Interlaced Autoregressive Image Generation

Anonymous authors
Paper under double-blind review

Abstract

Autoregressive (AR) models inherently impose a 1D sequential dependency graph, which creates a fundamental structural mismatch when applied to 2D spatial manifolds like natural images. We introduce InterlacedAR, an autoregressive image generator that replaces the traditional left-to-right causal chain with a 2D spatial hierarchy. Inspired by classical graphics interlacing (*e.g.*, Adam7 in PNG), our method partitions the token lattice into interleaved sublattices and decodes them level by level: early levels establish a sparse, isotropic global context, while later levels perform highly parallelized local refinements conditioned on previously generated neighbors. This design strictly preserves an exact likelihood under our proposed level-wise factorization, avoiding the heuristic approximations or multi-scale token bloat of prior parallel AR methods. On ImageNet 256 × 256, InterlacedAR-XL achieves an FID of 3.26 in just 10 decoding steps, running over 16× faster than raster-scan baselines while retaining comparable fidelity. Ultimately, InterlacedAR demonstrates that structurally aligning the decoding schedule with the 2D nature of images unlocks massive parallelism without sacrificing the simplicity of standard decoder-only Transformer backbones.

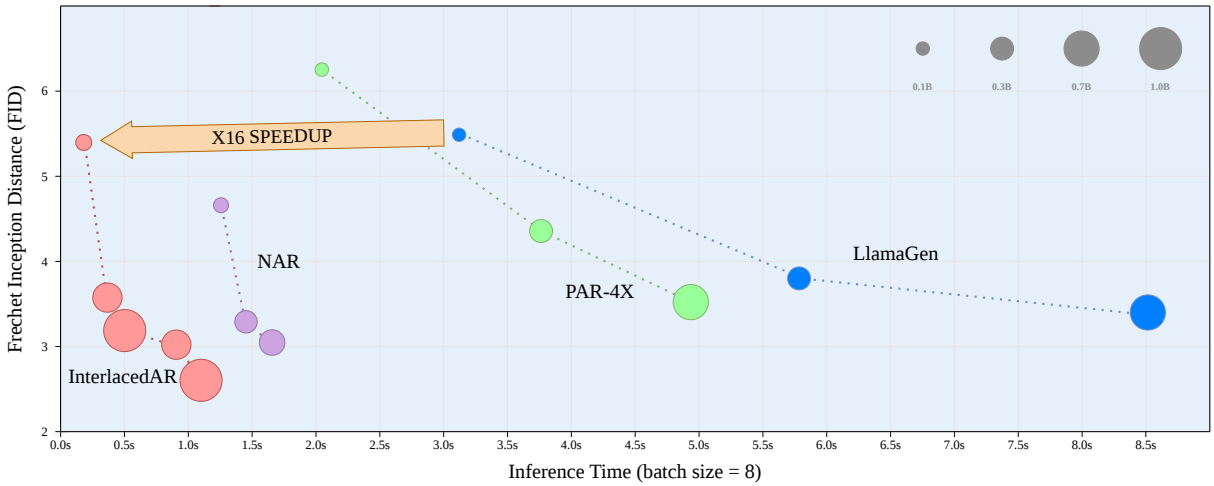


Figure 1: Class-conditional image generation quality and efficiency comparisons between various visual generation models. Different scales of the models from the same work are tested on the ImageNet 256×256 benchmark. We report InterlacedAR-256 (the leftmost three red dots) and InterlacedAR-384 (the remaining two red dots) to depict the complete trade-off.

1 Introduction

Autoregressive (AR) modeling has become the dominant paradigm for language generation: factorize a joint distribution into conditionals, train each by maximum likelihood, and sample token-by-token. Yet this recipe, so natural for inherently sequential text, fits images unnaturally. Language flows in one dimension where left-to-right order carries semantic meaning; images, however, extend in two dimensions where no

Table 1: Design comparison of AR image generation approaches. InterlacedAR is the only method with exact AR likelihood, standard tokenizer, 1× token count, and minimal decoding steps.

Method	Likelihood	Tokens	Tokenizer	Steps	Progressive
LlamaGen (Sun et al., 2024)	Exact AR	256	Standard	256	✗
VAR (Tian et al., 2024)	Exact AR	680	Multi-scale	10	✓
PAR (Wang et al., 2025)	Exact AR	256	Standard	64	✗
RandAR (Pang et al., 2025)	Exact AR	512	Standard	64	✗
MAR (Li et al., 2024b)	ELBO	256	Standard	64	✗
InterlacedAR	**Exact AR**	**256**	**Standard**	**10**	✓

canonical ordering exists. Flattening a 2D image into a 1D sequence, as in the standard raster-scan approach, imposes an arbitrary causality misaligned with spatial structure: context becomes anisotropic (right and below are always "future"), distant but semantically related regions must wait for intervening tokens, and the strong local correlations of natural images stretch into unnecessarily long dependency chains. The result is a decoding process that is both slow, requiring one forward pass per token, and statistically inefficient when many regions are nearly conditionally independent given their spatial neighbors.

Recent work addresses this bottleneck through alternative token orderings or parallel decoding schemes (He et al., 2025; Wang et al., 2025; Pang et al., 2025). However, these approaches often sacrifice the standard AR likelihood, require specialized multi-scale tokenizers (Tian et al., 2024), or introduce extra tokens beyond the target image grid. A natural question arises: can we preserve the simplicity and exact likelihood of decoder-only AR while exploiting 2D structure to dramatically reduce decoding steps?

We find an answer in a perhaps unexpected place: *progressive image rendering*. Three decades ago, the Adam7 algorithm (Roelofs & Koman, 1999) solved an analogous problem for image transmission, namely how to display a useful preview before all pixels arrive. Its solution was *interlacing*: partition the pixel grid into a sequence of sparse, interleaved sublattices arranged coarse-to-fine, transmit each in turn, and progressively refine the display. We observe that the same structure is ideally suited to autoregressive generation. We propose **InterlacedAR**, an autoregressive image generator that adopts interlacing as its factorization order. The idea is conceptually simple: generation proceeds through a small number of *interleaved passes*, starting from a sparse grid that captures global structure, then progressively predicting denser sublattices conditioned on the evolving canvas. Within each pass, all tokens are predicted jointly in a single forward step (*i.e.*, *block-parallel decoding*), reducing sequential rounds from $O(HW)$ to the number of passes. For a 16×16 token grid, this means 10 forward passes instead of 256, a **25**× reduction in sequential steps, while producing exactly the same set of image tokens as raster-scan AR.

Beyond computational efficiency, InterlacedAR admits a principled interpretation as *stochastic inverse synthesis*. The interlacing schedule defines a deterministic *analysis* operator that partitions an image-token grid into disjoint masked subbands (one per pass) and an exact *synthesis* operator that reassembles them by summation. Synthesis perfectly inverts analysis because the masks partition the lattice. Generation inverts this structure stochastically: at each pass, the model predicts the unknown subband conditioned on already-synthesized subbands and the class label, then deterministically assembles it onto the canvas. This viewpoint formalizes why interlacing is effective: it factors the generation problem into a sequence of subband predictions, each well-constrained by the coarser structure already in place.

Crucially, InterlacedAR is *drop-in compatible* with existing visual tokenizers and decoder-only Transformer backbones. Implementation requires only an interlaced attention mask that conditions each pass on previously generated tokens, plus lightweight pass embeddings (alongside standard class conditioning) so a single backbone can model all pass-specific distributions. Unlike approaches that process twice as many tokens as the target image grid (Tian et al., 2024; Pang et al., 2025), InterlacedAR operates directly on the standard token count with no architectural modifications (Table 1). On ImageNet-1K 256×256, InterlacedAR achieves a favorable quality-efficiency trade-off (Figure 1). Compared to size-matched LlamaGen-L, InterlacedAR-L reduces decoding from 256 to 10 steps with comparable FID (*e.g.*, 3.80 → 3.64). Our largest model achieves

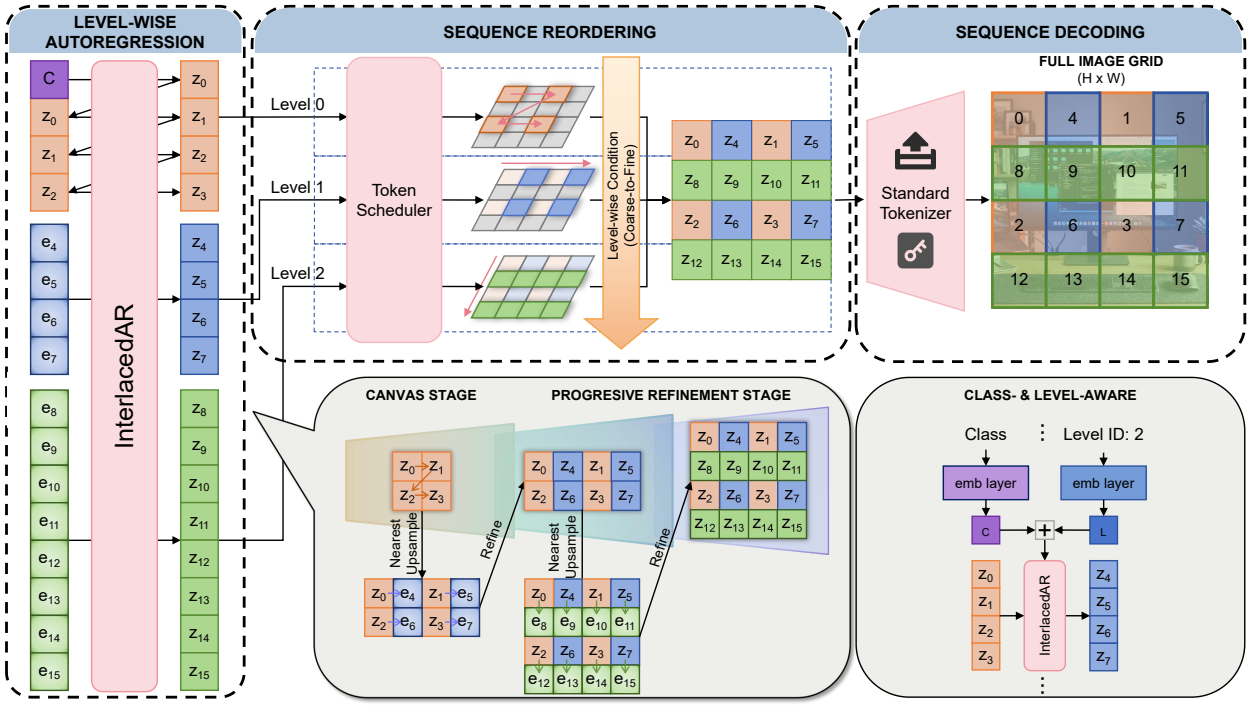


Figure 2: InterlacedAR framework. Tokens are generated by level-wise autoregression. The output sequence is reordered to the original raster-order. Tokens from different levels will be decoded by a **standard** VQ-VAE decoder. Level-wise autoregression process can be further illustrated as: **Canvas Stage**: Level-0 tokens (0–3) are decoded autoregressively, establishing global structure. **Progressive Refinement Stage**: The canvas is upsampled via nearest-neighbor interpolation to initialize query embeddings (*e.g.*, *e*4–*e*7) at unknown positions. The Interlaced AR model then refines all queries in a single forward pass, predicting the final tokens conditioned on the canvas. This upsample-refine loop repeats until all positions are filled. The InterlacedAR block is conditioned on both class (or prompt for t2i task) and level.

FID 3.26 with 10 steps and an FID of 2.60 at 384×384 using 15 steps, significantly surpassing accelerated baselines like PAR (Wang et al., 2025) in the quality-latency trade-off.

Contributions. **(1)** We introduce an interlaced AR factorization that preserves exact likelihoods while enabling block-parallel decoding in 10 passes instead of 256 sequential steps, for 256 × 256 resolution image generation. **(2)** We formalize this as stochastic inverse synthesis, casting generation as learned subband prediction under an exact analysis/synthesis structure, and realize it with a simple decoder-only Transformer. **(3)** Experiments on ImageNet demonstrate that InterlacedAR achieves comparable fidelity to strong AR baselines despite the 25× reduction in sequential decoding rounds and the 16× inference speed-up.

2 Related Work

Visual Autoregressive Models. Autoregressive modeling has been a central thread in generative vision. PixelRNN (van den Oord et al., 2016) and PixelCNN (Van den Oord et al., 2016; Salimans et al., 2017) demonstrated that recurrent or masked-convolutional architectures can parameterize an exact likelihood over images while conditioning on external information. Subsequent work improved the expressiveness and scalability of visual autoregressive modeling. PixelSNAIL (Chen et al., 2018) combined autoregressive convolutions with self-attention to better capture both local and long-range structure. Image Transformer (Parmar et al., 2018) and Sparse Transformer (Child et al., 2019) further showed that self-attention based autoregressive models can scale effectively to image generation. Inspired by progress in language modeling, iGPT (Chen et al., 2020) showed that a Transformer (Vaswani et al., 2017) trained autoregressively on sequences of

pixels can learn strong visual representations and produce coherent images. Subsequent work explored scaling decoder-only transformers and simplifying the pipeline. LlamaGen (Sun et al., 2024) reported that sufficiently scaled decoder-only transformers can rival or surpass diffusion models in quality, which reasserted the competitiveness of autoregression at scale.

AR Models with Parallel Decoding. Masked generative models (Chang et al., 2022; Li et al., 2024b) formulated image synthesis as masked token prediction with iterative refinement, which preserves many advantages of AR training while enabling highly parallel decoding at sampling time. Additional masked and iterative generative models such as MAGE (Li et al., 2022) and Muse (Chang et al., 2023) further demonstrate the effectiveness of parallel token prediction, though they generally optimize denoising- or reconstruction-style objectives rather than the standard autoregressive likelihood.

Closely related coarse-to-fine and non-raster autoregressive formulations have also been explored. Multiscale PixelCNN (Reed et al., 2017) generated images in a coarse-to-fine pyramid, reducing long dependency chains through multi-resolution prediction. Subscale Pixel Networks with multidimensional upscaling (Menick et al., 2018) modeled high-resolution images by generating interleaved spatial slices and then refining them, illustrating the benefits of non-raster spatial decompositions for scalable likelihood-based generation. VAR (Tian et al., 2024) proposed next-scale prediction in an image pyramid so that models forecast coarse structure before fine detail, which shortens the effective sequence and improves quality through a coarse-to-fine trajectory but requires a dedicated multi-scale residual tokenizer and incurs roughly double the number of tokens compared to single-scale models such as LlamaGen (Sun et al., 2024). RandAR (Pang et al., 2025) trains on randomized generation orders and decodes multiple positions at once, which requires conflict resolution and introduces additional sequence tokens representing sampling order, increasing effective sequence length and potentially compromising consistency. PAR (Wang et al., 2025) predicted blocks in chunks under an autoregressive objective, but the generation process is still one-by-one with each block, failing to adaptively increase parallelism when more information is decoded. ARPG (Li et al., 2025) sampled in randomized parallel with reconciliation mechanisms that traded speed for additional heuristics. NAR (He et al., 2025) instead restricts dependencies to local neighborhoods and predicts neighboring tokens in parallel, achieving efficient visual generation without multi-scale tokenization.

A common theme in these approaches is the trade-off between parallelism and either (1) architectural complexity, such as multi-scale tokenizers or auxiliary decoders, (2) increased token counts that offset speed gains, or (3) approximate likelihoods that complicate training. InterlacedAR sidesteps these trade-offs by observing that the spatial structure of images already provides a natural factorization: coarse-to-fine interlacing groups tokens into levels where within-level parallelism is justified by strong local correlations, without requiring architectural modifications or extra tokens. Our work targets a different axis: we exploit coarse-to-fine interlacing to parallelize decoding while preserving a standard decoder-only architecture and the usual autoregressive likelihood, avoiding multi-scale tokenizers, auxiliary decoders, or block-specific heads as highlighted in Table 1.

3 Method

We present InterlacedAR, an autoregressive image generator that follows a coarse-to-fine interlacing schedule. As illustrated in Figure 2, generation begins with a *canvas stage* that autoregressively decodes a sparse set of tokens capturing global structure, followed by a *progressive refinement stage* that iteratively upsamples the canvas and predicts newly introduced positions in a block-parallel fashion.

3.1 Overview: Stochastic Inverse Synthesis

The Adam7 algorithm (Roelofs & Koman, 1999) for progressive image rendering addresses an analogous problem: how to provide a useful preview before all pixels arrive. Its solution is to partition pixels into interleaved sublattices and reveal them from coarse to fine. We adopt the same structural idea for generative modeling and refer to the resulting viewpoint as *stochastic inverse synthesis*.

Let $\mathbf{z} = (z_0, \ldots, z_{T-1})$ denote image tokens on an $H \times W$ grid, where each $z_t \in \{0, \ldots, V-1\}$ indexes a discrete codebook learned by a VQ-VAE (Van Den Oord et al., 2017; Razavi et al., 2019). A standard

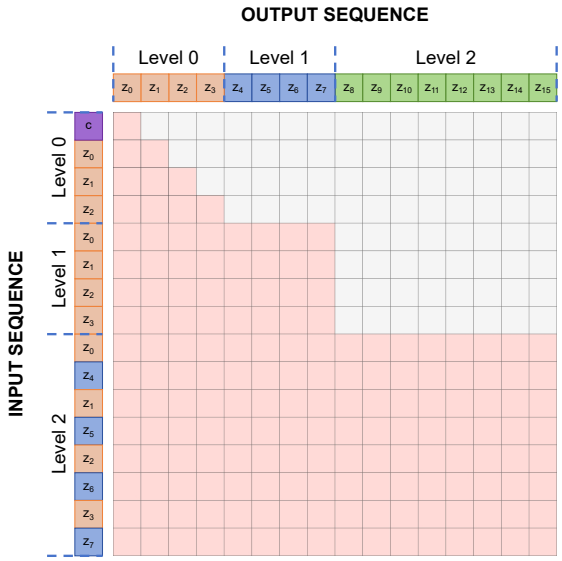


Figure 3: Attention mask structure enforcing no same-level dependencies for levels $k \geq 1$. For example, when predicting z_{10} within level 2, the model cannot attend to z_8 as they reside in the same parallel decoding level. Colored tiles indicate accessible positions within the attention matrix, where tokens in level 0 are generated **autoregressively**. The output token sequence is sampled in an interlaced spatial order, where missing tokens at the current level are conditioned on a context populated by nearest-neighbor upsampling of previously synthesized levels.

raster-scan autoregressive model factorizes the joint distribution as:

$$p_\theta(\mathbf{z} \mid \mathbf{c}) = \prod_{t=0}^{T-1} p_\theta(z_t \mid \mathbf{z}_{<t}, \mathbf{c}), \tag{1}$$

along a one-dimensional scan order, where $\mathbf{c}$ denotes conditioning information such as a class label. Sampling under this factorization requires $T = H \cdot W$ sequential decoding steps.

InterlacedAR instead partitions the token lattice into K disjoint subsets $\{G_0, \ldots, G_{K-1}\}$ arranged from coarse to fine. This partition defines a deterministic *analysis* operator:

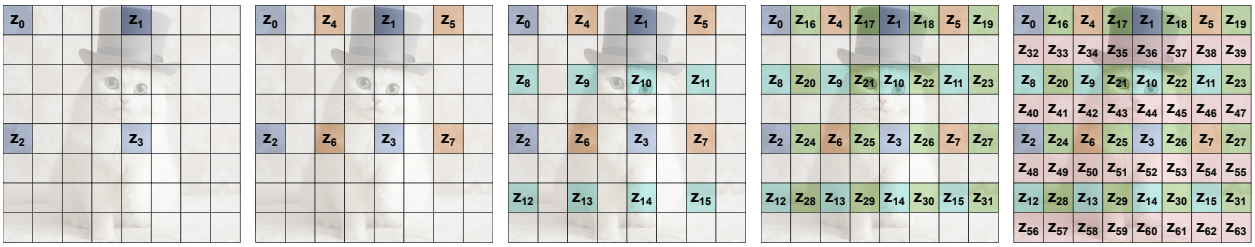
$$\mathbf{z}^{(k)} = M_k \odot \mathbf{z}, \quad \text{where} \sum_{k=0}^{K-1} M_k = \mathbf{1}, \tag{2}$$

with each M_k a binary mask selecting positions in level k. The corresponding *synthesis* operator reconstructs the full grid by summing the subbands:

$$\mathbf{z} = \sum_{k=0}^{K-1} \mathbf{z}^{(k)}. \tag{3}$$

Because the masks partition the lattice, synthesis exactly inverts analysis: no information is discarded or duplicated. InterlacedAR therefore defines an exact likelihood under its own level-wise factorization over the same token grid, rather than under the standard raster order.

Generation proceeds by stochastic inverse synthesis: at each level k, the model predicts the unknown subband $\mathbf{z}^{(k)}$ conditioned on previously synthesized subbands $\hat{\mathbf{z}}_{<k}$ and a conditioning signal $\mathbf{c}$ (*e.g.*, a class label), then assembles the prediction onto the canvas. Figure 2 illustrates this process as an iterative upsample-refine loop (central lower part): known tokens are scattered onto a finer grid, query tokens are predicted, and the updated canvas is assembled. This interpretation is useful because it makes explicit that the method changes the factorization order while keeping the support and reconstruction target unchanged.



(a) Level 0 (b) Level 1 (c) Level 2 (d) Level 3 (e) Level 4

Figure 4: Interlacing decoding schedule. Different colors denote different decoding levels. In this example, the token grid is partitioned into five levels. Levels are decoded sequentially, while tokens within a non-initial level are decoded in parallel conditioned on previously generated levels. Level 0 is decoded autoregressively.

3.2 Interlaced AR

The effectiveness of interlacing relies on a basic statistical property of natural images: nearby regions are strongly correlated. When a token is surrounded by known neighbors from coarser levels, its uncertainty is reduced, and prediction becomes a constrained local refinement problem rather than an unconstrained global one. This motivates a factorization in which later levels can be decoded in parallel once a sufficiently informative coarse scaffold has been established.

Setup. Let the image token grid be partitioned into interleaved levels $\{\mathcal{G}_0, \ldots, \mathcal{G}_{K-1}\}$ from coarsest to densest, and define the set of previously generated tokens before level k as $\mathbf{z}_{<k} \triangleq \{z_s : s \in \bigcup_{j<k} \mathcal{G}_j\}$.

Interlacing Decoding Schedule. A practical way to define these levels is through checkerboard-like or offset sampling patterns over spatial indices (x, y). For the example in Figure 4, the image is tokenized into an 8×8 grid and the base interlacing block size (**BS**) is set to 4×4. Here, **BS** denotes the spatial stride between tokens assigned to the coarsest level along each axis. Tiling this 4×4 pattern across the image yields $K = 5$ decoding levels:

1. **Level 0 ($\mathcal{G}_0$, autoregressive generation):** Tokens are selected by strongly subsampling the full grid. In the 8×8 example, tokens at $(0, 0)$, $(4, 0)$, $(0, 4)$, and $(4, 4)$ correspond to z_0, z_1, z_2, z_3 ($\mathcal{G}_0 = \{z_0, z_1, z_2, z_3\}$). This first pass captures coarse global structure.
2. **Levels $k > 0$ ($\mathcal{G}_k$, parallel generation):** Subsequent levels fill the remaining positions between already-generated tokens. For a $K = 5$ schedule:
   - $\mathcal{G}_1$ fills horizontal half-step positions, *e.g.*, $(2, 0)$, $(6, 0)$, $(2, 4)$, and $(6, 4)$, corresponding to z_4, z_5, z_6, z_7.
   - $\mathcal{G}_2$ fills vertical half-step positions, *e.g.*, $(0, 2)$, $(2, 2)$, $(4, 2)$, $(6, 2)$, $(0, 6)$, $(2, 6)$, $(4, 6)$, and $(6, 6)$, corresponding to $z_8, \ldots, z_{15}$.

The level sequence is constructed so that the already-generated tokens $\mathbf{z}_{<k}$ always form a sparse but regular grid. This ensures that each new prediction is conditioned on spatially nearby context and that intermediate previews remain spatially coherent.

Interlaced AR Modeling. We enforce *no same-level dependencies* through masked attention, as illustrated in Figure 3. Under this parameterization, tokens within a level are treated as conditionally independent given previously generated levels:

$$p_\theta(\mathbf{z}) = \underbrace{\prod_{t \in \mathcal{G}_0} p_\theta(z_t \mid z_{<t}, \mathbf{c})}_{\text{Canvas Stage}} \times \underbrace{\prod_{k=1}^{K-1} p_\theta(\mathbf{z}_{\mathcal{G}_k} \mid \mathbf{z}_{<k}, \mathbf{c})}_{\text{Progressive Refinement Stage}} \quad ,$$

where condition $\mathbf{c}$ is the addition of class/text embedding and level id embedding, which is consistent with the right lower part of Figure 2.

Tokens in $\mathcal{G}_0$ are decoded one by one, corresponding to the *Canvas Stage* in Figure 2. For each later level, all tokens in $\mathcal{G}_k$ are predicted in parallel while preserving a coarse-to-fine causal order across levels. Training uses the corresponding masked objective:

$$\mathcal{L}_{\text{Interlace}}(\theta) = -\,\mathbb{E}_{(\mathbf{z},\mathbf{c})}\left[\sum_{t\in\mathcal{G}_0} \log p_\theta(z_t \mid z_{<t}, \mathbf{c}) \;+\; \sum_{k=1}^{K-1} \log p_\theta(\mathbf{z}_{\mathcal{G}_k} \mid \mathbf{z}_{<k}, \mathbf{c})\right].$$

We optimize this objective with standard cross-entropy over discrete VQ-VAE token indices. The level-based mask ensures that each prediction is conditioned only on the information allowed by the proposed factorization.

InterlacedAR Backbone. We implement this factorized distribution with a class- and level-aware GPT-style backbone. The Transformer blocks use AdaLN (Peebles & Xie, 2023), with modulation driven by the combination of a class embedding and a level embedding. Each token receives a level embedding corresponding to its decoding level, and this signal is aligned with the input token sequence before AdaLN modulation. This conditioning allows a single backbone to model all levels while respecting coarse-to-fine causality across levels and parallel prediction within each level.

Heuristic Spatial Ordering. Although tokens within each level are predicted in parallel, the decoder still consumes a serialized input sequence. We therefore define a *Heuristic Spatial Ordering* (HSO) for the input token indices within each level $\mathcal{G}_k$, as illustrated in Figure 2. The query embeddings at unknown positions are initialized by nearest upsampling, so that, e_4 in Figure 2 (an unknown position in level 1) is either z_0 or z_1. To break ties, we apply a consistent *min-idx* rule, resulting in $e_4 = z_0$. Giving another example with more generation levels, in Figure 4, for each output token $z_t \in \mathcal{G}_k$, we choose the spatially nearest token $z_s \in \mathbf{z}_{<k}$ as its associated context token in the input sequence. For example, before sampling z_9 in Figure 4, tokens z_4 and z_6 are equidistant, and are sampled in the previous level; we select z_4 because it has the smaller index. This ordering encourages the backbone to place nearby spatial context close to each query in the serialized representation while keeping the model architecture unchanged.

Error Propagation. We argue that InterlacedAR offers a more robust error-handling mechanism than raster-scan AR. In raster-order token generation scheme, errors accumulate unidirectionally along a 1D sequence, leading to structural drift where the bottom of an image may become semantically disconnected from the top. In contrast, InterlacedAR's Level 0 establishes a sparse but uniform 2D skeleton across the entire canvas. This changes the nature of error propagation from linear accumulation to localized refinement. Each subsequent token is conditioned on multi-directional neighbors that anchor it to the global structure, effectively confining potential sampling errors within small local regions and preventing large-scale structural collapse. See §4.4 for detailed analysis.

3.3 Complexity and Parallelism Analysis

Let the image be tokenized into a grid of $H \times W$ discrete codes, giving $T = H \times W$ autoregressive positions. A standard raster-scan decoder must apply the Transformer T times, once per token, which quickly becomes the dominant cost at sampling time. In contrast, InterlacedAR partitions the grid into $K = 2 \times \log \mathbf{BS} + 1$ interlaced levels $\{G_0, \ldots, G_{K-1}\}$ as described in §3.2. Only the coarsest level G_0 is decoded sequentially; all later levels G_k with $k > 0$ are filled in a small number of block-parallel rounds given the coarse context $\mathbf{z}_{<k}$.

For a base interlacing block size $\mathbf{BS}$, the coarsest level G_0 contains approximately $(H/\mathbf{BS}) \times (W/\mathbf{BS})$ tokens, while the remaining $T - |G_0|$ positions are spread across the finer levels. The resulting number of decoding rounds scales as $O(|G_0| + K) = O(\frac{T}{\mathbf{BS}^2} + \log \mathbf{BS})$ rather than $O(T)$: the backbone GPT is applied once per token in G_0 and then once per level to fill all remaining positions in parallel. In the default ImageNet setting with $H = W = 16$ and $\mathbf{BS} = 8$, this reduces the number of decoding steps from 256 for a raster-scan LlamaGen baseline to only 10 steps for InterlacedAR, while covering exactly the same 16×16 token grid.

Because each level G_k for $k > 0$ is decoded in parallel, the effective batch size seen by the backbone can be much larger than the physical batch of images: at the finest level the Transformer processes all remaining positions across all images in a single forward pass. Together with standard KV caching, this yields substantial speedups without changing the backbone architecture. Unlike multi-scale methods that introduce auxiliary tokens at additional resolutions (Tian et al., 2024), InterlacedAR does not increase the total number of

processed tokens; the FLOPs per forward pass remain comparable to raster AR, and the gains come primarily from reducing the number of decoding rounds.

System Throughput and KV-Cache Efficiency. In practice, the efficiency gains of InterlacedAR extend beyond the reduction of sequential steps. In standard AR, the KV-cache must be updated and fetched for every single token, creating a significant memory bandwidth bottleneck. In InterlacedAR's refinement stages, tokens within the same level are computed in a single GPU kernel call. This reduces the KV-cache synchronization frequency by up to 25×, allowing the model to better utilize the parallel compute units of the GPU and significantly increasing throughput, especially in large batch-size inference scenarios.

4 Experiment

Table 2: Comparison of class-conditional image generation methods on ImageNet 256 × 256. All experiments use top-$k = 0$, top-$p = 1.0$, and temperature = 1.0. Images generated at 384 × 384 are resized to 256 × 256 for evaluation. Metrics include Fréchet inception distance (FID), inception score (IS), precision, and recall. ↓ and ↑ indicate whether lower or higher is better. Reported numbers for NAR (He et al., 2025), RandAR (Pang et al., 2025), and PAR (Wang et al., 2025) are taken from (He et al., 2025). FLOPs are measured per generated image, and time is measured per batch of 8 images.

Type	Model	Tokenizer	Resolution	#Para.	Steps	FLOPs	FID↓	IS↑	Prec.↑	Recall↑	Time (s)
Mask	MaskGIT (Chang et al., 2022)	-	256	227 M	8	-	6.18	182.1	0.80	0.51	-
	MAR-B (Li et al., 2024a)	Standard	256	208 M	100	-	2.31	281.7	0.82	0.57	-
	MAR-L (Li et al., 2024a)	Standard	256	479 M	100	-	1.78	296.0	0.81	0.60	-
VAR	VAR-d16 (Tian et al., 2024)	Multi-scale	256	310 M	10	563.54G	3.30	274.4	0.84	0.51	0.49
	VAR-d20 (Tian et al., 2024)	Multi-scale	256	600 M	10	1.09T	2.57	302.6	0.83	0.56	0.61
	VAR-d24 (Tian et al., 2024)	Multi-scale	256	1.0 B	10	1.88T	2.09	312.9	0.82	0.59	0.70
AR (Raster)	LlamaGen-B (Sun et al., 2024)	Standard	256	111M	256	104.85G	5.46	193.6	0.83	0.45	3.12
	LlamaGen-L (Sun et al., 2024)	Standard	256	343M	256	346.18G	3.80	248.2	0.83	0.51	5.81
	LlamaGen-L (Sun et al., 2024)	Standard	384	343M	576	815.15G	3.07	256.1	0.83	0.52	12.58
	LlamaGen-XL (Sun et al., 2024)	Standard	256	775M	256	790.92G	3.39	227.1	0.81	0.54	8.54
	LlamaGen-XL (Sun et al., 2024)	Standard	384	775M	576	1.86 T	2.62	244.0	0.80	0.57	18.66
	LlamaGen-XXL (Sun et al., 2024)	Standard	384	1.4B	576	-	2.34	253.9	0.80	0.59	24.91
	PAR-B-4X (Wang et al., 2025)	Standard	256	111M	67	-	6.21	204.4	0.86	0.39	2.04
	PAR-L-4X (Wang et al., 2025)	Standard	256	343M	67	-	4.32	189.4	0.87	0.43	3.75
	PAR-XL-4X (Wang et al., 2025)	Standard	256	775M	67	-	3.50	234.4	0.84	0.49	4.94
AR (Random)	RandAR-L (Pang et al., 2025)	Standard	256	343M	88	-	2.55	288.8	0.81	0.58	3.20
	RandAR-XL (Pang et al., 2025)	Standard	256	775M	88	-	2.25	317.7	0.80	0.60	4.73
AR (Neighboring)	NAR-B (He et al., 2025)	Standard	256	130M	31	-	4.65	212.3	0.83	0.47	1.24
	NAR-M (He et al., 2025)	Standard	256	290M	31	-	3.27	257.5	0.82	0.53	1.41
	NAR-L (He et al., 2025)	Standard	256	372M	31	-	3.06	263.9	0.81	0.53	1.68
AR (Interlacing)	InterlacedAR-B	Standard	256	153M	10	148.19G	5.39	261.8	0.84	0.43	0.18
	InterlacedAR-L	Standard	256	494M	10	500.41G	3.64	262.5	0.81	0.52	0.33
	InterlacedAR-L	Standard	384	494M	15	1.17T	3.12	284.8	0.84	0.47	0.80
	InterlacedAR-XL	Standard	256	1.0B	10	1.14T	3.26	274.9	0.83	0.46	0.52
	InterlacedAR-XL	Standard	384	1.0B	15	2.47T	2.60	266.3	0.84	0.54	1.12

4.1 Experimental Setup

We evaluate InterlacedAR on class-conditional ImageNet generation at 256 × 256 and 384 × 384 resolutions. We report Fréchet inception distance (FID, 50k samples), inception score (IS), and precision/recall.

Architecture and Tokenizer. We use the VQ-VAE tokenizer from LlamaGen (Sun et al., 2024), with vocabulary size 16,384 and a 16× downsampling factor. The InterlacedAR backbone follows the LLaMA-style architecture (Touvron et al., 2023) at B, L, and XL scales, using RMSNorm, SwiGLU, and 2D-RoPE. Unless otherwise stated, the base interlacing block size is set to **BS**=8.

Training. Models are trained for 400 epochs with batch size 2048 using AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay = 0.05). We linearly warm up the learning rate to 4×10^{-4} during the first 100 epochs and then cosine-decay it to 1×10^{-5} over the remaining 300 epochs. Data augmentation uses ten-crop with one crop sampled per epoch. For classifier-free guidance (CFG) (Ho & Salimans, 2022), class embeddings are dropped with probability 0.1, together with token dropout of 0.1.

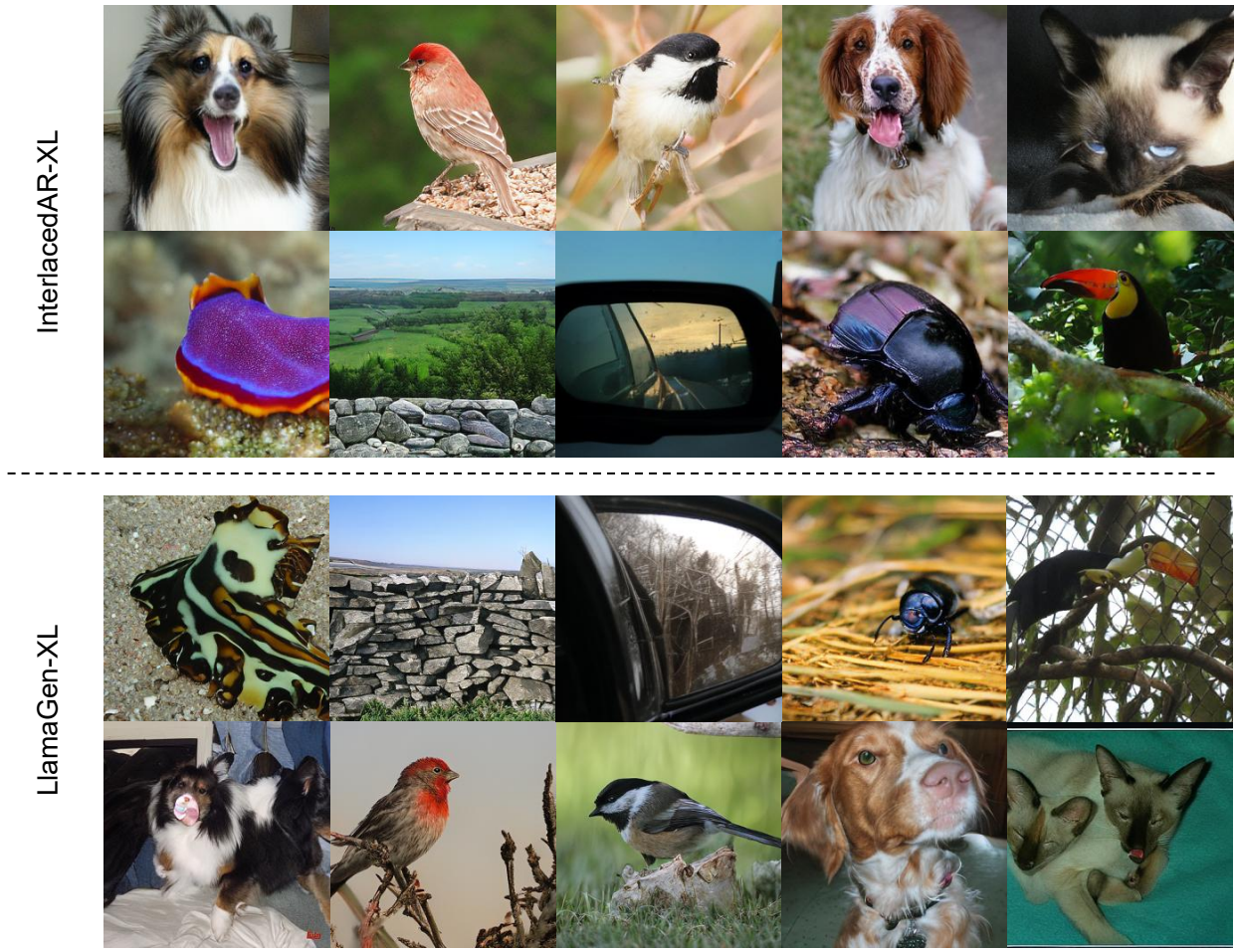


Figure 5: Class-conditional samples from InterlacedAR-XL and LlamaGen-XL (Sun et al., 2024) for the same set of ImageNet classes.

Inference. By default, InterlacedAR uses 10 decoding steps at 256×256 and 15 decoding steps at 384×384, both with **BS**=8. We use standard KV caching and report wall-clock time for a batch size of 8. Following prior work (Wang et al., 2025; He et al., 2025), timing includes the Transformer backbone but excludes the comparatively small cost of the VQ-VAE decoder. All evaluations use temperature 1.0, top-$k = 0$, top-$p = 1.0$. CFG scale is set to a constant during each inference.

4.2 Class-conditional Image Generation

Table 2 compares InterlacedAR with representative image generation baselines on ImageNet. Overall, InterlacedAR occupies a favorable point in the quality-latency trade-off: it uses substantially fewer decoding rounds than raster-scan autoregressive models while remaining competitive in FID.

When comparing methods with similar numbers of decoding rounds, InterlacedAR-B is notably faster than VAR-d16 (0.18s vs. 0.49s per batch of 8 images), while avoiding the additional token overhead of multi-scale tokenization. When comparing methods with similar backbone scale, InterlacedAR-L reduces decoding from 256 to 10 steps relative to LlamaGen-L and achieves a slightly better FID (3.64 vs. 3.80). These results indicate that the interlaced factorization can convert computation into sampling speed more efficiently than raster-scan decoding on this benchmark.

InterlacedAR also remains competitive in image quality at larger scales. InterlacedAR-XL achieves FID 3.26 at 256×256 using 10 decoding steps, and InterlacedAR-XL reaches FID 2.60 at 384×384 using 15

steps. Some baselines, such as NAR or RandAR, attain lower FID in certain settings, but they do so with substantially more decoding steps and higher latency. For example, at 256×256, NAR-L uses 31 steps and 1.68s per batch, whereas InterlacedAR-L uses 10 steps and 0.33s.

Across model scales and resolutions, the same trend remains visible: increasing model size improves fidelity, while the interlaced schedule keeps the number of sequential rounds small. At 384×384, InterlacedAR-L and -XL run in 0.80s and 1.12s, respectively, whereas LlamaGen-L and -XL require 12.58s and 18.66s. Taken together, these results suggest that InterlacedAR provides a practical operating point for low-latency autoregressive image generation without requiring multi-scale tokenizers or additional sequence tokens.

4.2.1 Qualitative Comparisons

Figure 5 shows side-by-side class-conditional samples from InterlacedAR-XL and LlamaGen-XL (Sun et al., 2024) for the same ImageNet classes. Both models produce diverse and visually plausible samples. The qualitative differences are generally modest, which is consistent with the relatively small FID gap in Table 2. These examples suggest that the substantial reduction in decoding rounds does not introduce obvious visual artifacts in the final samples.

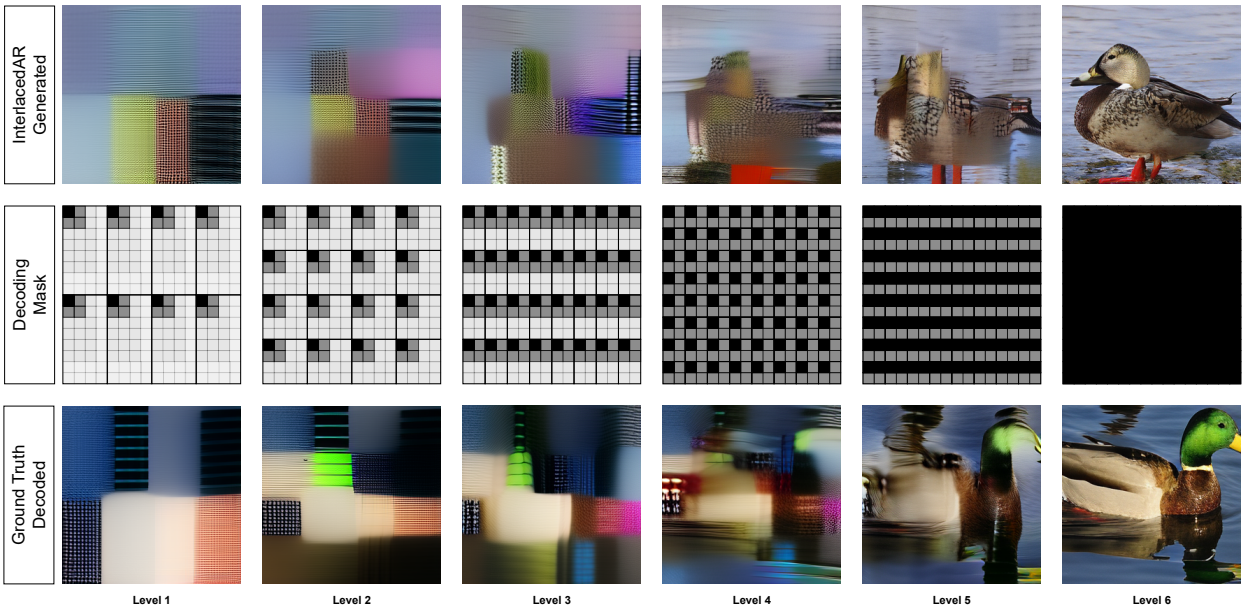


Figure 6: Level-wise decoding comparison between InterlacedAR-XL samples and ground-truth image tokens for the same class. Missing lattices within each level are filled by nearest-neighbor interpolation.

4.2.2 Level-wise Decoding Comparison and Structural Robustness

Figure 6 visualizes the progressive refinement process across the final six levels of the InterlacedAR decoding schedule ($\mathbf{BS} = 8$, image size 256). To visualize the intermediate semantic states, we populate ungenerated positions using nearest-neighbor interpolation from the current level's anchors and pass the resulting grid through the VQ decoder.

Analysis of Early-Stage Artifacts. As observed in the early levels (e.g., Level 1 and 2), the rendered images exhibit visible blocky artifacts. We emphasize that these artifacts stem from a domain gap: the pre-trained VQ decoder is optimized for dense 16×16 (or larger) token grids, whereas the early Canvas Stage provides a highly sparse representation. This is further validated by the Ground Truth Decoded intermediate representation, which shows nearly identical distortions when the same sparse masks are applied to ground-truth tokens. Crucially, while these early levels lack high-frequency photorealism, they successfully establish the global semantic scaffolding, such as the placement of objects and the overall color palette.

Table 3: Ablation on base interlacing block size (**BS**). Inference time is measured per batch of 8 images. Lower FID is better. All models are trained for **100 epochs**.

Model	BS	Steps	FID↓	Time (s)↓
LlamaGen	–	256	7.18	3.12
InterlacedAR	1	256	6.97	3.08
InterlacedAR	2	66	7.23	0.88
InterlacedAR	4	20	7.65	0.36
InterlacedAR	8	10	7.75	0.17

Table 4: Ablation on HSO strategies. Lower FID is better. All models are trained for **100 epochs** with different HSO strategies.

HSO Strategy	#Para.	FID↓
closest & min idx	111M	7.12
closest & max idx	111M	7.34
closest & left up	111M	7.22
unattached & max idx	111M	24.98

Mitigating Error Propagation via 2D Anchoring. The progressive visualization provides empirical evidence for why InterlacedAR is robust against the long-range structural drift common in raster-scan models. In a traditional raster-order sequence, an error in an early token propagates unidirectionally, often leading to global incoherence by the time the model reaches the bottom of the image. In contrast, our canvas acts as a 2D structural anchor.

By scattering anchor tokens uniformly across the spatial lattice, InterlacedAR transforms the generation task from an unconstrained sequential prediction into a series of localized conditional refinements. As shown in Figure 6, the transition from Level 3 to Level 6 demonstrates that each parallel refinement step is tightly bounded by the surrounding spatial context from previous levels. This multi-directional constraint prevents the accumulation of sampling errors, ensuring that local details (e.g., texture and edges) are always synthesized in harmony with the pre-defined global structure. We provide statistic analysis to discuss the advantage of InterlacedAR sampling scheme in terms of robustness to error propagation in Section 4.4.

Coarse-to-Fine Trajectory. Despite the initial sparsity, the model demonstrates a clear trajectory of increasing mutual information. The early levels prioritize the global information (the object layout), while the later levels leverage strong local spatial correlations to fill in the fine-grained details. This factorization provides a more robust hierarchy for image synthesis than the rigid raster-scan dependency.

4.3 Ablation Study

4.3.1 Effectiveness of Interlacing Schedule

To study the effect of schedule granularity, we vary the base interlacing block size **BS**. In this ablation, we use a fixed HSO strategy and a GPT-B backbone without AdaLN (Peebles & Xie, 2023). All models are trained for 100 epochs.

Table 3 shows the trade-off between decoding speed and FID. We use raster-scan LlamaGen (Sun et al., 2024) as the sequential baseline. InterlacedAR with $\mathbf{BS} = 1$ is effectively equivalent to raster autoregression, requiring 256 sequential steps and yielding similar performance (FID 6.97 vs. 7.18; 3.08s vs. 3.12s per batch). As **BS** increases, the number of decoding steps decreases substantially. At $\mathbf{BS} = 8$, the model requires only 10 steps and 0.17s per batch, compared with 256 steps and 3.12s for the raster baseline. This acceleration is accompanied by a moderate increase in FID from 7.18 to 7.75.

These results suggest that $\mathbf{BS} = 8$ is a practical default operating point in our setting: it provides roughly an 18× reduction in wall-clock time with only a modest degradation in FID relative to the raster baseline. We therefore use $\mathbf{BS} = 8$ in the main experiments.

4.3.2 Effectiveness of HSO for Input Tokens

We evaluate the impact of different HSO strategies on the input token sequence using a GPT-B backbone without AdaLN (Peebles & Xie, 2023), fixed $\mathbf{BS} = 8$, and 100 training epochs. Results are shown in Table 4.

We compare four strategies: (1) **closest & min idx**, which chooses the nearest known token and breaks ties using the minimum index; (2) **closest & max idx**, which uses the maximum index for tie-breaking; (3)

Table 5: Ablation on class and level awareness. Lower FID is better.

Condition	#Para.	GFLOPs	FID↓
None aware	111M	104.70	6.35
Class aware	153M	148.19	5.93
Level aware	153M	148.19	5.81
Class & Level aware	153M	148.19	5.39

closest & left up, which prioritizes the top-left token under ties; and (4) **unattached & max idx**, which selects the token used least frequently as input.

Spatially local strategies perform substantially better than the **unattached** alternative. In particular, **closest & min idx** achieves the best FID of 7.12. These results support the intuition that preserving local spatial relationships in the serialized input sequence is important for interlaced decoding, while the specific tie-breaking rule has a comparatively smaller effect.

4.3.3 Effectiveness of Class & Level Awareness

We next evaluate four conditioning variants using the InterlacedAR-B backbone. As shown in Table 5, both class-aware and level-aware conditioning improve image quality relative to the unconditioned variant.

Adding only class-aware conditioning reduces FID from 6.35 to 5.93, while adding only level-aware conditioning reduces it to 5.81. The full class-and-level-aware model achieves the best result, with FID 5.39. This pattern suggests that class information and level information play complementary roles: the former provides global semantic context, while the latter informs the model about its position in the interlaced decoding process.

Generalization to Text-to-Image and Open-domain Tasks. While this study validates InterlacedAR on class-conditional ImageNet, the underlying mechanism is fundamentally dataset-agnostic and task-independent. The main contribution—replacing 1D raster scanning with a 2D interlaced schedule—operates on the latent token lattice, which remains the standard representation for modern autoregressive models such as LlamaGen (Sun et al., 2024) and Chameleon (Team, 2024). In text-to-image (T2I) settings, text embeddings are typically injected via cross-attention, a process that is orthogonal to the spatial order of token decoding. Therefore, InterlacedAR can be seamlessly integrated into T2I frameworks as an efficient decoding plug-in. Moreover, because the interlaced schedule exploits universal spatial redundancies inherent in all natural images, we expect the efficiency-quality trade-off to hold across open-domain datasets and higher resolutions, where the benefit of reducing sequential steps becomes even more pronounced.

4.4 Robustness of InterlacedAR Sampling

To provide evidence for robustness of InterlacedAR sampling scheme towards error propagation, and address concerns regarding the sensitivity of the sparse Canvas Stage, we conduct analysis by injecting types of noise into the initial sampling steps. We compare InterlacedAR against the raster-order baseline, LlamaGen (Sun et al., 2024), under several perturbation strategies: (1) **Probability Inversion**: calculating the normalized probability as $1 - P(x)$ to force the model toward unlikely tokens; (2) **Temperature Scaling**: setting temp=2.0 to increase sampling entropy; and (3) **Gaussian Noise**: adding $\mathcal{N}(0, \sigma^2)$ to the logits. For InterlacedAR, noise is injected when sampling tokens in the Canvas Stage (4 tokens for image size $= 256 \times 256$, and 9 tokens for image size $= 384 \times 384$), while for LlamaGen (Sun et al., 2024), it is applied to an equivalent number of initial tokens.

We summarize the results in Figure 7. Notably, under the most severe perturbation of Probability Inversion, InterlacedAR demonstrates significantly better recovery than LlamaGen (Sun et al., 2024). For instance, in the -B configuration at 256×256 resolution, InterlacedAR's FID increases by only 10.7% (from 5.39 to 5.97), whereas LlamaGen (Sun et al., 2024) exhibits a larger FID increase by 14.7% (from 5.46 to 6.32). In the -XL

configuration at 384×384 resolution, InterlacedAR's FID increases by only 8.4% (from 2.60 to 2.82), while LlamaGen's (Sun et al., 2024) FID increases by 12.5% (from 2.62 to 2.95).

This performance gap suggests that while raster-order models suffer from severe sequential drift (an initial sampling error propagates and accumulates along the 1D chain), InterlacedAR exhibits inherent robustness against error propagation. Because the initial tokens serve as a sparse global scaffold rather than a rigid linear starting point, the subsequent parallel refinement stages can effectively confine and relieve early sampling uncertainties. These findings confirm that InterlacedAR's 2D level-wise schedule provides a fundamentally more fault-tolerant topology for image synthesis compared to traditional raster-order approaches, without requiring explicit backtracking or verification modules.

Besides, increasing the sampling temperature to 2.0 or injecting Gaussian noise on logits in the first few steps slightly improves the FID for both models (e.g., from 2.60 to 2.52 for InterlacedAR-XL), suggesting that a degree of stochasticity in the early stages can successfully enhance sample diversity without compromising overall structural coherence.

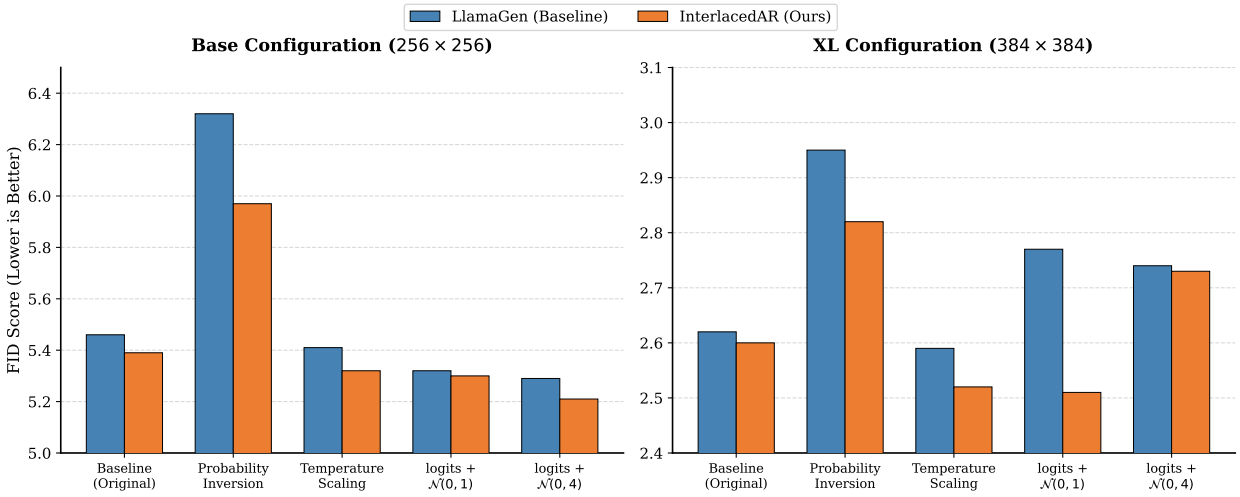


Figure 7: Comparison of FID scores under initial sampling noise. We perturb the early stages of the sampling process to test the robustness towards error propagation.

5 Conclusion and Limitation

We presented InterlacedAR, an autoregressive image generation method based on a coarse-to-fine interlaced decoding schedule inspired by classical graphics interlacing. By partitioning the token grid into interleaved levels and predicting higher levels in parallel conditioned on previously generated ones, InterlacedAR reduces the number of sequential decoding rounds from hundreds to as few as 10 while remaining compatible with a standard decoder-only Transformer backbone and a single-scale tokenizer.

On ImageNet-1K at 256×256, InterlacedAR achieves competitive FID and IS relative to strong autoregressive baselines while substantially reducing wall-clock sampling time. The ablation studies further indicate that the interlacing block size, heuristic spatial ordering, and class/level-aware conditioning each contribute to the final quality-latency trade-off.

At the same time, the present study is limited to class-conditional image generation with discrete visual tokens, and some qualitative analyses are affected by the reconstruction behavior of the underlying VQ decoder at very sparse intermediate grids. Extending the approach to richer conditioning settings, continuous-token representations, higher resolutions, or video generation remains an important direction for future work.

Impact Statement

This work advances autoregressive image generation by reducing inference latency without sacrificing quality or requiring specialized architectures. The primary societal implications are those common to image generation broadly: potential misuse for generating misleading content, alongside beneficial applications in creative tools, accessibility, and design. We do not introduce new capabilities beyond existing generative models; rather, we make existing capabilities faster and more efficient, which could lower computational costs and energy consumption for practitioners deploying such systems.

References

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*, pp. 11315–11325, 2022.

Huiwen Chang, Jack Hessel, Morgan Campbell, Peter Yu, Xudong Zhang, Puneet Dokania, Xiangyu Hu, Alan Yuille, Yanan Zhou, et al. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*, 2023.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1691–1703, 2020.

Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model. *arXiv preprint arXiv:1712.09763*, 2018.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Yefei He, Yuanyu He, Shaoxuan He, Feng Chen, Hong Zhou, Kaipeng Zhang, and Bohan Zhuang. Neighboring autoregressive modeling for efficient visual generation. *arXiv preprint arXiv:2503.10696*, 2025.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Haopeng Li, Jinyue Yang, Guoqi Li, and Huan Wang. Autoregressive image generation with randomized parallel decoding. *arXiv preprint arXiv:2503.10568*, 2025.

Tianhong Li, Huiwen Chang, Shweta Mishra, Zhuowen Tu, Kaiming He, and Han Zhang. Mage: Masked generative encoder to unify representation learning and image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2329–2338, 2022.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024a.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024b.

Jacob Menick, Nal Kalchbrenner, Aaron van den Oord, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. *arXiv preprint arXiv:1812.01608*, 2018.

Ziqi Pang, Tianyuan Zhang, Fujun Luan, Yunze Man, Hao Tan, Kai Zhang, William T Freeman, and Yu-Xiong Wang. Randar: Decoder-only autoregressive visual generation in random orders. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 45–55, 2025.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4055–4064, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Scott Reed, Aaron van den Oord, Nal Kalchbrenner, Sergio Gómez Colmenarejo, Ziyu Wang, Dan Belov, and Nando de Freitas. Parallel multiscale autoregressive density estimation. *arXiv preprint arXiv:1703.03664*, 2017.

Greg Roelofs and Richard Koman. *PNG: The Definitive Guide*. O'Reilly & Associates, Inc., 1999. ISBN 1565925424.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *International Conference on Learning Representations*, 2017.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37: 84839–84865, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6000–6010, 2017. ISBN 9781510860964.

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 Computer Vision and Pattern Recognition Conference*, pp. 12955–12965, 2025.