L blocks for the current group only, and caches projected committed rows. Let n_ℓ be IntraAR's active count at block ℓ, a_ℓ the per-row cost of query, output, and MLP computation, k_ℓ the per-row key/value projection cost, and $\Psi_\ell(\mathcal{V})$ the attention cost over the set $\mathcal{V}$ of visible edges. Replay computes N query rows per layer, N key/value projections for its prediction rows, and another $N - |G_S|$ for the post-write states it caches. IntraAR projects N key/value rows per layer and

updates only n_ℓ queries. Excluding heads and conditioning,

$$F_{\text{IntraAR}} - F_{\text{replay}} = -\sum_{\ell=1}^{L}\Big[(N-n_\ell)\,a_\ell + (N-|G_S|)\,k_\ell\Big] + \sum_{\ell=1}^{L}\Big[\Psi_\ell(\mathcal{V}_{\text{IntraAR}}) - \Psi_\ell(\mathcal{V}_{\text{replay}})\Big]. \tag{24}$$

The net difference combines saved row computation with the difference in visible attention edges. Storing a full projected key/value history across L layers takes up to $2LNd_{\text{kv}}b$ bytes for b bytes per scalar, whereas a streaming IntraAR implementation keeps Ndb bytes of active or committed states plus one layer's $2Nd_{\text{kv}}b$ projected values. These expressions count representation storage; Tables 10–12 report the measured full-process peaks, which also include weights, the decoder, and workspaces.

C Evaluation Protocol

The evaluation is built to answer one question: what changes when the same pretrained generator samples its grid in one pass instead of one code per pass? Every IntraAR model is therefore paired with a Raster model trained from the same weights, on the same tokens, in the same order, and both sample the same unguided distribution. The five video settings differ in how much of the clip the condition fixes, from no condition (Sky, Taichi) and a class label (UCF101) to one observed frame (BAIR) and five observed frames (K600); the more the condition leaves open, the more coherence depends on passing drawn codes between groups. The three ImageNet backbones differ in grid size (16×16 or 32×32), vocabulary (1,024 to 16,384), and depth (24 or 36 layers), which tests whether the result depends on one architecture.

C.1 Models, Tokens, and Conditions

Table 7 specifies the eight model/task pairs. The target-grid size is the product of its dimensions. K600 has 5,120 input codes: 2,048 condition codes and 3,072 targets. BAIR uses a separate first-frame conditioning network and predicts all 8,192 target codes. LlamaGen-XL has 36 layers and width 1,280; OmniLM has 24 layers and width 1,536; RAR-L has 24 layers and width 1,024 (Sun et al., 2024; Wang et al., 2024; Yu et al., 2025). BAIR uses the VideoGPT paper architecture with 16 layers, width 512, four attention heads, vocabulary 1,024, tokenizer embedding width 256, and a ResNet-34 first-frame condition. Sky and Taichi use TATS-base with 24 layers and width 1,024 (Yan et al., 2021; Ge et al., 2022).

Table 7: Model and task configurations. $L/P/S$ denotes backbone depth, prefix depth, and default commitments; d is backbone width. Target grids contain only generated positions. Added epochs round input exposure up to at least 2×10^9 visual tokens per seed. Train/test counts are source examples; FVD reference collections are defined in Appendix C.2.

Dataset / backbone	$L/P/S$	d	Target grid	V	Train / test	Epochs
ImageNet / LlamaGen-XL (Sun et al., 2024)	36/20/16	1,280	16×16	16,384	1,281,167 / 50,000	7
ImageNet / OmniLM (Wang et al., 2024)	24/8/16	1,536	32×32	8,192	1,281,167 / 50,000	2
ImageNet / RAR-L (Yu et al., 2025)	24/8/16	1,024	16×16	1,024	1,281,167 / 50,000	7
UCF101 / OmniLM (Wang et al., 2024)	24/8/16	1,536	$5\times32\times32$	8,192	9,537 / 3,783	41
K600 / OmniLM (Wang et al., 2024)	24/8/16	1,536	$3\times32\times32$	8,192	427,548 / 29,791	1
BAIR / VideoGPT (Yan et al., 2021)	16/8/8	512	$8\times32\times32$	1,024	43,264 / 256	6
Sky / TATS-base (Ge et al., 2022)	24/8/16	1,024	$4\times16\times16$	16,384	996 / 111	1961
Taichi / TATS-base (Ge et al., 2022)	24/8/16	1,024	$4\times16\times16$	16,384	3,049 / 285	641

Default groups are equal contiguous chunks of native raster order, so video groups advance through the latent frames in time order and later groups hold later frames. Layers are numbered from 1; default commitments use $\ell_s = P + s$, which spends the full budget $S = L - P$ of Theorem 2. The group-count sweep fixes P and L and uses $\ell_s = P + s(L-P)/S$ for $S \in \{2, 4, 8, 16\}$. LlamaGen-XL has $P = 20, L = 36$; the UCF101 and K600 OmniLM models have $P = 8, L = 24$. The fixed-random control orders target positions by a deterministic hash permutation and divides this order into equal groups. Each model/task pair uses one permutation shared across the three seeds. All rows retain their native spatial or spatiotemporal coordinates.

C.2 Image and Video Evaluation Sets

ImageNet uses 1,281,167 training images and 50,000 validation images. We generate 50 images per class, totaling 50,000 images per seed, and compute FID, spatial FID, Inception Score, precision, and recall with the ADM evaluation protocol. UCF101 uses split 1 with 9,537 training and 3,783 test videos. Its FVD reference contains 2,048 fixed test sources; generated class labels match these sources. K600 uses the CVDF split with 427,548 training and 29,791 validation sources, and 2,048 fixed conditions and references for FVD. A deterministic source-identifier hash ranking selects these reference subsets independently of method and seed. Each clip contains the central 17 consecutive frames. I3D evaluation resizes UCF101 to 128^2 and K600 to 64^2 while retaining all frames. K600 conditions encode only the first five true frames separately from the future targets. NLL uses all 50,000 ImageNet, 3,783 UCF101, or 29,791 K600 held-out examples.

Sky uses 996 training and 111 test sources. Its FVD reference includes all 225 test subdirectories, with central 16-frame clips, 2,048 generated clips, and ten evaluation repetitions per seed. NLL uses one clip per test source, selecting the first valid subdirectory in sorted order to weight sources equally. Taichi uses the original 3,049/285 train/test split. Its FVD protocol compares 2,048 training-reference clips with 2,048 generated clips over ten repetitions; NLL uses all 285 test sources. Clips take 16 frames at stride four from the central 64 source frames. FVD values use these task-specific reference populations.

I3D features are known to emphasize per-frame content over motion (Ge et al., 2024). Since IntraAR changes how choices are coordinated across positions and frames, we add a second video metric that is more sensitive to temporal structure: VideoMAE-v2 ViT-G/14 trained on Something-Something v2, following the content-debiased FVD implementation (Wang et al., 2023; Ge et al., 2024). Features are 1,408-dimensional pre-logit vectors. RGB values are divided by 255 and resized bilinearly to 224^2 without ImageNet normalization. Spatiotemporal patch features are averaged and passed through the final normalization. Feature extraction uses FP32 and collection statistics use FP64. I3D and VideoMAE use the same clips and collection membership; their Fréchet distances are reported separately.

BAIR measures conditional prediction, where one frame fixes the scene and leaves the arm motion open. It follows the first-frame-conditioned setting discussed by Yu et al. (2023), using the auxiliary camera view, 256 test conditions, 100 generated candidates per condition, and four repetitions within each seed. A candidate contains 16 predicted frames, including the first frame. PSNR and SSIM are first averaged over frames, separately maximized over the 100 candidates, then averaged over conditions and repetitions. These maxima may select different candidates. LPIPS uses VGG v0.1, native 64^2 inputs mapped to $[-1, 1]$, and FP32 computation. It is averaged over all frames and all candidates. FVD evaluates the complete generated collections against the corresponding real clips.

C.3 External Sampling Algorithms

The external decoders are run with the sampling recipes their authors use, including classifier-free guidance, so that each appears at its intended operating point. Guidance changes the distribution being sampled, which is why only the unguided Raster/IntraAR pairs define matched speedups and quality differences.

PAR (Wang et al., 2025) retains regional serial starting points, transition points, and three-dimensional grouping. NAR (He et al., 2025b) retains per-axis output heads and overlapping-logit mixing, adding $(D-1)dV$ output-matrix elements for grid rank D. LPD (Zhang et al., 2026) retains position queries and within-group visibility. ARPG (Li et al., 2026a) uses both passes and shared global K/V. ZipAR (He et al., 2025a) uses adaptive windows and rejection. PJD (Liao et al., 2026) uses two-dimensional drafts, row-causal attention, and stochastic acceptance. VVS (Dong et al., 2026) includes its drafter, caches, dynamic truncation, and skip constraints, with $\delta = 0.1$ and $s = 0.65$. On the 16×16 image grid, ZipAR's window of 16 covers the complete width and yields 256 rounds. PAR uses $N/g + g - 1$ rounds, while NAR uses the sum of target-grid axis lengths minus the number of axes plus one.

ImageNet guidance scales are 1.5 for PAR, 2 for NAR and LPD, 6.6/5.3 for ARPG-32/64, 2 for ZipAR, 3.5 for PJD, and 1.75 for VVS. PJD uses top-$k = 1000$; other image methods use no truncation, top-$p = 1$, and temperature 1. On class-conditioned video, PAR uses guidance 1.15 and

NAR 1.25. PAR uses top-$k = \min(8000, V)$ for video. Prefix-conditioned and unconditional tasks retain native conditions and disable guidance. Both LPD working points share one trained model, as do both ARPG points. ZipAR, PJD, and VVS reuse the continued Raster target; VVS additionally trains a class-conditioned drafter with the target vocabulary and width and the same added input-token exposure. These sampling settings determine the quality and cost of each external working point.

External decoders are compared by generation quality and system cost. Likelihoods are reported for Raster and the IntraAR variants, whose unguided decoders define exact normalized distributions.

C.4 Measurement and Statistical Units

Latency is measured end to end, including condition encoding and tokenizer decoding, because once the generator runs in one pass the decoder becomes a large share of the total (Appendix E.3); a generator-only timing would overstate the gain.

Each training seed is evaluated in a separate deployment process on one A100 80 GB SXM. At batch 1 and batch 4, 20 warmup requests precede 100 timed complete requests. Measured time includes condition encoding, generator execution, decoding, and CUDA synchronization; model loading and offline metric evaluation are outside the timed interval. If T_b is the total time for these 100 requests at batch b, batch-1 latency is $T_1/100$ seconds per output and batch-4 throughput is $400/T_4$ outputs per second. Peak memory is the full-process NVML peak, including the decoder and attention workspace. Component times are divided by the number of outputs. Training device-hours sum time across participating GPUs.

For each seed, group NLL is divided by the corresponding target-token count, and repeated generation metrics are averaged within the seed. Tables report the mean of the resulting three seed values. Paired differences and speedup ratios are computed within each seed before averaging. Training seeds are the independent evaluation units; groups and repeated candidates are nested within them.

D Where in the Grid the Gains Arise: Group-Level Diagnostics

Sample metrics such as FID and FVD summarize a whole grid, so they cannot show which groups gain from a written draw and which lose from being predicted with fewer layers. This appendix scores every group separately and reads each intervention through Theorem 1. Removing the code embedding targets the memory term Δ_{mem} (Appendix D.2); removing the prefix targets the fitting term $\mathcal{F}_\theta$ of early groups (Appendix D.3); merging the first two groups trades one term against the other, as in Equation (8) (Appendix D.4); and changing the number or order of groups changes the dependence term Γ together with the fitting term (Appendices D.5 and D.6). Appendix D.7 checks that later groups actually read the written rows, and Appendix D.8 explains why the diagnostics score groups by NLL.

D.1 Per-Group Loss

For training seed r, group G_s, and n held-out examples, the plotted teacher-forced loss is

$$u_{s,r} = -\frac{1}{n|G_s|} \sum_{a=1}^{n} \sum_{i \in G_s} \log q_{\theta_r,i}\left(z_i^{(a)} \mid C^{(a)}, M_{<s}^{\star,(a)}\right). \tag{25}$$

Teacher forcing gives every group the same ground-truth history in all compared models, so a difference in $u_{s,r}$ reflects how well a model predicts group s from that history, independently of errors made in earlier groups at sampling time. Curves average $u_{s,r}$ over seeds 17, 29, and 43, and whole-grid NLL weights each group by $|G_s|/N$. ImageNet uses $n = 50{,}000$ and 16 target codes per group for LlamaGen-XL and RAR-L; UCF101 uses $n = 3{,}783$ and 320 codes per group; K600 uses $n = 29{,}791$ and 192 codes per group, excluding its 2,048 condition codes. In video, raster groups follow time: the 16 UCF101 groups cover five latent frames and the 16 K600 groups cover three, so later groups are later frames.

D.2 The Written Draw Carries the Coupling

Figure 9 compares IntraAR with the no-sample-write control, which writes $e_{[\mathrm{M}]}$ in place of $e_{\widehat{Z}_i}$ and keeps groups, layers, and computation. By Proposition 1, a write passes on the information in the drawn codes minus what the write loses; the mask write loses all of it, so the control receives no information from earlier draws, and any gap between the curves is the information that only drawn codes supply.

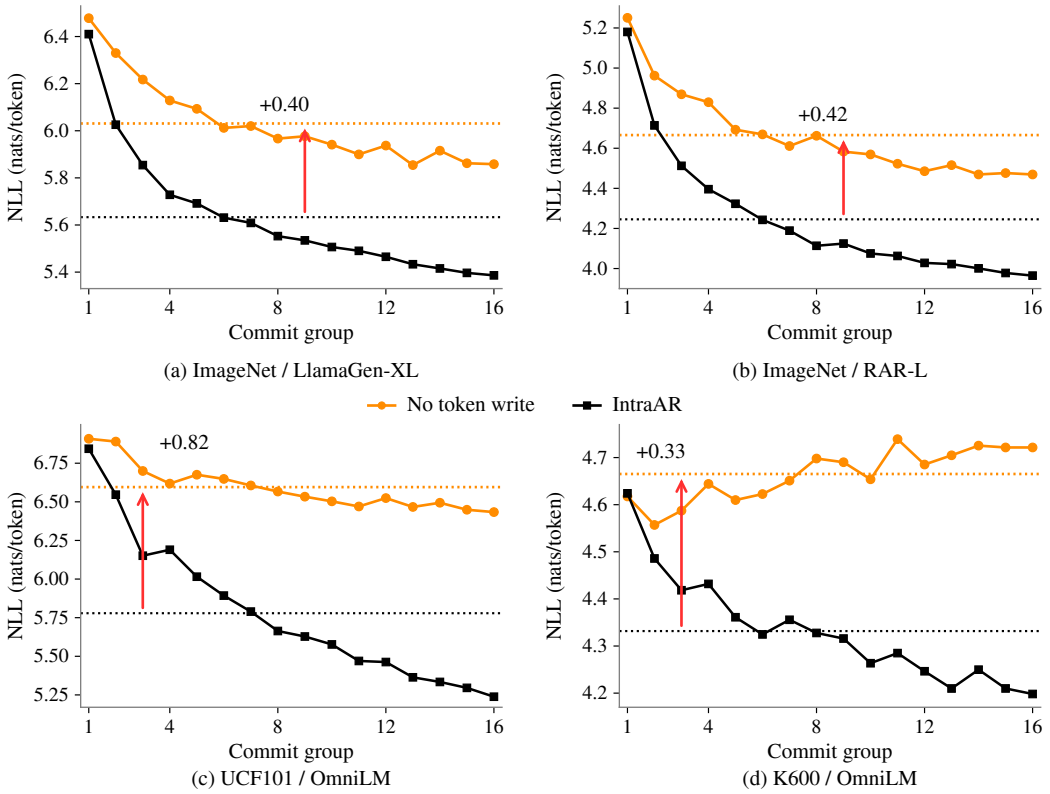


Figure 9: Sample writes reduce later-group prediction loss. Each panel compares separately trained IntraAR and a control that writes the fixed mask embedding in place of each sampled token. Both use 16 groups and the same depth schedule. Curves are three-seed mean teacher-forced NLL, using all 50,000 ImageNet, 3,783 UCF101, or 29,791 K600 held-out examples. Dotted lines show token-weighted whole-grid NLL; red arrows show the control-minus-IntraAR difference in nats/token. Group denominators are defined in Appendix D.1.

Three observations follow. First, the curves nearly coincide at group 1, which has no earlier draw to read in either model; the two models are equally good at the part of the task that does not involve memory. Second, the gap opens from group 2 on and produces whole-grid differences of 0.40 nats/token on LlamaGen-XL, 0.42 on RAR-L, 0.82 on UCF101, and 0.33 on K600. Third, the gap evolves differently in images and videos. On ImageNet it reaches about 0.4 nats/token by group 4 and then stays nearly constant: once the first rows fix the global layout, later rows depend mostly on their local neighborhood, which the deterministic states already describe. On UCF101 and K600 it keeps widening up to the last group, because later groups are later frames whose content depends on the motion sampled in earlier frames. On K600 the control's loss even rises with the group index as its targets move further from the five observed frames, while IntraAR's loss falls, because each drawn frame becomes new evidence for the next. In video, the value of passing drawn codes between groups therefore grows with the length of the generated history, and on UCF101 the control ends with a gap of about 1.2 nats/token at the last group.

D.3 The Prefix Serves the Earliest Groups

The no-prefix control deletes blocks $1, \ldots, P$ and keeps the group memberships and the later commitment layers, so every group loses P layers: 20 for LlamaGen-XL and 8 for the other models. If

the prefix only added depth, all groups would suffer about equally. Figure 10 shows that on the image models the penalty is concentrated on the first groups, as requirement R1 of Section 4 predicts, while on video it extends over all groups.

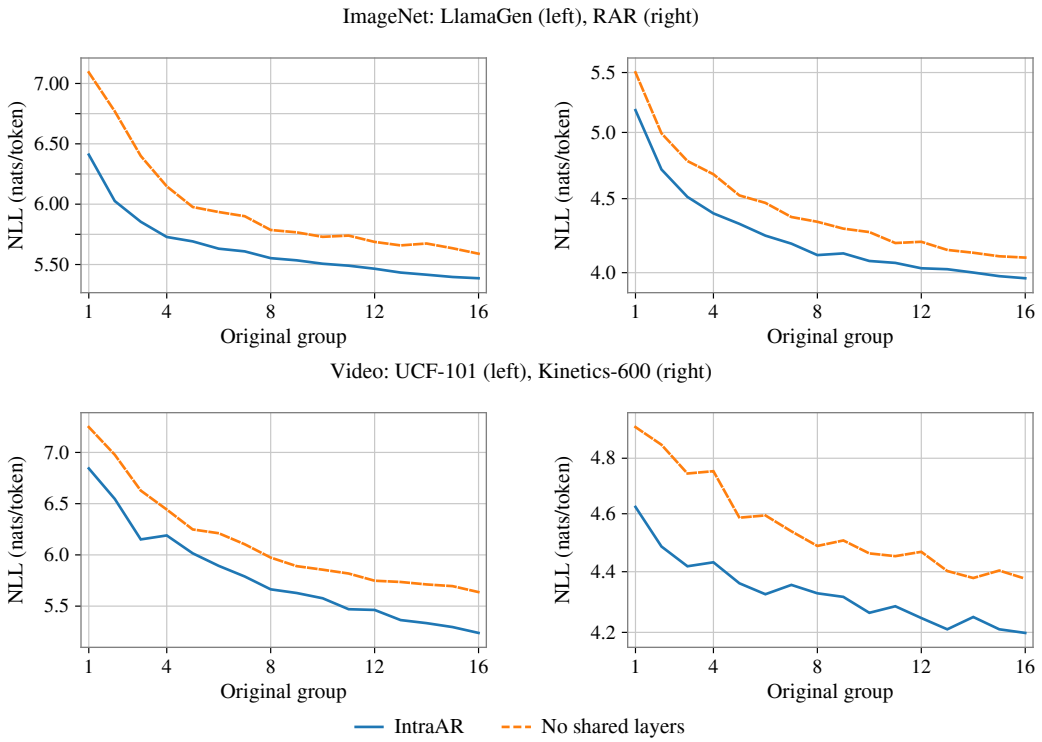


Figure 10: Removing prefix depth increases early prediction error. The legend's "No shared layers" removes the first 20 blocks of LlamaGen-XL or the first 8 blocks of RAR-L and OmniLM, retaining the original 16 group memberships and later commitment depths. Each variant is trained separately. NLL is teacher-forced nats per target token, averaged over three seeds and the full held-out sets: 50,000 ImageNet images, 3,783 UCF101 videos, and 29,791 K600 videos. RAR and K600 use logarithmic vertical axes.

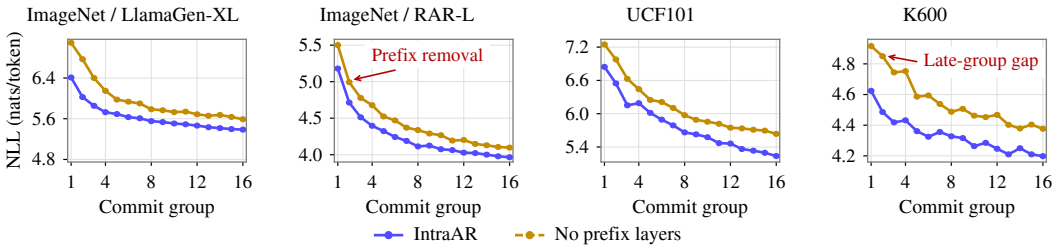


Figure 11: Prefix-removal penalties across four model/task pairs. The same three-seed losses as Figure 10 are arranged by model, with all 16 groups retained. The removed prefix is 20 layers for LlamaGen-XL and 8 for the other models. NLL uses natural logarithms and each group's target-token count. The first commitments show the largest loss increases; the K600 panel also retains the later-group gap under a known five-frame condition.

On LlamaGen-XL the gap is about 0.7 nats/token at group 1 and about 0.2 at group 16, although the last group also loses 20 of its 36 layers. A late group conditions on 15 written groups and needs little depth of its own; an early group has no draws to lean on and depends on the context that the prefix builds. Figure 11 places the four models side by side and shows the same ordering on RAR-L. On UCF101 the gap stays between about 0.3 and 0.4 nats/token across all groups, and on K600 a clear gap remains for later groups as well. In video, later groups must relate their frame to all earlier frames and to the condition, and the prefix is where the target rows first build this joint spatiotemporal context; this is consistent with removing the prefix raising UCF101 I3D-FVD from 195.59 to 249.26 (Table 5).

D.4 Delaying the First Draw

The merge intervention tests Equation (8) at the first boundary, where it is hardest to satisfy: G_1 is predicted after only $P+1$ layers. The merged model predicts $G_1 \cup G_2$ together at ℓ_2, which is layer 22 for LlamaGen-XL and layer 10 for OmniLM, keeps all later groups and layers, and is retrained. Every original group is scored separately.

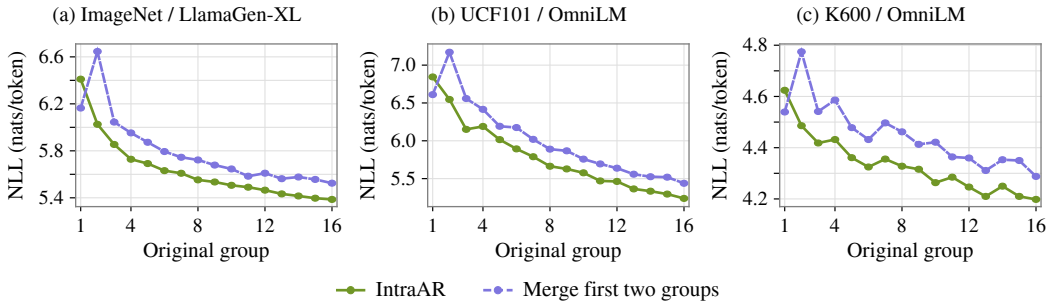


Figure 12: Delaying the first draw helps the first group and hurts all later groups. Teacher-forced NLL of all 16 original groups when G_1 and G_2 are merged at the second commitment layer; later groups and layers are unchanged. Means over three seeds on 50,000 ImageNet images, 3,783 UCF101 videos, and 29,791 K600 videos; each value is the summed group loss divided by its token count.

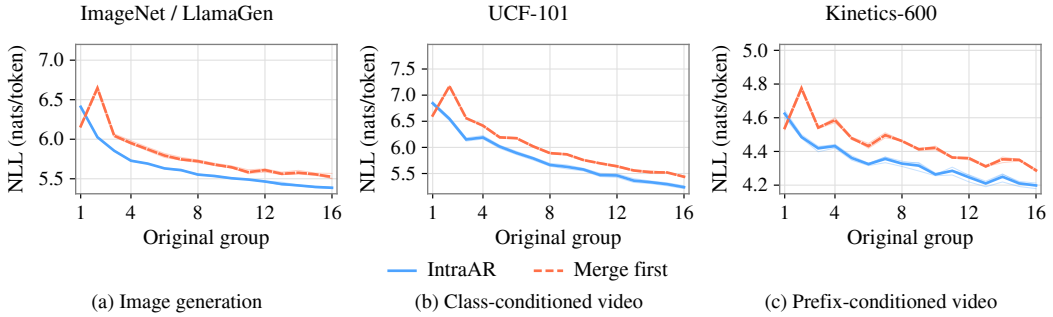


Figure 13: First-boundary merging changes the whole prediction profile. These are the same three-seed teacher-forced losses as Figure 12, shown with individual panel scales. The control merges only the first two original groups and retains all later memberships and depths. Each point uses all held-out targets in its original group: 50,000 ImageNet examples with 16 codes per group, 3,783 UCF101 examples with 320, or 29,791 K600 examples with 192. Units are nats/token. The group-2 increase exceeds the group-1 decrease in all three settings.

Paired merge-minus-IntraAR differences for $(G_1, G_2, G_{3:16})$ are $(-0.25, +0.62, +0.16)$ on LlamaGen-XL, $(-0.23, +0.62, +0.23)$ on UCF101, and $(-0.09, +0.29, +0.12)$ on K600 nats/token. The one extra layer helps G_1 only slightly, whereas losing access to G_1's draw costs G_2 between 2.5 and 3.4 times as much. The suffix loses as well, although it reads the same ground-truth codes of G_1 and G_2 in both models: in the merged model, the states from which G_2 is written never saw G_1, so every later group reads a less informed memory. The gain of the early draw therefore propagates beyond the next group. K600 shows the smallest effects in all three terms, because its five observed frames already fix much of what G_1 would otherwise decide.

D.5 How Many Sampling Steps per Pass

Theorem 2 states that more steps can only lower the dependence floor, while each step moves the first commitment closer to the prefix. Each $S \in \{2, 4, 8, 16\}$ uses equal raster-order groups, commitment layers $\ell_s = P + s(L-P)/S$, and its own trained model; $S = 16$ is the default. Figure 14 divides each metric by its value at $S = 16$.

Quality improves monotonically with S on all three tasks, with diminishing returns. From $S = 2$ to $S = 8$, FID or FVD falls from about 165–205% of the 16-group value to about 102–112%; the last doubling adds little. By Lemma 2, each refinement removes the information shared between the

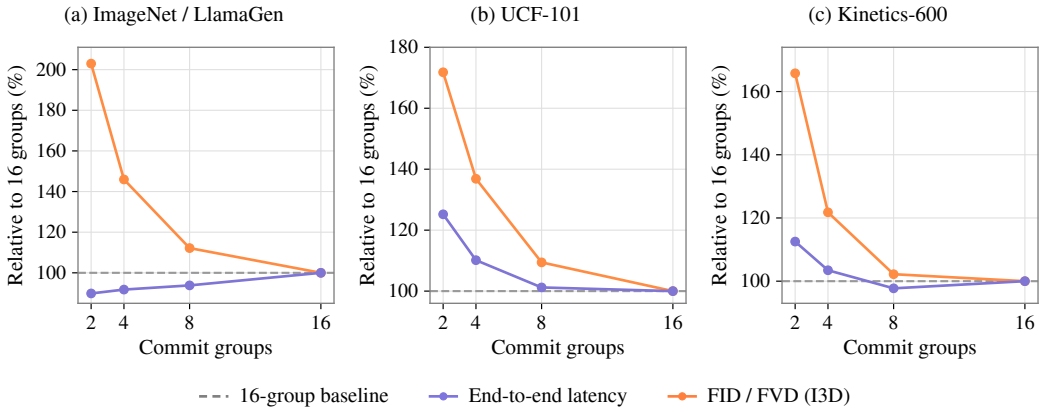


Figure 14: Relative quality and latency changes depend on the task. Each curve divides a three-seed metric mean by its 16-group value, with 100% marking the common reference. ImageNet/LlamaGen-XL uses FID on 50,000 generated images per seed; UCF101 and K600 use I3D-FVD on 2,048 generated clips with OmniLM. Latency covers the complete batch-1 pipeline on one A100. The four observed group counts use fixed prefix and total depth, as in Figure 7; lower values indicate lower distance or latency.

two halves of a group; the curves show that this amount is large for the first splits of large groups and small once groups are fine. The earlier first commitment (ℓ_1 moves from $P+8$ to $P+1$) never outweighs this gain, consistent with the merge result above. Latency follows the grid size. More groups freeze rows earlier and lower the average number of active rows, while each commitment adds a head evaluation and a sampling step. On UCF101, with 5,120 target codes, the saved updates dominate and $S = 2$ is about 25% slower than $S = 16$. On K600, with 3,072 codes, $S = 2$ is about 13% slower and $S = 8$ about 2% faster than $S = 16$. On ImageNet, with 256 codes, the head evaluations dominate and $S = 2$ is about 10% faster. On UCF101, more sampling steps therefore improve quality and speed at the same time.

D.6 Group Order

The default groups follow the raster order of pretraining. Figure 15 compares one fixed random permutation per model/task pair, with the same group sizes, commitment layers, and native coordinates, each trained for its own order.

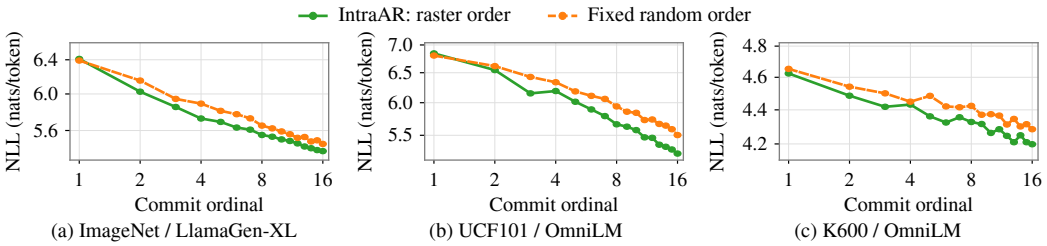


Figure 15: Prediction loss changes with group order. The control uses one fixed random target permutation shared across three seeds; IntraAR uses raster order. Both retain native coordinates, 16 equal-sized groups, and the same commitment depths. Points show mean teacher-forced NLL over all 50,000 ImageNet, 3,783 UCF101, or 29,791 K600 held-out examples. The horizontal axis is commitment ordinal: equal indices across curves contain different token members. Lines connect the 16 evaluated groups.

The two orders start from nearly the same loss at the first ordinal, and raster order becomes lower at almost every later ordinal. Two effects compete here. A random order spreads each group over the whole grid, so its members are far apart and weakly dependent, which lowers Γ. Raster order keeps the order in which the backbone was pretrained, which lowers the fitting term $\mathcal{F}_\theta$ of continued training. The measured profiles show that the second effect dominates for these models, so raster order is the default.

D.7 Later Groups Read the Written Rows

The previous subsections show that written draws lower the loss. Figure 16 checks the path by which this happens: how much attention an active row puts on memory rows just before it is committed. The value at position $i \in G_s$ is $a_i = \sum_{j \in G_{<s}} A_{ij}^{(\ell_s)}$, the attention mass on earlier memory at its commitment layer, on a 16×16 diagnostic grid with $L = 24$ and $P = 8$.

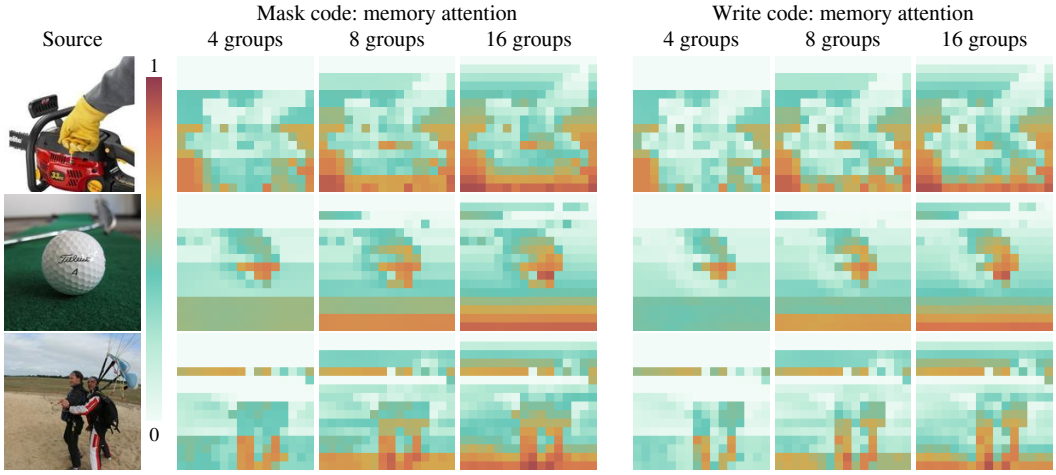


Figure 16: Memory-read mass at each position's commitment. Rows are three Imagenette cases (chainsaw use, golf ball and club, and parachuting). The two column blocks compare fixed-mask and code writes; columns within each block use $S = 4, 8, 16$ with $L = 24$, $P = 8$, and commitment depth $\ell_s = 8 + 16s/S$. On the 16×16 diagnostic grid, the value at position $i \in G_s$ is the sum of its last pre-commitment attention probabilities over earlier memory, $a_i = \sum_{j \in G_{<s}} A_{ij}^{(\ell_s)}$. All 18 maps share [0,1] under teacher forcing. The first group has no earlier memory and hence zero read mass. Warmer colors indicate more attention to memory.

Read mass is zero for the first group, grows toward the lower rows as more groups are committed, and grows with S, since a finer schedule puts more committed rows in memory at each commitment. Its uniform reference is $|G_{<s}|/256$. The mask-write and code-write models allocate attention in nearly the same pattern, so the control does read its memory rows; its higher loss in Figure 9 comes from what those rows contain. The difference between the two models lies in the content of memory, which is the drawn code.

D.8 Why the Diagnostics Use NLL

Entropy or top-1 maps are a tempting alternative to per-group NLL because they need no ground truth. Figure 17 shows why they cannot replace it. A small grouped RGB-code probe (3/4/3-bit codes on a 16×16 grid, 16 row groups, 64 tanh units, trained with Adam on 56 Imagenette images) is evaluated with identical teacher-forced history at three checkpoints.

For the church case, entropy falls over training while held-out NLL rises from 4.5575 to 4.7902 between steps 256 and 1024: the probe becomes more confident and less accurate. Confidence therefore measures neither the fitting term $\mathcal{F}_\theta$ nor the information in memory, and Figure 6 gives the converse case, where similar top-1 maps hide large differences between full distributions. Per-group NLL scores the full distribution against the data and is the quantity that Theorem 1 decomposes.

E Complete Results Behind Section 5.2: Quality, Time, and Memory

Section 5.2 reports quality and batch-1 latency. A decoder that runs in one pass also changes throughput at larger batches, peak memory, training cost, and the share of time spent outside the generator. This appendix reports all of them under the definitions of Appendix C.4 and explains each difference by the computation that Appendix B.8 counts.

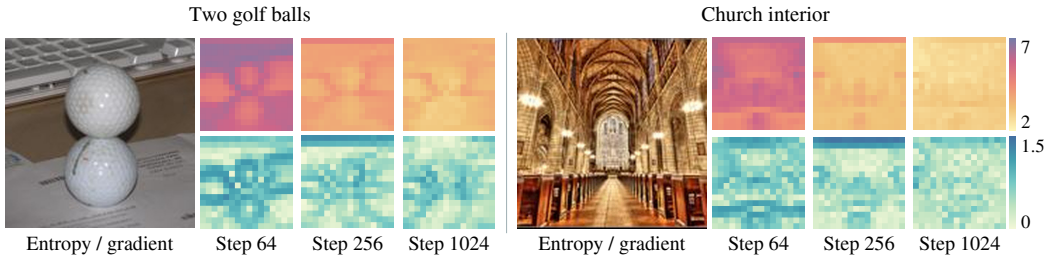


Figure 17: Spatial uncertainty across fixed training checkpoints. Each held-out Imagenette case is followed by entropy (top) and spatial entropy-gradient magnitude (bottom) at steps 64, 256, and 1024. The grouped RGB-code AR probe uses a 16×16 grid, a 1024-color vocabulary, 16 row groups, and identical teacher-forced history at all checkpoints. Entropy is $H_i = -\sum_v p_i(v) \log p_i(v)$ in nats; the lower row is $\|\nabla H\|_2$ in nats per grid step. All entropy maps share [2,7]; gradients share [0,1.5] with the same asinh display transform. A top-row band marks the transition from empty to available history.

E.1 Video Throughput and Memory

Table 8 adds batch-4 throughput and batch-1/4 peak memory to the video comparison of Table 1. IntraAR has the highest batch-4 rate on UCF101, K600, Sky, and Taichi. The exception is BAIR, where NAR reaches 1.03 videos/s against IntraAR's 0.85 although its batch-1 latency is higher (3.40 versus 1.94 s). BAIR is the only task where the batch-1 and batch-4 rankings differ, and batch-1 latency, the quantity compared in Section 5.2, still favors IntraAR by a factor of 1.8. Across PAR settings, larger groups raise the rate and worsen I3D-FVD on every task, the trade-off that motivates the paper. IntraAR's peak memory stays within 1.1 GiB of Raster's at batch 4 on all five tasks.

Table 8: Video throughput and full-process memory across all five datasets. Rate is batch-4 videos/s; B1/B4 are batch-1/batch-4 NVML peaks in GiB. Means over three seeds. Method, backbone, sampling, and task conditions match Table 1, including PAR (Wang et al., 2025) and NAR (He et al., 2025b). Throughput is measured directly at batch 4.

Method	UCF101			K600			BAIR			Sky			Taichi		
	OmniLM			OmniLM			VideoGPT			TATS-base			TATS-base		
	Rate↑	B1↓	B4↓	Rate↑	B1↓	B4↓	Rate↑	B1↓	B4↓	Rate↑	B1↓	B4↓	Rate↑	B1↓	B4↓
Raster	0.02	**13.80**	**43.11**	0.03	14.34	**43.98**	0.05	**3.18**	**8.81**	0.28	4.69	**9.02**	0.25	4.78	**9.32**
PAR-4 (Wang et al., 2025)	0.06	14.15	46.27	0.09	14.43	45.81	0.16	3.36	10.32	0.98	4.77	9.34	0.86	4.78	9.76
PAR-16 (Wang et al., 2025)	0.21	14.87	49.80	0.28	14.78	48.61	0.53	3.57	11.20	3.17	4.94	9.89	2.72	5.00	10.19
NAR (He et al., 2025b)	0.33	14.38	45.48	0.30	14.43	45.00	**1.03**	3.37	10.07	4.01	4.81	9.42	3.65	4.84	9.78
IntraAR	**0.35**	13.93	43.55	**0.38**	**14.27**	44.22	0.85	3.28	9.83	**7.11**	**4.67**	9.19	**5.71**	**4.70**	9.62

E.2 Image Likelihood, Additional Metrics, and Samples

Table 9 adds teacher-forced NLL, sFID, and Inception Score to Table 2. NLL is within 0.04 nats/token of Raster on all three backbones and is lower for RAR-L (4.25 versus 4.29), while FID on RAR-L is higher (9.47 versus 9.24). The two measure different things. Teacher-forced NLL scores each group given ground-truth history, whereas a sample conditions every group on earlier draws that IntraAR made with fewer layers than a full pass; an imperfect early draw is inherited by all later groups. The same inheritance makes the no-prefix control lose more FID or FVD per nat of NLL than the no-write control on three of four settings (Section 5.3), and it explains why equal likelihood can come with a slightly higher FID.

Table 9: Complementary ImageNet likelihood and image-quality metrics. Three-seed means for LlamaGen-XL (Sun et al., 2024), OmniLM (Wang et al., 2024), and RAR-L (Yu et al., 2025). NLL is nats/token and IS uses Inception features. Sampling is matched and unguided; FID, precision/recall, latency, and throughput appear in Table 2.

Method	LlamaGen-XL			OmniLM			RAR-L		
	36L, $d = 1280$			24L, $d = 1536$			24L, $d = 1024$		
	NLL↓	sFID↓	IS↑	NLL↓	sFID↓	IS↑	NLL↓	sFID↓	IS↑
Raster	**5.60**	8.87	**66.71**	**5.46**	**6.85**	**131.48**	4.29	**5.71**	**120.74**
IntraAR	5.63	**8.78**	66.59	**5.46**	7.06	129.27	**4.25**	5.73	119.73

Figure 18 shows unguided IntraAR samples for 24 classes, two per class; the two samples of a class differ in pose, viewpoint, and background while each keeps a coherent object layout.



Figure 18: IntraAR samples for 24 ImageNet classes (LlamaGen-XL, unguided). Each class contributes two independent samples; adjacent rows share a class.

Coordination is also needed within a single image. The supports, canopies, and seats of a carousel must repeat consistently across the image, the spatial counterpart of the shared motion phase in video. Figure 19 shows 16 independent Raster and IntraAR samples of this class; both keep the repeated parts consistent within each sample while varying viewpoint and composition across samples.



Raster · 16 samples IntraAR · 16 samples

Figure 19: Repeated structure within one image. Raster (left) and IntraAR (right) each contribute 16 independent samples of the ImageNet carousel class. Repeated supports, canopies, and seats must agree with each other across the image, a spatial counterpart of the shared choices in video.

E.3 Where Time and Memory Go

Tables 10–12 split each deployment into training GPU-hours, batch-1 generator and decoder time, and full-process peak memory. Three patterns follow from the cost accounting of Appendix B.8.

Generator time collapses and the decoder remains. On UCF101, generator time falls from 171.97 to 2.12 s per clip, while tokenizer decoding stays at about 2.1 s. Decoding thus takes about half of IntraAR's complete UCF101 latency and more than half on K600 (2.10 of 3.81 s); on ImageNet it is about a tenth or less. Further speedups of the generator alone therefore yield diminishing end-to-end gains on video.

Table 10: Matched image training and component costs. Means over three seeds. Train: GPU-hours per deployment recipe; Gen./Dec.: batch-1 seconds/output; B1/B4: NVML GiB. Conditioning and synchronization also contribute to total latency. Model definitions and references: Table 7.

Method	Train↓	Gen.↓	Dec.↓	B1↓	B4↓
ImageNet / LlamaGen-XL (36L, $d = 1280$)					
Raster	27.52	2.70	**0.02**	**4.30**	5.70
IntraAR	**24.77**	**0.17**	**0.02**	4.37	**5.63**
ImageNet / OmniLM (24L, $d = 1536$)					
Raster	**36.52**	17.70	**0.05**	5.41	**7.99**
IntraAR	38.93	**0.54**	**0.05**	**5.23**	9.45
ImageNet / RAR-L (24L, $d = 1024$)					
Raster	18.84	1.66	**0.01**	**3.23**	4.28
IntraAR	**18.15**	**0.14**	**0.01**	3.27	**4.04**

Peak memory depends on what each decoder stores. Raster keeps a key–value cache for all L layers that grows to the full grid, whereas IntraAR keeps no cache across passes but carries all target rows through the pass at once (Appendix B.8). IntraAR's batch-4 peak is lower on the 256-code grids (5.63 versus 5.70 GiB on LlamaGen-XL) and higher on ImageNet/OmniLM's 1,024-code grid (9.45 versus 7.99 GiB), where the full set of active rows and their attention workspace dominate. On video, weights, the tokenizer decoder, and workspaces dominate both peaks, which differ by at most 1.02 GiB.

Training cost depends on grid length. Matched token exposure gives IntraAR a lower training cost on the two 256-code ImageNet grids (24.77 versus 27.52 GPU-hours on LlamaGen-XL, 18.15 versus 18.84 on RAR-L) and a higher one on every grid of 1,024 codes or more, most on UCF101 (48.90 versus 39.47). Frozen rows save query and MLP work, whereas active rows attend over the entire grid instead of a causal prefix, which adds attention work growing with the square of the grid length. The balance also depends on model width; on every grid of 1,024 codes or more in this study the added attention work outweighs the savings, by 7.6% on BAIR's narrow VideoGPT and by 24% on UCF101.

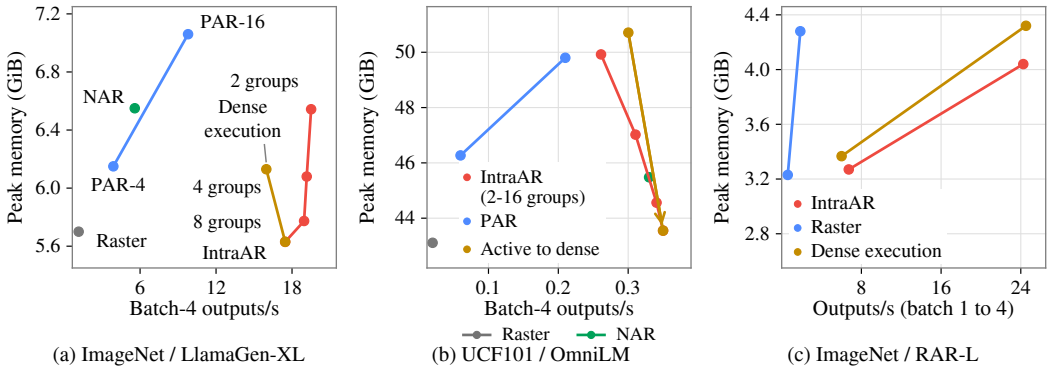
Table 11: Matched video training and component costs. Means over three seeds. Train: GPU-hours per deployment recipe; Gen./Dec.: batch-1 seconds/output; B1/B4: NVML GiB. Conditioning and synchronization also contribute to total latency. Model definitions and references: Table 7.

Method	Train↓	Gen.↓	Dec.↓	B1↓	B4↓
UCF101 / OmniLM (24L, $d = 1536$)					
Raster	**39.47**	171.97	2.07	**13.80**	**43.11**
IntraAR	48.90	**2.12**	**2.06**	13.93	43.55
K600 / OmniLM (24L, $d = 1536$)					
Raster	**43.26**	108.80	**2.05**	14.34	**43.98**
IntraAR	47.42	**1.39**	2.10	**14.27**	44.22
BAIR / VideoGPT (16L, $d = 512$)					
Raster	**7.36**	71.72	**0.18**	**3.18**	**8.81**
IntraAR	7.92	**1.74**	**0.18**	3.28	9.83
Sky / TATS-base (24L, $d = 1024$)					
Raster	**14.39**	11.82	0.12	4.69	**9.02**
IntraAR	15.00	**0.30**	**0.11**	**4.67**	9.19
Taichi / TATS-base (24L, $d = 1024$)					
Raster	**14.38**	13.08	**0.20**	4.78	**9.32**
IntraAR	15.07	**0.34**	**0.20**	**4.70**	9.62

Table 12: External image deployment costs. Means over three seeds. Train: GPU-hours per deployment recipe; Gen.: batch-1 seconds/output; B1/B4: NVML GiB. Conditioning and synchronization also contribute to total latency. External methods and citations follow Table 3. VVS includes target and drafter; operating points that share a checkpoint list that checkpoint's training cost. Model definitions and references: Table 7.

Method	Train↓	Gen.↓	B1↓	B4↓
ImageNet / LlamaGen-XL (36L, $d = 1280$)				
Raster	27.52	2.70	**4.30**	5.70
PAR-4 (Wang et al., 2025)	29.19	0.92	4.45	6.15
PAR-16 (Wang et al., 2025)	29.89	0.37	4.77	7.06
NAR (He et al., 2025b)	31.48	0.68	4.72	6.55
LPD-20 (Zhang et al., 2026)	48.05	0.43	4.67	6.11
LPD-32 (Zhang et al., 2026)	48.05	0.69	4.63	6.05
ARPG-32 (Li et al., 2026a)	43.81	0.51	4.87	6.47
ARPG-64 (Li et al., 2026a)	43.81	0.93	4.81	6.42
ZipAR-16 (He et al., 2025a)	27.52	5.32	4.45	6.46
PJD-6 (Liao et al., 2026)	27.52	1.25	4.92	7.21
VVS (Dong et al., 2026)	31.35	3.48	4.77	6.84
IntraAR	**24.77**	**0.17**	4.37	**5.63**

Table 12 gives the same breakdown for the external ImageNet recipes of Table 3. LPD and ARPG each reuse one trained model for their two operating points, and ZipAR, PJD, and VVS reuse the continued Raster model; VVS adds 3.82 GPU-hours for its drafter, and its timing includes drafting, caches, verification, and skip constraints. IntraAR has the lowest generator time (0.17 s), the lowest training cost, and the lowest batch-4 memory of all recipes.



(a) ImageNet / LlamaGen-XL (b) UCF101 / OmniLM (c) ImageNet / RAR-L

Figure 20: Memory and throughput depend on the execution setting. ImageNet/LlamaGen-XL and UCF101/OmniLM use batch-4 working points for IntraAR with 2, 4, 8, or 16 groups, PAR, Raster, NAR, and dense queries. RAR-L connects batch 1 to batch 4 for IntraAR, Raster, and dense queries. Each point averages three processes, one per training seed, on one A100 80 GB; memory is the full-process NVML peak in GiB and rates include the complete generation pipeline. Dense queries reuse the corresponding IntraAR model.

Figure 20 relates peak memory to throughput for the group-count sweep and the execution controls. On UCF101, more groups move IntraAR toward both higher rate and lower memory, since rows are frozen earlier. On ImageNet/LlamaGen-XL, fewer groups are slightly faster but use more memory, because more rows stay active in the later layers. On RAR-L, dense queries are slightly faster than IntraAR at batch 4 while using more memory: on a 256-code grid the query and MLP work saved by freezing is small, so the cost of indexing the active rows is not recovered. The benefit of active-row execution is therefore largest on long grids, which are the video grids this paper targets.

E.4 Conditional Video Prediction: K600 and BAIR

Figure 3 showed class-conditioned clips, where the label fixes the action and leaves pose, motion phase, and scene open. The two prefix-conditioned tasks fix more. On K600, five observed frames fix the scene and the hands, and Figure 21 shows that the generated frames continue the motion of the cards and the knife from where the prefix leaves off.

On BAIR, one frame fixes the scene and leaves the arm motion open, and the candidates differ exactly there (Figures 4 and 5 in Section 5.2). Table 13 and Figure 22 report three estimands (Appendix C.2). Best-of-100 PSNR and SSIM ask whether some candidate comes close to the recorded future, which rewards coverage of the plausible futures. Mean LPIPS averages the perceptual error of all candidates, which rewards typical candidates. FVD compares whole collections. IntraAR is within 0.09 dB of Raster in best PSNR, has the lowest mean LPIPS (0.1620), and the lowest VideoMAE-FVD (91.56), at 1.94 s instead of 71.92 s per clip. PAR-16 is worst on every metric: large independent groups produce candidates whose parts disagree, which both lowers their best match and raises their average error.

Table 13: BAIR conditional evaluation / VideoGPT (Yan et al., 2021). Means over three seeds. I3D/MAE are FVD. PSNR (dB) and SSIM separately select the best of 100 candidates after averaging 16 frames; LPIPS averages all candidates. The condition frame is never copied into predictions. PAR/NAR use the recipes and citations in Table 1.

Method	I3D↓	MAE↓	PSNR↑	SSIM↑	LPIPS↓
Raster	**111.16**	92.57	**18.54**	0.7560	0.1622
PAR-4 (Wang et al., 2025)	117.18	101.42	18.31	0.7525	0.1679
PAR-16 (Wang et al., 2025)	130.75	120.29	17.94	0.7420	0.1811
NAR (He et al., 2025b)	112.03	98.06	18.50	**0.7577**	0.1648
IntraAR	112.87	**91.56**	18.45	0.7536	**0.1620**

E.5 Fidelity and Coverage of the External Image Recipes

The external ImageNet recipes of Table 3 use classifier-free guidance at different scales, which moves each of them along a fidelity–coverage trade-off. Figure 23 pairs their FID with recall and precision. PJD reaches the highest precision (87.86%) at the lowest recall (43.62%); LPD moves from 78.84% to 77.84% precision and from 57.87% to 60.33% recall between its two settings. Since guidance selects a point on this curve, FID differences among guided recipes mix decoding quality with the chosen guidance, which is why Section 5.2 draws matched conclusions only from

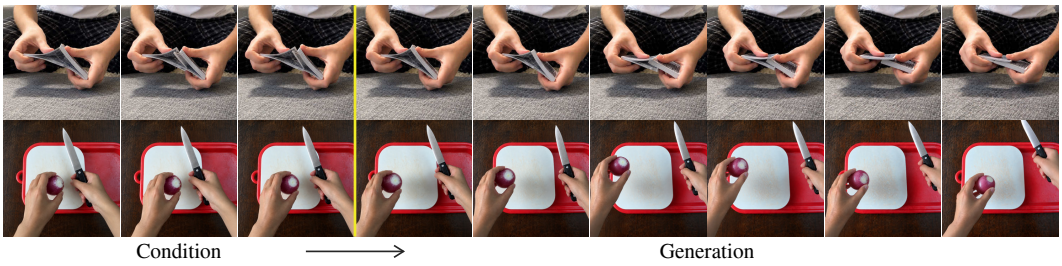


Figure 21: Video continuation from a short observed prefix. The K600 examples show shuffling cards (top) and chopping vegetables (bottom). Columns display frame indices 0, 2, 4, 6, 8, 10, 12, 14, and 16. The first three displayed instants belong to the five-frame input (indices 0–4); the remaining six show IntraAR predictions. The vertical boundary separates displayed conditions from generated frames.

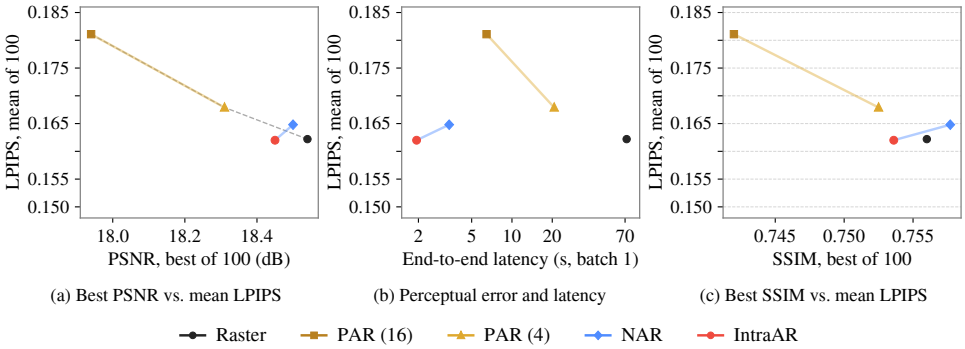


Figure 22: Candidate reconstruction and average perceptual error on BAIR. VideoGPT generates 100 candidates for each of 256 first-frame conditions, with four repetitions per seed and three seeds. PSNR and SSIM independently select their best candidate after averaging 16 frames; VGG v0.1 LPIPS averages all candidates and frames. Points are seed means. The center panel uses complete batch-1 latency on a logarithmic axis.

the unguided Raster/IntraAR pairs. Figure 24 shows four independent samples per decoder for two classes. All four decoders produce complete objects with the class's characteristic parts, so the differences in Table 3 are properties of the sample population that the metrics measure over 50,000 images per seed.

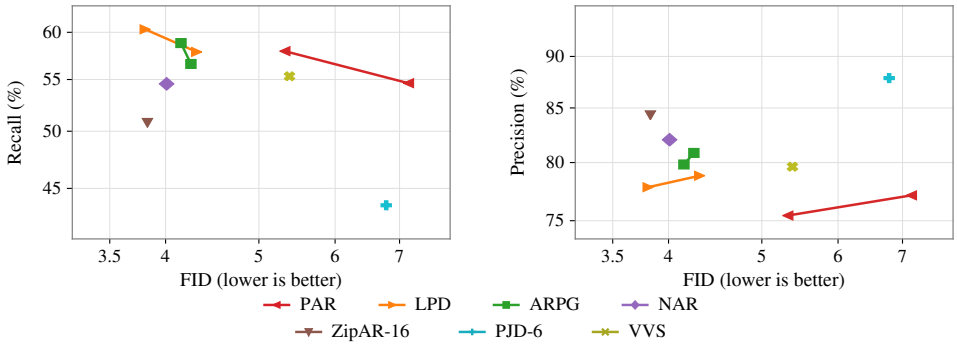


Figure 23: Fidelity and coverage of external ImageNet recipes. ImageNet/LlamaGen-XL working points pair FID with recall (left) and precision (right), using ADM features on 50,000 generated images per seed, 50 per class. Values average three seeds; precision/recall are percentages. Lines connect only PAR, LPD, or ARPG settings. Axes are logarithmic; recipes follow Appendix C.3 and values appear in Table 3.

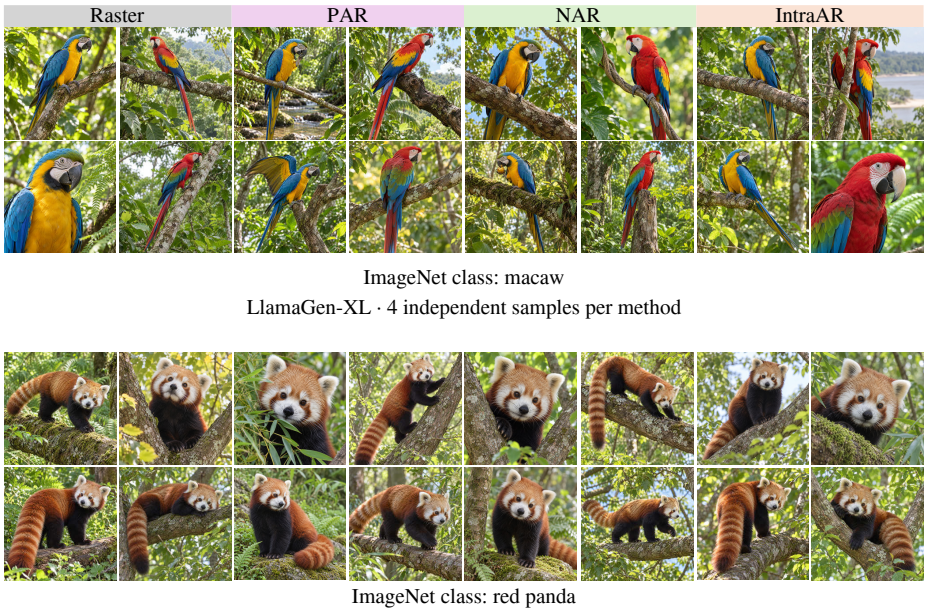


Figure 24: Same-class ImageNet samples across decoders. Raster, PAR, NAR, and IntraAR on LlamaGen-XL, four independent samples each, for macaw (top) and red panda (bottom). PAR and NAR use their guidance scales from Table 3; Raster and IntraAR are unguided.

F Why Coherence Is Judged on Complete Samples

Coherence is a joint property: the positions of a clip must agree on the choices that the condition leaves open. Two tempting shortcuts, matching a global statistic of real frames and assembling a frame from the appearance of earlier frames, miss exactly these choices, as the three controls below show on real images and videos. Section 5 therefore uses whole-clip Fréchet distances (I3D, VideoMAE-v2) and complete samples.

A global statistic can match while the scene is lost. Figure 25 takes frame 75 of the UCF101 clip `v_Archery_g16_c01` and projects random images onto its Fourier amplitude, its phase, or both, followed by the same RGB constraint. Matching the amplitude alone reproduces the target's spectrum up to a residual of about 2.14×10^{-16}, yet the image shows no person and no scene (RGB RMSE 0.367 on $[0, 1]$). The phase-only projection keeps edges and contours at their correct positions, and matching both recovers the target: the spatial layout lives in the phase, which a spectrum or other per-frame summary discards. In all three controls, copied or rearranged content passes the check, while the choices that make a clip coherent are decided jointly across positions.

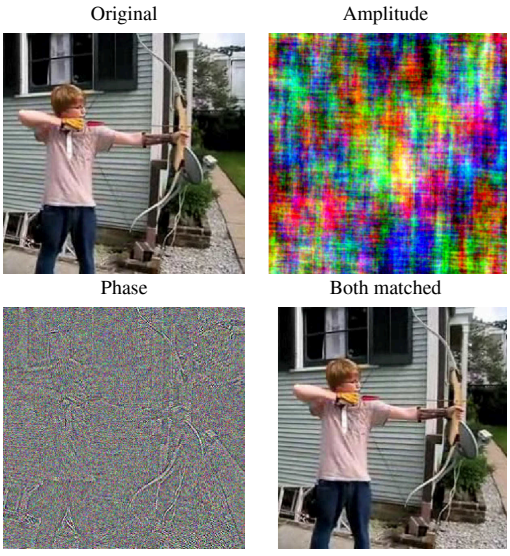


Figure 25: Matched amplitude and layout. The original Archery frame and projections that match its Fourier amplitude, its phase, or both.

Appearance copied from an earlier frame misses the new motion. Figure 26 reconstructs frames 9 and 17 of a UCF101 BaseballPitch clip by retrieving, for every patch, the most similar patch of

frame 1. The retrieval is even given the target frames. Background and static regions are recovered, but the moving leg (boxed) is not, and its retrieval offsets scatter in all directions. The content a future frame adds must be generated as a new choice.

Similar appearance does not identify the same part. Figure 27 queries a dog's head and body and a truck's body and wheel against other images of the class; the best matches often land on other parts or on the background, because a color-and-gradient descriptor does not know which part it describes.

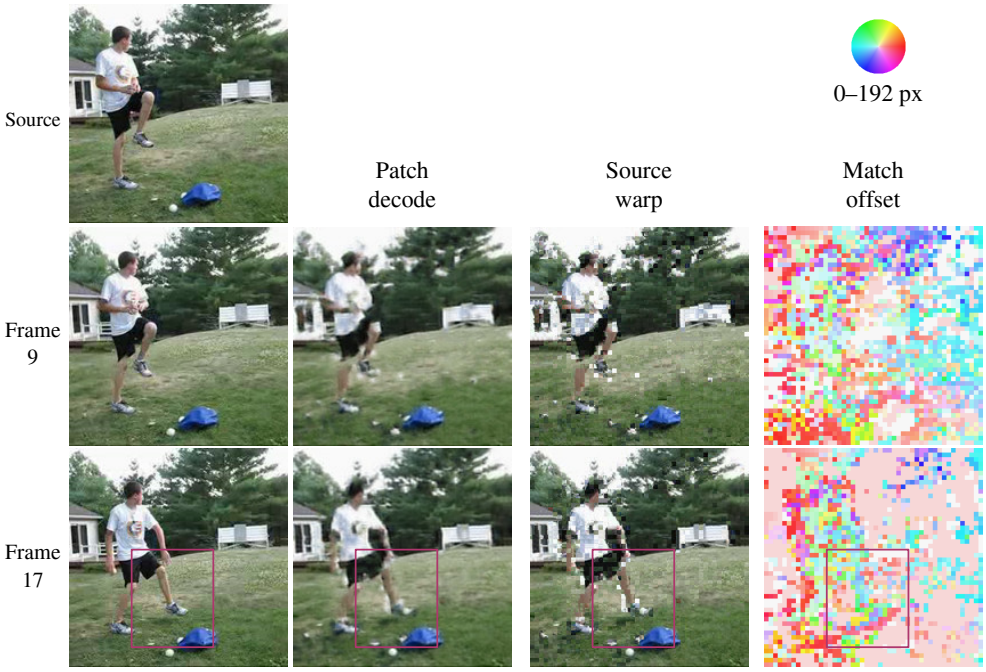


Figure 26: Fixed-source reconstruction on UCF101 BaseballPitch. Frames 9 and 17 rebuilt from 9^2 patches of frame 1 by cosine retrieval: patch decoding, RGB sampling with the same indices, and retrieval offsets (hue: direction). Boxes: leg regions.

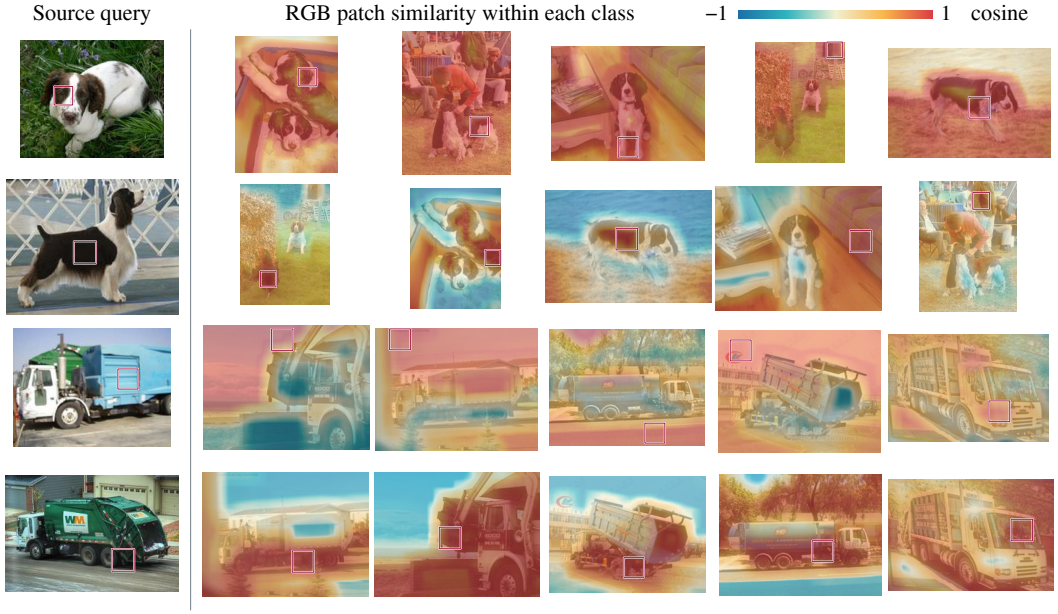


Figure 27: Within-class appearance correspondence on Imagenette. Dog head/body and truck body/wheel queries retrieve the five best-matching patches (boxes) in other images of the class; overlays show cosine similarity of a 14-dimensional color-and-gradient descriptor.