MIXTOK: ADAPTIVE VISUAL TOKENIZATION WITH GLOBAL 1D TOKENS AND SPARSE 2D REFINEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Efficient visual tokenization requires balancing representation compactness with the preservation of fine-grained image content. Conventional grid-based tokenizers allocate tokens uniformly across spatial locations despite non-uniform visual complexity, whereas compact 1D tokenizers aggregate global information but can struggle to preserve local details. We introduce MixTok, a hybrid visual tokenizer that combines a compact 1D global representation with sparse 2D tokens for local refinement. A content-adaptive router ranks candidate locations by refinement priority and determines the number of 2D tokens for each image under a target average token budget. At the highest-scoring locations, 2D token features replace the spatial features recovered from the 1D representation, and a shared decoder reconstructs the image from the resulting mixed features. On ImageNet 256×256, MixTok achieves highly competitive reconstruction and generation performance, with an rFID of 1.195 and a gFID of 1.95 at respective average budgets of approximately 128 and 131 content tokens per image.

1 INTRODUCTION

Visual tokenizers compress high-dimensional images into compact latent representations that support reconstruction and provide a modeling space for downstream generators (Esser et al., 2021; Yu et al., 2024b; Rombach et al., 2022; Peebles & Xie, 2023). A central challenge is to allocate representational capacity efficiently: visual complexity varies both within and across images, yet compact representations must retain the information needed for faithful reconstruction and generation. This calls for tokenizers that adapt token allocation to image content.

Existing visual tokenizers typically represent an image as either a spatial grid of 2D tokens or a compact 1D latent sequence. Conventional 2D tokenizers preserve an explicit correspondence between tokens and spatial locations, enabling localized representations of structures and textures (Esser et al., 2021). However, their uniform spatial allocation assigns the same token density to visually simple regions, such as smooth sky, as to regions containing intricate objects or textures, potentially wasting representational capacity. In contrast, 1D tokenizers aggregate information across the image into compact sequences without a one-to-one spatial correspondence (Yu et al., 2024b). Although this global aggregation improves compactness, preserving fine-grained local details within a short sequence remains challenging (Wang et al., 2026). These complementary limitations motivate combining a compact global representation with selectively allocated local capacity.

Beyond spatial allocation, most existing tokenizers assign the same number of tokens to all images at a given resolution, regardless of content complexity. This fixed per-image budget can allocate unnecessary capacity to simple images while limiting detail preservation in complex ones (Duggal et al., 2025). An effective content-adaptive allocation strategy should therefore determine both where local tokens offer the greatest benefit and how many tokens to assign to each image, coordinating spatial selection and per-image allocation under a target average token budget.

To this end, we introduce MixTok, which adaptively allocates 2D tokens for local refinement. Built upon off-the-shelf 1D and 2D tokenizers, MixTok first maps a small set of 1D tokens into a spatial feature map that provides a base representation of the image. A content-adaptive router then determines where and how much to refine: a grid scorer ranks spatial locations by refinement priority, while a top-p budget controller determines the number of selected locations from the resulting score distribution. The selected 2D features refine the 1D-derived base grid, and a shared decoder

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.

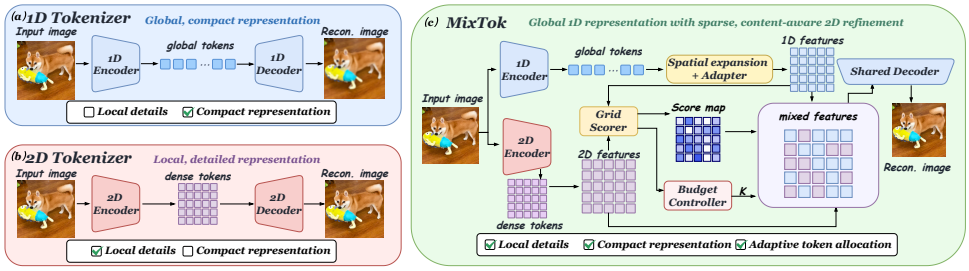

Figure 1: **Comparison of different visual tokenization paradigms.** (a) 1D tokenizer provides a compact global representation but may lose fine-grained local details. (b) 2D tokenizer preserves local details using a dense spatial token grid, but incurs substantial token overhead. (c) Our MixTok combines compact 1D tokens with content-adaptively selected 2D tokens, achieving both representation compactness and local-detail preservation.

reconstructs the image from the combined representation. This design retains the compactness of the global representation while allocating additional capacity to local details that require refinement. For downstream image generation, we employ a two-stage MaskGIT (Chang et al., 2022) pipeline. The first stage generates the 1D base tokens, which guide the selection of refinement locations and the number of 2D tokens to generate. The second stage generates these selected 2D tokens, after which the shared decoder produces the image from the combined representation. On ImageNet at 256×256, MixTok achieves an rFID of 1.195 and a gFID of 1.95 with respective average budgets of approximately 128 and 131 tokens per image, demonstrating high-quality reconstruction and generation with compact representations.

Our main contributions are as follows:

- We introduce MixTok, a hybrid visual tokenizer that combines compact 1D global representations with sparse 2D local refinement through spatial feature replacement and a shared decoder.
- We develop a content-adaptive router that jointly selects refinement locations and determines per-image token counts through learned spatial scoring and top-p selection under a target average token budget.
- We extend MixTok to downstream image generation through a two-stage MaskGIT pipeline and demonstrate its reconstruction and generation performance on ImageNet at 256×256.

2 Related Work

Visual tokenization. Visual tokenizers map images into compact latent representations for reconstruction and downstream generation (Ramesh et al., 2021; Chang et al., 2023; Gu et al., 2022; Tian et al., 2024; Hansen-Estruch et al., 2025). VQ-VAE (van den Oord et al., 2017) learns discrete representations by quantizing encoder features into codebook entries arranged on a spatial grid. Subsequent work improves this framework through reconstruction objectives, model architectures, and quantization schemes (Razavi et al., 2019; Lee et al., 2022; Zhu et al., 2025; Zhao et al., 2025). VQGAN (Esser et al., 2021) introduces perceptual (Zhang et al., 2018) and adversarial objectives (Goodfellow et al., 2014) to improve reconstruction quality, while ViT-VQGAN (Yu et al., 2022) adopts Transformer-based encoders and decoders (Vaswani et al., 2017) with improved codebook learning. MAGVIT-v2 (Yu et al., 2024a) introduces lookup-free quantization based on binary latent codes, enabling larger token vocabularies. These methods improve the quality of spatial representations, but their fixed latent grids impose uniform token density across image regions.

Another line of work replaces spatial grids with compact 1D latent sequences. TiTok (Yu et al., 2024b) uses learnable latent tokens to aggregate image information into a short discrete sequence. TA-TiTok (Kim et al., 2025) extends this approach with text-conditioned decoding and a simplified single-stage training procedure. In the continuous setting, MAETok (Chen et al., 2025) uses masked modeling and auxiliary feature prediction to learn compact, semantically structured representations

for diffusion models. Beyond sequence compactness, SelfBootTok (Chi et al., 2026) separates global and local token groups and learns to predict local details from global tokens, allowing downstream generators to model only the global tokens. However, recovering local content solely from a compact global representation may remain challenging when image-specific details are not adequately encoded in that representation. MixTok addresses this limitation by retaining additional 2D tokens at selected spatial locations to complement the compact 1D representation.

Adaptive visual tokenization. Adaptive visual tokenization seeks to balance representation size and reconstruction quality by varying token counts or spatial allocation. One direction supports flexible per-image token budgets. FlexTok (Bachmann et al., 2025) learns an ordered, coarse-to-fine 1D sequence that supports reconstruction at different chosen lengths. Such flexibility enables variable token budgets but does not, by itself, specify how to select a budget for each image. ALIT (Duggal et al., 2025) adapts representation length through recurrent token allocation, progressively adding latent tokens to improve reconstruction. CAT (Shen et al., 2025) uses caption-based complexity assessment to select an image-dependent compression ratio. A related direction adapts the spatial distribution or support of tokens. DQ-VAE (Huang et al., 2023) varies coding granularity according to regional information density, assigning fewer codes to smooth regions and more to regions containing fine details. GPSToken (ZHANG et al., 2025) represents image regions using parameterized 2D Gaussians with associated texture features, allowing token positions and spatial supports to adapt to image content. However, spatial adaptation alone does not adjust the image-level token budget to the overall information content of each image. In contrast, MixTok jointly determines where to allocate 2D refinement tokens and how many each image requires, assigning additional capacity according to the refinement needs of the 1D representation while maintaining a target average token budget.

3 PRELIMINARY

Visual tokenizers map images into compact latent representations for reconstruction and downstream generation. Given an image $x \in \mathbb{R}^{H\times W\times 3}$, an encoder E transforms it into continuous latent features $Z = E(x)$. A vector quantizer then maps these features into discrete codebook indices and their corresponding quantized features:

$$(T, F) = \mathrm{VQ}_{\mathcal{C}}(Z), \tag{1}$$

where T denotes the discrete token representation modeled by downstream generative models, and F denotes the quantized features used for image reconstruction.

2D grid tokenization. A 2D tokenizer preserves the spatial correspondence between tokens and image locations. Its encoder produces a latent grid $Z_{\mathrm{2D}} = E_{\mathrm{2D}}(x)$, which is quantized into discrete tokens and spatially organized features:

$$(T_{\mathrm{2D}}, F^q_{\mathrm{2D}}) = \mathrm{VQ}_{\mathcal{C}_{\mathrm{2D}}}(Z_{\mathrm{2D}}). \tag{2}$$

Here, T_{2D} denotes the spatial token grid and F^q_{2D} denotes the corresponding quantized feature grid. The pretrained post-quantization projection maps it to the decoder input space:

$$F_{\mathrm{2D}} = P_{\mathrm{2D}}(F^q_{\mathrm{2D}}) \in \mathbb{R}^{h\times w\times d_{\mathrm{2D}}}. \tag{3}$$

The resulting feature grid preserves spatial organization and can be used for local refinement and decoding.

1D sequence tokenization. A 1D tokenizer compresses an image into a compact latent sequence without explicit correspondence between tokens and spatial locations. Following TiTok (Yu et al., 2024b), the encoder produces a sequence of latent features:

$$Z_{\mathrm{1D}} = E_{\mathrm{1D}}(x) \in \mathbb{R}^{L\times d_{\mathrm{1D}}}, \tag{4}$$

which is quantized into discrete tokens and their corresponding features:

$$(T_{\mathrm{1D}}, F_{\mathrm{1D}}) = \mathrm{VQ}_{\mathcal{C}_{\mathrm{1D}}}(Z_{\mathrm{1D}}). \tag{5}$$

Here, T_{1D} denotes the compact token sequence and F_{1D} denotes its corresponding quantized features.

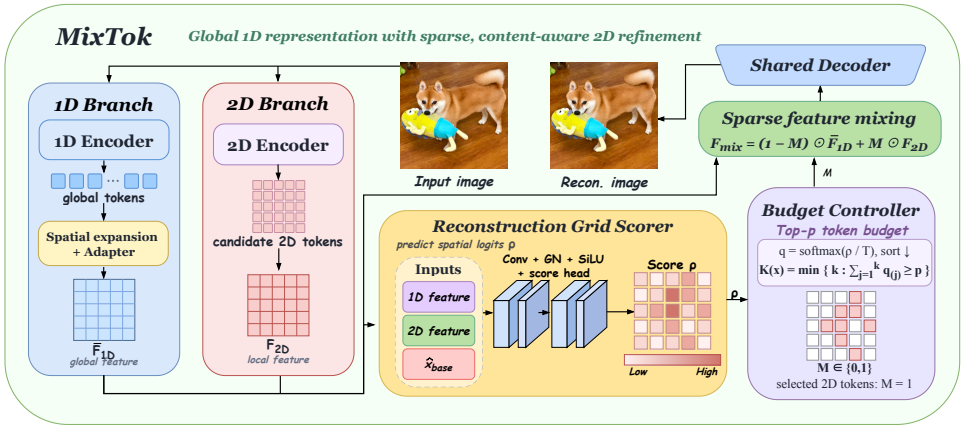

Figure 2: **Overview of MixTok.** Compact 1D tokens provide a base feature grid, while content-adaptive top-p selection allocates 2D refinement tokens across spatial locations. The selected 2D features replace the corresponding base features, and a shared decoder reconstructs the image from the mixed grid.

4 METHOD

4.1 HYBRID VISUAL REPRESENTATION

To combine the compactness of 1D representations with the spatial detail of 2D grids, MixTok supplements a short 1D token sequence with selected 2D tokens for local refinement.

Formally, given an image x, the 1D and 2D branches produce a length-L token sequence T_{1D} and an $h \times w$ token grid T_{2D}, respectively. The 2D grid provides $N = hw$ candidate refinement locations. Let $\mathcal{R} = \{r_1, \ldots, r_K\}$ denote the selected locations, where each $r_k \in \{1, \ldots, N\}$ indexes a spatial position under a fixed flattening order. MixTok retains the complete 1D sequence together with the selected 2D tokens:

$$\mathcal{T}_{\text{MixTok}}(x) = \left(T_{\text{1D}}, \{(r_k, t^{\text{2D}}_{r_k})\}_{k=1}^{K}\right), \qquad B = L + K, \tag{6}$$

where $t^{\text{2D}}_{r_k}$ denotes the 2D token at location r_k and B denotes the number of retained content tokens. The selected locations are stored as side information to recover their spatial positions during decoding and are not counted as content tokens.

To combine the two token types, MixTok maps their quantized features into a shared decoder feature space. Since 1D tokens do not preserve explicit spatial correspondence, their quantized features F_{1D} are first expanded into a spatial feature grid using the spatial query grid S:

$$\widetilde{F}_{\text{1D}} = G_{\text{1D}}(F_{\text{1D}}, S). \tag{7}$$

A learnable feature adapter then projects the expanded features into the decoder input space:

$$\bar{F}_{\text{1D}} = P_{\text{1D}}(\widetilde{F}_{\text{1D}}) \in \mathbb{R}^{h \times w \times d_{\text{2D}}}. \tag{8}$$

In contrast, the decoder-aligned features from the 2D branch, $F_{\text{2D}} \in \mathbb{R}^{h \times w \times d_{\text{2D}}}$, preserve spatial organization and match the decoder input dimension. Therefore, $\bar{F}_{\text{1D}}$ and F_{2D} can be combined within the same feature space.

MixTok constructs a complete decoder input grid by replacing the 1D-derived feature with the corresponding 2D feature at each selected refinement location:

$$F_{\text{mix},i} = \begin{cases} F_{\text{2D},i}, & i \in \mathcal{R}, \\ \bar{F}_{\text{1D},i}, & i \notin \mathcal{R}, \end{cases} \qquad \hat{x} = D(F_{\text{mix}}), \tag{9}$$

where $i \in \{1, \ldots, N\}$ indexes the spatial locations of the feature grid under the same flattening order used above.

4.2 Content-Adaptive Token Allocation

The benefit of local refinement varies across spatial regions and images. When the 1D representation already captures a region well, allocating additional 2D tokens provides limited improvement. For reconstruction, MixTok therefore uses a content-adaptive router comprising a Reconstruction Grid Scorer and a Budget Controller. The former scores candidate locations, and the latter uses these scores to determine the refinement locations and token budget for each image.

Reconstruction grid scoring. The Reconstruction Grid Scorer predicts refinement priorities from the aligned features of both branches and the base reconstruction $\hat{x}_{\text{base}} = D(\bar{F}_{\text{1D}})$:

$$\rho^{\text{rec}}(x) = \Phi^{\text{rec}}_{\text{grid}}\left(\bar{F}_{\text{1D}}, F_{\text{2D}}, \hat{x}_{\text{base}}\right), \tag{10}$$

where $\rho^{\text{rec}}_i(x)$ denotes the refinement score at location i. Sorting these scores in descending order gives $\pi_1(x), \ldots, \pi_N(x)$, where $\pi_k(x)$ denotes the k-th highest-scoring location. The scorer receives auxiliary supervision from local reconstruction-error reductions between the 1D base and full 2D reconstructions, providing a proxy for refinement benefit. It is also optimized through the mixed reconstruction loss using a differentiable selection surrogate as shown in Appendix A.2.2.

Budget control. The Budget Controller uses top-p selection (Holtzman et al., 2020) to determine how many 2D tokens to retain from the score distribution predicted by the Reconstruction Grid Scorer. We first normalize the scores using a softmax with temperature $T > 0$:

$$q_i(x) = \frac{\exp(\rho^{\text{rec}}_i(x)/T)}{\sum_{j=1}^{N} \exp(\rho^{\text{rec}}_j(x)/T)}. \tag{11}$$

Given a threshold $p \in (0, 1]$, we select the shortest prefix of the ranking whose cumulative probability reaches p:

$$K(x) = \min\left\{ k \in \{1, \ldots, N\} \;\middle|\; \sum_{j=1}^{k} q_{\pi_j(x)}(x) \geq p \right\}. \tag{12}$$

The refinement set and total number of retained content tokens are

$$\mathcal{R}(x) = \{\pi_1(x), \ldots, \pi_{K(x)}(x)\}, \qquad B(x) = L + K(x). \tag{13}$$

This rule allocates fewer refinement tokens when the probability mass is concentrated on a small number of locations and more tokens when it is more dispersed. The threshold p controls the trade-off between refinement coverage and token budget. We calibrate p on a subset of training images to match the target average token budget and keep it fixed during evaluation, while allowing token counts to vary across images.

Training objective. Following common practice (Sun et al., 2024; Yu et al., 2022), MixTok is optimized with a weighted combination of reconstruction, perceptual, and adversarial losses:

$$\mathcal{L}_{\text{MixTok}} = \lambda_{\text{rec}} \|x - \hat{x}\|_1 + \lambda_{\text{per}} \mathcal{L}_{\text{LPIPS}}(x, \hat{x}) + \lambda_{\text{adv}} \mathcal{L}_{\text{GAN}}, \tag{14}$$

where x and $\hat{x}$ denote the input and reconstructed images, respectively, and λ_{rec}, λ_{per}, and λ_{adv} control the weights of different loss terms. Detailed loss definitions and weights are provided in Appendix A.2.3.

4.3 Image Generation with MixTok

MixTok generates its hybrid representation in two stages: global 1D tokens establish a base representation, which then guides the selection and generation of sparse 2D refinement tokens (Figure 3). Given a class condition c, a pretrained TiTok MaskGIT (Yu et al., 2024b) generates a length-L sequence $\hat{T}_{\text{1D}}$. Codebook lookup, spatial expansion, and feature alignment convert this sequence into the base grid $\bar{F}_{\text{1D}}$, following Section 4.1.

Refinement location selection. During generation, the 2D features required by the Reconstruction Grid Scorer are unavailable. We therefore introduce a Generation Grid Scorer that predicts refinement priorities from the 1D base grid alone: $\rho^{\text{gen}} = \Phi^{\text{gen}}_{\text{grid}}(\bar{F}_{\text{1D}}) \in \mathbb{R}^N$. The Generation Grid Scorer is trained by distilling the frozen Reconstruction Grid Scorer, learning to predict refinement priorities using only the 1D base features. The Budget Controller applies the top-p rule in Section 4.2 to the outputs of the Generation Grid Scorer, yielding the refinement set $\mathcal{R}$ and its size K.

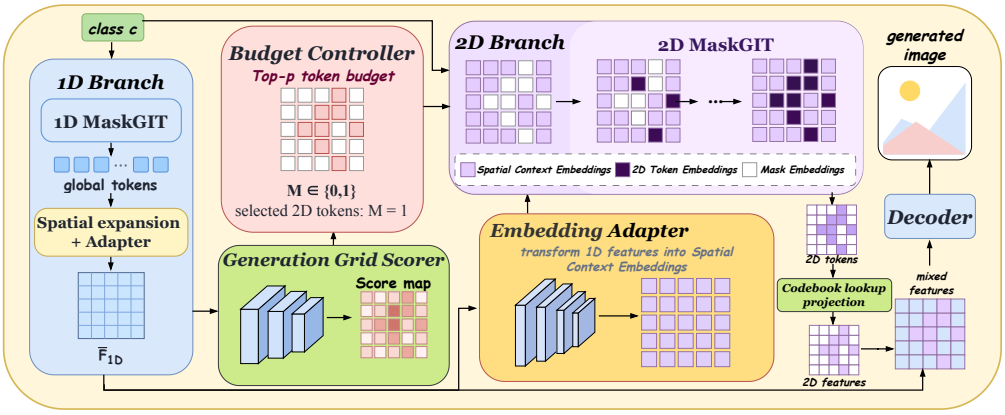

Figure 3: **Two-stage masked generation with MixTok.** A 1D MaskGIT generates global tokens that guide refinement location selection and provide spatial context for a 2D MaskGIT. Features obtained from the generated 2D tokens replace the corresponding base features before shared decoding.

Spatial context embeddings. The base grid lies in the image decoder's feature space, whereas the 2D generator processes inputs in its token embedding space. To use the base representation as a generation condition, we introduce an Embedding Adapter A that maps the base features to spatial context embeddings:

$$H_{\text{ctx}} = A(\bar{F}_{1\text{D}}). \tag{15}$$

The Generation Grid Scorer and Budget Controller determine which locations to generate, while the Embedding Adapter supplies continuous context at the remaining locations. The initial input embeddings are

$$e_i^{(0)} = \begin{cases} E_{\text{tok}}([\text{MASK}]), & i \in \mathcal{R}, \\ H_{\text{ctx},i}, & i \notin \mathcal{R}, \end{cases} \tag{16}$$

where E_{tok} is the 2D generator's token embedding table. Positional embeddings are added before the Transformer.

Sparse 2D generation and decoding. A fine-tuned Halton-MaskGIT (Besnier et al., 2025) generates 2D tokens only in $\mathcal{R}$, conditioned on class c and the context embeddings at unselected positions. Sampling follows the Halton order restricted to $\mathcal{R}$. Each generated token replaces its mask embedding with $E_{\text{tok}}(t_i)$, while the context embeddings at unselected positions remain fixed. The generated 2D tokens are converted to decoder-aligned features and combined with the base grid using Eq. (9) to produce the final image.

Generation training. To train the Embedding Adapter A, we decode the base features into images and re-encode these base reconstructions with the 2D tokenizer. The resulting tokens are looked up in the frozen pretrained Halton-MaskGIT (Besnier et al., 2025) token embedding table to obtain regression targets for A. We then freeze A and fine-tune the Halton generator using the 2D tokens of the original training images as targets, applying cross-entropy only to masked positions within $\mathcal{R}$.

5 Experiment

5.1 Experimental Setup

Dataset and metrics. We train on ImageNet (Deng et al., 2009; Russakovsky et al., 2015) 256×256 and evaluate reconstruction on all 50,000 validation images using rFID (Heusel et al., 2017) and peak signal-to-noise ratio (PSNR). Released tokenizer checkpoints are reevaluated with common preprocessing and metric computation. We report gFID on 50,000 class-balanced samples per sampling seed using the ADM evaluator (Dhariwal & Nichol, 2021).

Table 1: Reconstruction quality and average content-token count. Token counts are rounded to the nearest integer. Baseline configurations are detailed in Appendix A.4. Estimated bitrates are reported in Appendix A.6.

Tokenizer	Avg. tokens	rFID ↓	PSNR (dB) ↑
TiTok S-128 (Yu et al., 2024b)	128	1.7222	17.72
TiTok BL-128 VQ (Kim et al., 2025)	128	1.5063	19.83
One-D-Piece L-256 (Miwa et al., 2025)	128	1.4406	17.71
FlexTok d18-d28 (Bachmann et al., 2025)	128	1.1422	16.76
DetailFlow-16 (Liu et al., 2025)	128	1.1370	19.46
LlamaGen VQ-16 (Sun et al., 2024)	256	2.1872	20.78
MaskGIT-VQGAN (Chang et al., 2022)	256	2.1304	18.15
MixTok (ours)	128	1.1950	19.51

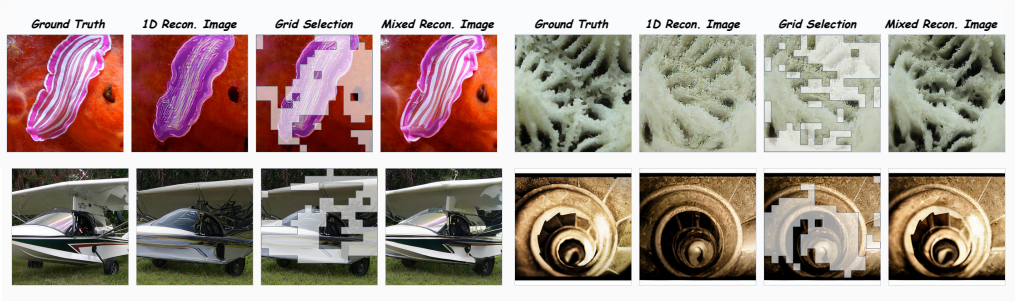

Figure 4: **Qualitative reconstruction results.** Selective 2D refinement recovers fine details and sharpens structures that are blurred or missing in the 1D base reconstructions, while retaining a compact token representation.

Model configurations. Our main configuration combines TiTok-L-32 (Yu et al., 2024b) (32 1D tokens) with LlamaGen VQ-16 (Sun et al., 2024) (a 16×16 2D grid and the shared image decoder). For reconstruction, the fixed variant retains 96 2D tokens and the top-p variant uses approximately 96.1 on average. We also evaluate transferability using ALIT (Duggal et al., 2025) and its VQGAN backbone.

Training strategy. Starting from pretrained tokenizers, we first align the 1D features to the shared decoder space, then train with a fixed refinement budget before fine-tuning with top-p allocation. The 1D encoder and quantizer remain frozen throughout tokenizer training. Detailed training and evaluation settings are provided in Appendix A.

5.2 Image Reconstruction

Reconstruction comparison. As shown in Table 1, MixTok achieves an rFID of 1.1950 and a PSNR of 19.51 dB with approximately 128 content tokens on average. At a comparable token budget, it improves rFID over both TiTok variants and One-D-Piece, while remaining competitive with FlexTok and DetailFlow. Figure 4 shows that 2D refinement improves stripe patterns and curved boundaries over the 1D base reconstructions.

Transferability across tokenizer backbones. For the ALIT–VQGAN combination, we use 64 1D tokens and initialize MixTok-ALIT from the released ALIT-128 checkpoint. We train with a fixed budget of 64 2D tokens for 20 epochs, followed by 5 epochs of top-p training. At evaluation, we retain exactly 64 2D tokens, giving 128 content tokens in total to match the ALIT-128 baseline.

Table 2 shows that, at the same 128-token budget, MixTok-ALIT uses 64 1D and 64 2D tokens and improves rFID from 2.9453 to 2.4720 and PSNR from 18.71 to 18.76 dB over ALIT-128. These

Table 2: Transferability to ALIT–VQGAN at a fixed budget of 128 content tokens.

Tokenizer	Tokens	rFID ↓	PSNR (dB) ↑
ALIT-128	128	2.9453	18.71
MixTok-ALIT	128	**2.4720**	**18.76**

results support the hybrid representation at a matched content-token budget and demonstrate transferability beyond TiTok–LlamaGen.

5.3 Image Generation

Table 3: Class-conditional ImageNet generation. Unmarked gFID values are from our ADM reevaluations, while * denotes an author-reported result. Latency and FLOPs are measured on one RTX 5090 at batch size 1. Configuration and measurement details are provided in Appendix A.8.

Generator	Params	Tokens	Steps	gFID ↓	Latency ↓ (ms)	FLOPs ↓ (T/image)
MaskGIT (Chang et al., 2022)	227M	256	8	6.06*	—	—
AdaGen-MaskGIT-L (Ni et al., 2026)	206M	256	16	2.4883	372.0	3.623
TiTok-S-128 (Yu et al., 2024b)	295M	128	64	1.9994	1018.4	9.949
Halton-MaskGIT-L (Besnier et al., 2025)	480M	576	32	2.5143	591.5	26.690
One-D-Piece-L-256 (Miwa et al., 2025)	177M	256	16	2.4144	350.1	3.526
LPD-L-256 (Zhang et al., 2026)	337M	256	20	2.3863	**215.5**	0.879
LlamaGen-L (Sun et al., 2024)	343M	256	256	3.6803	4951.8	**0.599**
DetailFlow-32 (Liu et al., 2025)	326M	256	39	2.6315	669.9	5.627
MixTok (ours)	321M	130.66	$8+8$	**1.9511**	270.1	1.452

As shown in Table 3, MixTok achieves a gFID of 1.95 with approximately 131 tokens. At a similar token budget, it improves gFID over TiTok-S-128 from 2.00 to 1.95 while reducing sampling rounds from 64 to 16. Compared with DetailFlow-32 at a similar parameter count, MixTok achieves a lower gFID (1.95 vs. 2.63) with approximately half as many tokens and fewer rounds (16 vs. 39). Under our single-image latency protocol, MixTok is $3.77\times$ and $2.48\times$ faster than these TiTok and DetailFlow configurations, respectively. Starting from the same generated 1D tokens, sparse 2D refinement improves gFID from 4.33 to 1.95 with eight additional sampling rounds, demonstrating its contribution to generation quality. Figure 5 illustrates improved local details after 2D refinement during generation. Quantitative results and sampling-round ablations are provided in Table 9 and Appendix A.7.

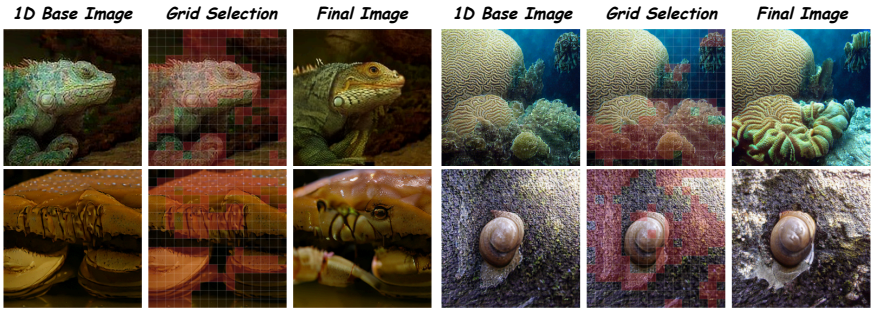

Figure 5: Qualitative class-conditional generation with MixTok.

5.4 Ablation Studies

Grid scorer. To evaluate whether learned scores identify effective refinement locations, we compare the Reconstruction and Generation Grid Scorers with random, uniform, and gradient-entropy rankings. We hold each image's top-p budget $K(x)$ from the Reconstruction Grid Scorer fixed across all strategies, isolating spatial ranking from budget allocation. Uniform ranking follows a predefined spatially dispersed order, while gradient-entropy ranking uses the Shannon entropy of grayscale Sobel gradient magnitudes. As shown in Table 4, the Reconstruction Grid Scorer achieves the best rFID of 1.1950. Using only 1D base features, the Generation Grid Scorer also outperforms all tested heuristics, achieving an rFID of 1.3479 and a PSNR of 18.4776. These results support learned spatial selection and show that distillation preserves useful refinement priorities without access to 2D features.

Table 4: Grid Scorer ablation with identical per-image budgets. Random ranking reports mean $\pm$ standard deviation over three seeds; the other rankings are deterministic.

Selection strategy	rFID ↓	PSNR ↑
Reconstruction Grid Scorer	**1.1950**	**19.5074**
Generation Grid Scorer	1.3479	18.4776
Random	1.5460 ± 0.0093	17.2147 ± 0.0010
Uniform	1.6604	17.1749
Gradient entropy	1.4012	18.0077

Budget controller. To assess whether matching refinement budgets to image content improves reconstruction, we compare top-p allocation with fixed $K = 96$ and shuffled budgets, using the same Reconstruction Grid Scorer. The shuffled control randomly permutes the top-p budgets across images, preserving their distribution and mean while removing their correspondence with image content. Each strategy refines the top-ranked locations for its assigned budget. As shown in Table 5, top-p allocation achieves the best rFID and PSNR at a comparable average budget. Shuffling the budgets degrades both metrics, supporting the benefit of content-dependent allocation beyond merely varying token counts.

Table 5: Budget Controller ablation with a fixed Grid Scorer. The shuffled control preserves the top-p budget histogram and reports mean $\pm$ standard deviation over three permutations.

Allocation strategy	Avg. K	rFID ↓	PSNR ↑
Fixed $K = 96$	96.0000	1.2052	19.4591
Top-p dynamic K	96.1093	**1.1950**	**19.5074**
Shuffled K	96.1093	1.2236 ± 0.0068	19.4181 ± 0.0005

6 Conclusion and Future Work

We presented MixTok, a hybrid visual tokenizer that combines compact 1D global tokens with sparse 2D tokens to preserve local details. Its content-adaptive router determines both where to refine and how many tokens to allocate to each image under a target average budget. The resulting representation supports reconstruction and two-stage masked generation, achieving an rFID of 1.195 and a gFID of 1.95 on ImageNet at average budgets of approximately 128 and 131 content tokens, respectively. Ablations and experiments across tokenizer backbones support the effectiveness of this design. These results demonstrate the value of combining global compression with content-adaptive local refinement for compact, high-quality visual representations.

Limitations and future work. Although MixTok retains and generates fewer discrete content tokens, reconstruction still requires dense 2D encoding, and the second-stage generator operates on the full spatial grid. Evaluation is currently limited to ImageNet at 256×256 and a narrow range of average token budgets. Future work will explore sparse computation and extend adaptive global–local token allocation to broader budgets, higher resolutions, and more diverse visual data.

AI USE STATEMENT

We used generative AI tools, primarily Codex, to assist with method implementation and minor refinements, experimental design and parameter selection, debugging, data processing, experiment execution, and metric aggregation. We also used these tools for translation, manuscript revision, literature search, and reference checking. The main methodological advances were proposed by the authors. The authors reviewed and revised the AI-assisted code, checked experimental logs and metrics, consulted original references, and reviewed the manuscript. The authors take full responsibility for the final content of this work, including its claims, code, and reported results.

ETHICS STATEMENT

This study uses ImageNet and publicly available pretrained models and does not involve new data collection or studies with human participants. Like other image generation systems, the method may inherit biases from training data and could be misused to produce misleading imagery. These risks should be considered when deploying downstream systems.

REPRODUCIBILITY STATEMENT

Our method and experimental details are provided in Sections 4 and 5.1 and Appendix A. Source code and configuration files for reconstruction and generation are available at `https://anonymous.4open.science/r/MixTok-70D6/`. The repository documents the required datasets, pretrained checkpoints, and other external assets.

REFERENCES

Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. FlexTok: Resampling images into 1D token sequences of flexible length. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 2241–2292. PMLR, 13–19 Jul 2025.

Victor Besnier, Mickael Chen, David Hurych, Eduardo Valle, and MATTHIEU CORD. Halton scheduler for masked generative image transformer. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 83027–83047, 2025.

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)*, pp. 11315–11325, June 2022.

Huiwen Chang, Han Zhang, Jarred Barber, Aaron Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Patrick Murphy, William T. Freeman, Michael Rubinstein, Yuanzhen Li, and Dilip Krishnan. Muse: Text-to-image generation via masked generative transformers. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 4055–4075. PMLR, 23–29 Jul 2023.

Hao Chen, Yujin Han, Fangyi Chen, Xiang Li, Yidong Wang, Jindong Wang, Ze Wang, Zicheng Liu, Difan Zou, and Bhiksha Raj. Masked autoencoders are effective tokenizers for diffusion models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 8145–8171. PMLR, 13–19 Jul 2025.

Haozhe Chi, Jinghan Li, Hao Jiang, Wu Sheng, Yi Ma, Jing Wang, and Yadong Mu. Balancing image compression and generation with bootstrapped tokenization, 2026.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021.

Shivam Duggal, Phillip Isola, Antonio Torralba, and William Freeman. Adaptive length image tokenization via recurrent allocation. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 33702–33728, 2025.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12873–12883, June 2021.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10696–10706, June 2022.

Philippe Hansen-Estruch, David Yan, Ching-Yao Chuang, Orr Zohar, Jialiang Wang, Tingbo Hou, Tao Xu, Sriram Vishwanath, Peter Vajda, and Xinlei Chen. Learnings from scaling visual tokenizers for reconstruction and generation. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 22023–22043. PMLR, 13–19 Jul 2025.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Mengqi Huang, Zhendong Mao, Zhuowei Chen, and Yongdong Zhang. Towards accurate image coding: Improved autoregressive image generation with dynamic vector quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22596–22605, June 2023.

Dongwon Kim, Ju He, Qihang Yu, Chenglin Yang, Xiaohui Shen, Suha Kwak, and Liang-Chieh Chen. Democratizing text-to-image masked generative models with compact text-aware one-dimensional tokens. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18442–18452, October 2025.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11523–11532, June 2022.

Yiheng Liu, Liao Qu, Huichao Zhang, Xu Wang, Yi Jiang, Yiming Gao, Hu Ye, Xian Li, Shuai Wang, Daniel K. Du, Fangmin Chen, Zehuan Yuan, and Xinglong Wu. Detailflow: 1d coarse-to-fine autoregressive image generation via next-detail prediction, 2025.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11976–11986, June 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 51772–51783, 2024.

Keita Miwa, Kento Sasaki, Hidehisa Arai, Tsubasa Takahashi, and Yu Yamaguchi. One-d-piece: Image tokenizer meets quality-controllable compression, 2025.

Zanlin Ni, Yulin Wang, Yeguo Hua, Renping Zhou, Jiayi Guo, Jun Song, Bo Zheng, and Gao Huang. Adagen: Learning adaptive policy for image synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(3):2695–2713, 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. Featured Certification.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8821–8831. PMLR, 18–24 Jul 2021.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, December 2015. ISSN 0920-5691. Publisher Copyright: © 2015, Springer Science+Business Media New York.

Junhong Shen, Kushal Tirumala, Michihiro Yasunaga, Ishan Misra, Luke Zettlemoyer, LILI YU, and Chunting Zhou. Cat: Content-adaptive image tokenization. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 712–742. Curran Associates, Inc., 2025.

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.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 84839–84865. Curran Associates, Inc., 2024.

Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularizing generative adversarial networks under limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7921–7931, June 2021.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Hualiang Wang, Siming Fu, Weinan Jia, Yuning Lu, Mu Liu, Jidong Jiang, and Xiaomeng Li. Prompt yourself: Awakening textual semantics in 1d visual tokenizers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14864–14874. Institute of Electrical and Electronics Engineers Inc., June 2026. IEEE/CVF Conference on Computer Vision and Pattern Recognition 2026 (CVPR 2026) ; Conference date: 03-06-2026 Through 07-06-2026.

Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved VQ-GAN. In *International Conference on Learning Representations*, 2022.

Lijun Yu, José Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David Ross, and Lu Jiang. Language model beats diffusion - tokenizer is key to visual generation. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 765–783, 2024a.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 128940–128966. Curran Associates, Inc., 2024b.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Zhengqiang ZHANG, Rongyuan Wu, Lingchen Sun, and Lei Zhang. Gpstoken: Gaussian parameterized spatially-adaptive tokenization for image representation and generation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 11218–11249. Curran Associates, Inc., 2025.

Zhuoyang Zhang, Luke Huang, Chengyue Wu, Shang Yang, Kelly Peng, Yao Lu, and Song Han. Locality-aware parallel decoding for efficient autoregressive image generation. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 1010–1033, 2026.

Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 90844–90868, 2025.

Yongxin Zhu, Bocheng Li, Yifei Xin, Zhihua Xia, and Linli Xu. Addressing representation collapse in vector quantized models with one linear layer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22968–22977, October 2025.

A Appendix

A.1 Implementation Details

The settings below describe the main TiTok–LlamaGen experimental configuration. Spatial expansion G_{1D}, feature alignment P_{1D}, and the generation Embedding Adapter A are separate modules with different output spaces.

Tokenizer modules. We use TiTok-L-32 (Yu et al., 2024b) with a 4,096-entry codebook of 12-dimensional vectors and LlamaGen VQ-16 (Sun et al., 2024) with a 16,384-entry codebook of 8-dimensional vectors. Spatial expansion G_{1D} is initialized from the TiTok decoder backbone and produces a 16×16 grid. The feature adapter P_{1D} uses a 1×1 projection to 256 channels, two residual convolutional blocks, and a final 1×1 projection. Each residual block contains two 3×3 convolutions with GroupNorm and SiLU. The 2D branch's post-quantization projection P_{2D} maps its 8-dimensional codebook vectors to the same 256-channel decoder space.

Grid Scorers. The Reconstruction Grid Scorer projects $\bar{F}_{1D}$, F_{2D}, their difference, and its absolute value through separate 1×1 convolutions. The base reconstruction is average-pooled to the grid resolution and processed by two 3×3 convolutions with an intermediate SiLU. These five outputs are summed with a learned spatial embedding. Three blocks of 3×3 convolution, eight-group GroupNorm, and SiLU precede a 1×1 score head. All hidden features have 128 channels. The Generation Grid Scorer uses the same hidden width, three-block trunk, and score head, with a single projection of $\bar{F}_{1D}$ and a learned spatial embedding as its input.

Generation modules. The Embedding Adapter A adds a convolutional residual to the 256-channel base grid and projects it to 768 channels. The residual branch uses two 3×3 convolutions with 128 hidden channels and SiLU, followed by a zero-initialized 1×1 convolution to 256 channels. A final 1×1 projection produces the spatial context embeddings. The 2D generator is initialized from the pretrained Halton-MaskGIT base model (Besnier et al., 2025), with 12 Transformer layers, 12 attention heads, hidden dimension 768, feed-forward dimension 3,072, and dropout 0.1. It retains the full spatial sequence and one register token; sparse refinement restricts the prediction targets and token updates to $\mathcal{R}$.

A.2 Reconstruction Training Details

A.2.1 Training Schedule

Table 6 summarizes the released three-stage recipe. The 1D encoder and quantizer remain frozen throughout. Alignment pretraining optimizes only G_{1D} and P_{1D}, with the complete 2D tokenizer frozen. Fixed-budget training begins with 0.5 epoch of Reconstruction Grid Scorer optimization, then jointly optimizes spatial expansion, feature alignment, the 2D encoding path, the post-quantization projection, the image decoder, and the scorer. Adversarial training starts at epoch 1.5, with a 0.01-epoch discriminator warm-up. Top-p fine-tuning initializes from the fixed-budget EMA model, retains the discriminator, resets the optimizer, and enables joint and adversarial training from the start.

We use AdamW (Loshchilov & Hutter, 2019) with $(\beta_1, \beta_2) = (0.9, 0.999)$, cosine learning-rate decay, BF16 mixed precision, and gradient clipping at norm 1. The discriminator uses $(0.5, 0.9)$ and a constant learning rate. Weight decay is zero for the 2D branch, scorer, and discriminator; biases and normalization parameters are excluded from weight decay. Tokenizer training uses bicubic resizing of the shorter image side to 256 pixels, followed by a center crop, without random cropping or flipping. TiTok inputs lie in $[0, 1]$, and LlamaGen inputs lie in $[-1, 1]$.

A.2.2 Scorer Supervision and Differentiable Selection

Reconstruction Grid Scorer supervision. Let $\hat{x}_{2D} = D(F_{2D})$ be the reconstruction using the complete 2D feature grid. We form an auxiliary supervision map from the difference between the base and 2D reconstruction errors:

$$g_i = \max\left\{0,\ \mathrm{Avg}_{u \in \Omega_i}\left[\mathrm{MSE}_{\mathrm{ch}}(\hat{x}_{\mathrm{base},u}, x_u) - \mathrm{MSE}_{\mathrm{ch}}(\hat{x}_{2D,u}, x_u)\right]\right\}, \tag{17}$$

Table 6: Reconstruction training settings for the main configuration. Batch sizes are global across eight GPUs. Dashes denote frozen or inactive modules.

Setting	Alignment	Fixed budget	Top-p
Epochs	2.5	20	6
Global batch size	96	192	160
Refinement allocation	None	$K = 96$	$T = 2,\ p = 0.91$
LR: G_{1D}, P_{1D}	5×10^{-5}	10^{-6}	5×10^{-7}
LR: 2D encoder / pre-quantization projection	—	2×10^{-7}	10^{-7}
LR: P_{2D}, image decoder	—	8×10^{-7}	4×10^{-7}
LR: Reconstruction Grid Scorer	—	10^{-5}	5×10^{-6}
LR: discriminator	—	5×10^{-5}	5×10^{-5}
LR warm-up (updates)	1,000	0	0
Minimum LR (non-discriminator)	5×10^{-7}	0	0
Weight decay: G_{1D}, P_{1D}	10^{-4}	10^{-2}	10^{-2}
EMA decay	0.999	0.999	0.999

where Ω_i is the image region pooled into grid cell i, and $\mathrm{MSE}_{\mathrm{ch}}$ averages over color channels. The 37.5% of cells with the largest g_i values receive binary target 1; the remaining cells receive 0. Binary cross-entropy with the predicted logits defines $\mathcal{L}_{\mathrm{score}}$. Targets are detached from the computation graph. This map provides local error-reduction supervision; it is a proxy for the benefit of replacing an individual grid feature. Scorer inputs are also detached, while the mixed reconstruction supplies gradients to the scorer through the selection surrogate below.

Top-p training. Using the probabilities and ordering from Section 4.2, let $c_{j-1} = \sum_{\ell<j} q_{\pi_\ell}$. The forward pass uses the hard prefix mask $m_{\pi_j} = \mathbf{1}[c_{j-1} < p]$. For backpropagation, we use

$$s_{\pi_j} = \mathrm{sigmoid}\left(\frac{p - c_{j-1}}{\tau}\right), \qquad \widetilde{m} = m + s - \mathrm{sg}(s), \tag{18}$$

where sg stops gradients and $\tau = 0.01$. The mask is scattered back to the original spatial order before mixing. Budget and binarization penalties are

$$\mathcal{L}_{\mathrm{budget}} = (\langle s \rangle - 0.375)^2, \qquad \mathcal{L}_{\mathrm{binary}} = \langle s(1-s) \rangle, \tag{19}$$

where the averages cover the local minibatch and spatial positions. Fixed-budget training uses a hard top-96 mask with a sigmoid surrogate of normalized logits, temperature 0.7, and a log-odds bias for ratio 0.375. The budget penalties regularize the soft masks during training; evaluation uses hard selection.

A.2.3 Loss

Alignment pretraining combines image L_1, LPIPS (Zhang et al., 2018), feature L_1 between $\bar{F}_{1D}$ and F_{2D}, and feature-moment matching, with weights 1, 0.5, 0.05, and 0.05, respectively. Moment matching sums the L_1 differences of per-channel spatial means and standard deviations.

For joint fixed-budget and top-p training, the reconstruction objective in Eq. (14) is supplemented with feature and routing losses. Their coefficients are listed in Table 7. DINOv2 (Oquab et al., 2024) ViT-S/14 and CLIP (Radford et al., 2021) ViT-B/32 provide frozen, normalized image features. We use L_1 distances, averaging DINOv2's class-token and patch-token losses. Both backbones receive images resized to 224×224 and normalized using their respective input statistics. The discriminator uses a frozen ConvNeXt-Base (Liu et al., 2022) backbone with trainable convolutional heads. Its feature-matching term compares real and reconstructed features using L_1 distance. Adversarial training uses a hinge discriminator loss and the negative mean fake score for the generator, with LeCam regularization (Tseng et al., 2021) of weight 10^{-3} and EMA decay 0.999.

Table 7: Loss coefficients during joint reconstruction training. The image terms are implemented as $2(\mathcal{L}_1 + 0.3\mathcal{L}_{\text{LPIPS}})$. Adversarial and discriminator feature-matching terms follow the activation schedule in Appendix A.2.1.

Term	Weight	Term	Weight
Image L_1	2.0	Scorer supervision	1.0
LPIPS	0.6	Budget penalty	20.0
DINOv2 feature loss	0.5	Binarization penalty	0.01
CLIP feature loss	0.5	Generator adversarial loss	0.12
Discriminator feature matching	0.5		

A.3 Generation Training and Sampling

A.3.1 Generation Grid Scorer

The frozen reconstruction model supplies teacher logits ρ^{rec}, while the student predicts ρ^{gen} from $\bar{F}_{\text{1D}}$ alone. The distillation loss is

$$\mathcal{L}_{\text{distill}} = D_{\text{KL}}(q^{\text{rec}} \,\|\, q^{\text{gen}}) + \alpha \,\text{SmoothL1}(\rho^{\text{gen}} - \langle \rho^{\text{gen}} \rangle, \rho^{\text{rec}} - \langle \rho^{\text{rec}} \rangle), \tag{20}$$

where each distribution is a temperature-scaled softmax over locations, and logit means are computed separately for each image. We train the scorer for 20,000 updates on eight GPUs with batch size 32 per GPU (256 in total), temperatures of 2 for both distributions, and $\alpha = 0.05$. AdamW uses a constant learning rate of 2×10^{-4}, weight decay 10^{-2}, $(\beta_1, \beta_2) = (0.9, 0.999)$, and gradient clipping at norm 1. The frozen teacher forward pass uses BF16. Distillation supervises the full spatial distribution and does not depend on a top-p cutoff; $p = 0.75$ is used for subsequent location selection.

A.3.2 Embedding Adapter and 2D Generator

Cached training pairs. Generation training uses two cached views of each ImageNet training image: an ADM-style center crop and its horizontal flip. Each view provides paired 1D codes and original-image 2D targets. For adapter training, we also cache tokens obtained by re-encoding the 1D base reconstructions with the frozen 2D tokenizer. A separate cache stores the locations selected by the frozen Generation Grid Scorer with $T = 2$ and $p = 0.75$. The 2D generator batches examples with equal K to avoid padding the selected targets.

Embedding regression. Let t_i^{base} be the re-encoded base-image token and $E_{\text{tok}}^{\text{pre}}$ the fixed pre-trained Halton embedding table. The adapter loss averages over every spatial location:

$$\mathcal{L}_A = \frac{1}{N d \, v_E} \sum_{i=1}^{N} \left\| A(\bar{F}_{\text{1D}})_i - E_{\text{tok}}^{\text{pre}}(t_i^{\text{base}}) \right\|_2^2, \tag{21}$$

where d is the token embedding dimension and v_E is the fixed variance of all entries in the pre-trained embedding table, excluding the mask-token row. The tokenizer and embedding targets remain frozen. Adapter training uses BF16, while base features and the regression loss are computed in FP32.

Masked generator training. After adapter training, A remains frozen and all Halton generator parameters are fine-tuned. With equal probability per example, we mask selected positions using either the official arccos random-mask schedule or a Halton-prefix mask. The latter samples a completed-round index uniformly from 0 to 31 and masks the unrevealed suffix of the selected Halton order. Every example retains at least one masked target. Cross-entropy is averaged only over masked selected positions, using the original-image 2D codes as labels. The class condition is dropped with probability 0.1. Adapter and generator training use AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$ and gradient clipping at norm 1.

The adapter uses a cosine learning-rate schedule over 20,000 updates. The generator warms up for 2,500 updates, holds its learning rate until update 4,000, then decays by a cosine schedule to 2×10^{-6} at update 20,000.

Table 8: Continuous-context generation training settings. Each model is trained on eight GPUs.

Setting	Embedding Adapter	Halton generator
Optimizer updates	20,000	20,000
Global batch size	512	256
Learning rate	5×10^{-4}	2×10^{-5}
Weight decay	0.01	0.03
Warm-up updates	500	2,500
Training precision	BF16	BF16

A.3.3 Sampling Configuration

For the main generation results, the pretrained TiTok MaskGIT uses eight rounds, guidance scale 4.5 with linear guidance scheduling, and randomization temperature 9.5; softmax temperature annealing is disabled. The Generation Grid Scorer selects locations with $T = 2$ and $p = 0.75$. The Halton generator uses eight rounds with an arccos schedule, guidance weight $w = 3.0$, and categorical sampling at temperature 1. Its classifier-free guidance combines logits as $\ell_{\text{cfg}} = (1 + w)\ell_{\text{cond}} - w\ell_{\text{uncond}}$. The mask-token logit is excluded from sampling, and the context embeddings use scale 1. Thus, the final 1D/2D guidance settings are 4.5/3.0, with $8 + 8$ sampling rounds.

A.4 Evaluation Details

Reconstruction. Validation uses the center-crop implementation supplied with LlamaGen and the EMA tokenizer weights. The final allocation parameters are $T = 1$ and $p = 0.994897$, with no additional minimum or maximum token clamp. The threshold p was calibrated on a randomly sampled subset of the ImageNet training set after the 20-epoch fixed-budget stage to match the target average budget of approximately 96 2D tokens. These evaluation settings differ from the top-p fine-tuning values $T = 2$ and $p = 0.91$ in Table 6. For rFID, original and reconstructed images are clipped to $[0, 1]$, scaled by 255, rounded to uint8, and evaluated with 2,048-dimensional Inception features through TorchMetrics. PSNR is computed for each image in $[0, 1]$ before averaging over the validation set; it is not computed from a single dataset-averaged MSE.

Reconstruction baseline configurations. For Table 1, TiTok BL-128 uses the updated single-stage VQ training recipe. One-D-Piece uses the first 128 tokens of the released L-256 checkpoint. FlexTok uses discrete FSQ (Mentzer et al., 2024) tokens with a flow decoder, evaluated with 25 decoding steps, guidance scale 7.5, adaptive projected guidance, and seed 0.

Generation. The ADM evaluator (Dhariwal & Nichol, 2021) uses the official `VIRTUAL_imagenet256_labeled.npz` reference statistics. For the main results, we use sampling seeds 1–4, each generating 50,000 images with 50 examples per class. We report the mean and sample standard deviation of the four independently computed gFID values (Table 9). These repeats measure sampling variability for a fixed checkpoint and sampling configuration. The reported token count averages $L + K(x)$ over all four runs; position indices, class conditioning, and internal context embeddings are excluded from the content-token count.

Table 9: Generation quality before and after sparse 2D refinement across sampling seeds. Each seed uses 50,000 images. The final row reports the mean ± sample standard deviation across four runs. Token statistics refer to the final representation.

Seed	1D base gFID ↓	Full refinement gFID ↓	Mean K	Mean total tokens
1	4.2758	1.9518	98.66	130.66
2	4.3697	1.9509	98.53	130.53
3	4.3611	1.9597	98.75	130.75
4	4.3318	1.9420	98.69	130.69
Mean ± SD	4.3346 ± 0.0424	1.9511 ± 0.0072	98.66 ± 0.09	130.66 ± 0.09

The 1D base uses 32 generated tokens and eight sampling rounds, decoded through MixTok's spatial expansion, feature alignment, and shared decoder. Starting from the same generated 1D tokens, sparse 2D refinement improves mean gFID from 4.3346 to 1.9511 with eight additional sampling rounds and approximately 131 total content tokens.

A.5 Distribution of 2D Token Counts

Figure 6 shows image-dependent refinement budgets in both reconstruction and generation. For generation seeds 2–4, mean K ranges from 98.53 to 98.75, with similar distributions across seeds.

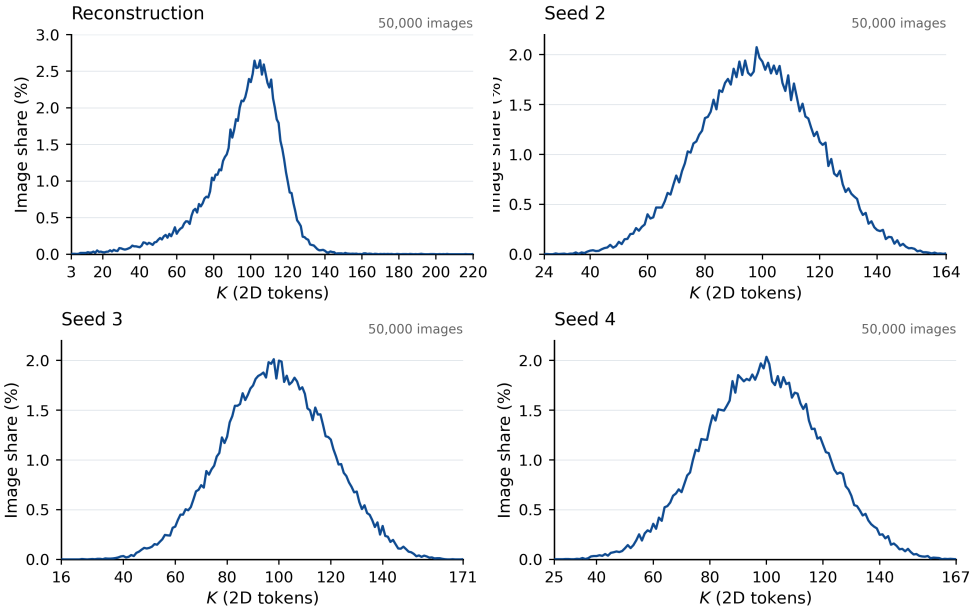

Figure 6: Distribution of selected 2D token counts K for reconstruction on the ImageNet validation set and generation with sampling seeds 2–4, with 50,000 images per panel. The vertical axis shows the percentage of images assigned each integer K. The fixed 32 1D tokens are excluded. Each panel spans the smallest to largest observed K, retaining zero-count bins within this range.

A.6 BITRATE ACCOUNTING

Table 10 supplements the reconstruction results in Table 1 with estimated bitrates under fixed-width coding, using the same checkpoints and decoding settings. For MixTok, 12 / 14 denotes the bit widths of the 1D / 2D token indices. Each token index from a vocabulary of size V is assigned $\lceil \log_2 V \rceil$ bits. The baseline configurations use fixed evaluation lengths and predefined token orders, requiring no per-image position or length fields. FlexTok's FSQ levels $[8, 8, 8, 5, 5, 5]$ yield 64,000 possible codes, requiring 16 bits per packed token index.

For MixTok, we account for the selected locations using a 256-bit binary mask over the 16×16 grid, with the selected 2D token indices stored in raster order. The mask determines both the positions and $K(x)$, so no separate length field is needed. With 12 bits per 1D token and 14 bits per 2D token, the average bitrate is

$$\overline{R}_{\text{MixTok}} = \frac{32 \times 12 + 14\,\mathbb{E}[K(x)] + 256}{256 \times 256}. \tag{22}$$

The measured mean $\mathbb{E}[K(x)] = 96.1093$ gives 0.03030 bpp, including 0.00390625 bpp for spatial side information. These are calculated representation payloads; no entropy coding is assumed, and file headers, byte padding, and shared model weights are excluded. Equal content-token counts can therefore correspond to different bitrates.

Table 10: Reconstruction quality and estimated fixed-width bitrate on ImageNet 256×256. Total bpp includes spatial side information. Token counts are rounded for display; bitrate uses unrounded counts.

Tokenizer	Avg. content tokens	Bits per token index	Total bpp ↓	rFID ↓	PSNR (dB) ↑
TiTok S-128 (Yu et al., 2024b)	128	12	0.02344	1.7222	17.72
TiTok BL-128 VQ (Kim et al., 2025)	128	13	0.02539	1.5063	19.83
One-D-Piece L-256 (Miwa et al., 2025)	128	12	0.02344	1.4406	17.71
FlexTok d18-d28 (Bachmann et al., 2025)	128	16	0.03125	1.1422	16.76
DetailFlow-16 (Liu et al., 2025)	128	13	0.02539	1.1370	19.46
LlamaGen VQ-16 (Sun et al., 2024)	256	14	0.05469	2.1872	20.78
MaskGIT-VQGAN (Chang et al., 2022)	256	10	0.03906	2.1304	18.15
MixTok (ours)	128.11	12 / 14	0.03030	1.1950	19.51

A.7 GENERATION SAMPLING-ROUND ABLATION

Using the main generation checkpoint, we vary only the 2D sampling rounds. The 1D stage uses eight rounds, and all other sampling settings are held fixed. Each setting generates 50,000 images with sampling seed 4. These single-seed results are reported separately from the four-seed main results in Table 9.

Table 11: Effect of 2D sampling rounds with all other settings fixed. All settings use sampling seed 4.

2D rounds	Total rounds	gFID ↓
4	12	2.0314
8	16	1.9420
16	24	1.9204
32	40	1.9191
48	56	1.9242
64	72	1.9355

Increasing the 2D stage from 4 to 8 rounds substantially improves gFID, while further increases yield diminishing returns, with performance saturating around 24–40 total rounds. We therefore use 8+8 rounds as the main efficiency-oriented configuration.

A.8 GENERATION EFFICIENCY

Generation table configurations. In Table 3, steps count sampling rounds; DetailFlow-32 uses eight causal plus 31 group-refinement passes. The starred MaskGIT result is officially reported for the released checkpoint and is not reevaluated with our ADM protocol. Halton-MaskGIT-L uses its native 384×384 output and a 24×24 token grid; the other models output 256×256. MixTok reports mean gFID over seeds 1–4 (sample SD: 0.0072), with 50,000 images per seed, as detailed in Appendix A.4.

Latency. We measure each model on one RTX 5090 with batch size 1 and ImageNet class 207. After 10 warm-up runs, we time 50 runs with sampling seeds 4–53 and report the median. Sampling-only and end-to-end runs are paired by seed with alternating order; Table 3 reports end-to-end latency. We synchronize CUDA before and after each wall-clock measurement. Timing covers class-conditioned generation through final RGB decoding, with outputs remaining on the GPU; model loading, file I/O, and FID evaluation are excluded. For MixTok, this includes both generators, spatial expansion and feature alignment, the Generation Grid Scorer, and the Embedding Adapter.

Implementation settings. We use PyTorch 2.10.0 with CUDA 12.8, four CPU threads per process, TF32 enabled, and cuDNN benchmarking. We retain each implementation's attention kernels, caches, guidance, and sampling loops, without `torch.compile` or CUDA Graphs. BF16 autocast is enabled while preserving explicit precision settings: MixTok's latent decoder, scorer, and RGB decoder use FP32, and AdaGen's backbone uses internal FP16 autocast. All models are benchmarked sequentially on the same RTX 5090, each in a separate process, on a shared host with unlocked GPU clocks. These measurements describe the tested implementations and workload; gFID values come from the separate evaluations described in Appendix A.4.

FLOPs and parameters. We count one complete class-207, seed-4 inference with PyTorch's `FlopCounterMode`, counting a multiply-add as two FLOPs. The counts include matrix multiplications, convolutions, supported attention operations, all sampling rounds, classifier-free guidance, and final image decoding. Elementwise operations, sorting, sampling, and data movement are excluded from FLOPs but included in latency. FLOPs are reported per image and are not averaged over the 50,000-image quality evaluations. Parameter counts for measured models include all parameters in the instantiated generators and auxiliary generation modules, excluding tokenizers. AdaGen includes its policy network; MixTok includes both generators, its scorer, and adapter. TiTok-S-128 and One-D-Piece-L-256 contain 294.991M and 177.374M parameters under this counting convention, respectively. Halton-L uses its native 384×384 output and is not a matched-resolution compute comparison.