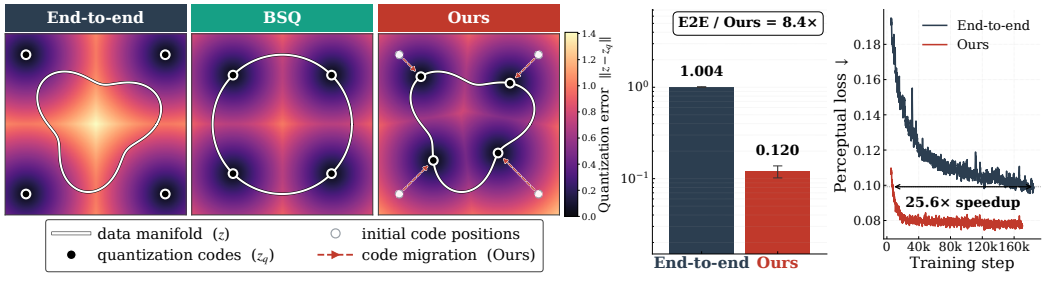
VISUAL TOKENIZATION WITH LATENT ANCHORED BINARY QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete visual tokenizers are bottlenecked by both codebook scalability and quantization error. While lookup-free and finite scalar quantization improve scalability, they forgo the adaptive codebook updates used in vector quantization to reduce quantization error. This mismatch leads to degraded reconstruction and slower convergence in training. To address this, we revisit the two-latent variational formulation that interprets codebook learning as continuous-latent recovery within the ELBO. This interpretation motivates a learned recovery map for binary codes, however direct realization of this objective leads to collapsed representation when the map and its target latents are jointly optimized. We therefore introduce latent-anchored binary quantization (LABQ), a tokenizer that learns binary codes and a recovery map against the fixed representations of a pretrained VAE encoder while fine-tuning its decoder. LABQ trains in 30 epochs, using approximately $6.7\times$ fewer tokenizer-training image exposures than BSQ based on the reported epoch budgets. Without adversarial supervision, some blurring and color-shift artifacts remain. We systematically compare PatchGAN, DINO, and pretrained VAE encoders as discriminators, finding that the SD3 VAE encoder, already attuned to the target latent geometry, yields the lowest rFID under the same ten-epoch adversarial fine-tuning schedule. With this design, LABQ achieves competitive results in discrete image reconstruction on ImageNet (rFID 0.29, LPIPS 0.038) and COCO (rFID 2.84, LPIPS 0.037), approaching its continuous VAE reference.



(a) (b) (c)

Figure 1: **Quantization error across realizations.** (a) Vanilla quantization leaves the error unbounded, BSQ bounds it by projecting the latent onto the hypersphere, and ours minimizes it by learning the recovery from codes back to the frozen anchored manifold. (b) Anchoring reduces normalized quantization error by $8.4\times$. (c) It takes $25.6\times$ fewer training steps to reach the end-to-end baseline's final training perceptual loss. The separate BSQ comparison concerns training image exposures, not wall-clock speed. Metrics and protocols are defined in Appendix B.

1 INTRODUCTION

Discrete tokenization has become a foundational ingredient of visual generative pipelines (Esser et al., 2021; Chang et al., 2022; Weber et al., 2024). The quantization bottleneck at the core of these tokenizers produces a heavily compressed representation that serves two purposes. First, the compression discards the high-frequency content of pixel space and high-dimensional continuous latents

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.

alike, content that carries little perceptual value yet dominates the cost of training a downstream generative model. Second, the resulting indices over a finite vocabulary cast visual generation as token prediction, allowing visual models to reuse the architectures, scaling recipes, and inference optimizations of large language models for scalable training and efficient sampling (Sun et al., 2024).

The cost of this compression is information loss at the quantization step, and two bottlenecks govern how small this loss can be made. The first is codebook scalability. Vector-quantized tokenizers (Van Den Oord et al., 2017) collapse and underutilize their codes as the vocabulary grows (Yu et al., 2023b), a failure that lookup-free quantizers (LFQ) (Yu et al., 2023b; Zhao et al., 2024; Mentzer et al., 2023; Weber et al., 2024) remove by construction through binarization, with vocabularies exponential in the number of bits. The second is the quantization error, the gap between the continuous value the code supplies to the decoder and the pre-quantization latent it replaces. This error also increase gradient bias, since the straight-through estimator approximates the quantizer's Jacobian by the identity. Crucially, binarization buys its exponential vocabulary by increasing the estimator bias through harsher quantizations, so the field has escaped the first bottleneck by amplifying the second.

Here, we focus on this underexplored bottleneck. Discrete visual tokenizers typically encode images into continuous representations before quantizing them into discrete codes. To understand the error introduced by this step, we revisit the variational formulation of discrete VAEs, treating the continuous representation and its code as two latent variables. The discrete-root ELBO isolates a conditional KL that measures how well the continuous representation can be recovered from the code. Under Gaussian conditionals with a fixed variance, this term is proportional to squared quantization error. Interestingly, the original VQ-VAE arrives at the same squared-error alignment through its commitment and codebook losses, despite not treating the continuous representation as a separate latent variable in the variational formulation (Van Den Oord et al., 2017). Codebook-free methods, however, keep the code vectors fixed and control the error through commitment losses or structural constraints such as BSQ's spherical normalization (Yu et al., 2023b; Zhao et al., 2024). This shifts the burden of alignment onto the continuous representation, allowing the recovery error to decrease as the target becomes easier to match, potentially at the cost of fine detail needed for reconstruction.

A naive solution is to realize the recovery term through a learned map from the codes to the continuous representation. However, we show that jointly optimizing the map and its target leads to latent collapse when the recovery term is strongly weighted. Reducing its weight leaves the map chasing a moving target, which compromises reconstruction quality. Motivated by these observations and the recovery term in our two-latent ELBO, we propose **Latent-Anchored Binary Quantization (LABQ)**, a codebook-free tokenizer trained within the frozen bottleneck of a pretrained VAE. The tokenizer learns to compress and reconstruct fixed VAE latents from the codes, supervised in latent and image space as we fine-tune the VAE decoder to reconstruct images from the recovered latents.

Finally, we complement this main result with an additional investigation of the adversarial pipeline. Specifically, we examine the role of the discriminator used in a subsequent perceptual fine-tuning stage, and find that a pretrained continuous VAE encoder, naturally aligned to the latent geometry the tokenizer is trying to preserve, achieves lower rFID than the standard PatchGAN (Isola et al., 2017) and more recently adopted DINO (Sauer et al., 2023; Wang et al., 2024b) discriminators under the same ten-epoch adversarial schedule.

Through extensive experiments on ImageNet and MS-COCO, we show that our tokenizer surpasses BSQ-36 on rFID, LPIPS, and SSIM at matched bits per pixel (bpp), using approximately $6.7\times$ fewer tokenizer-training image exposures based on the reported epoch budgets. At 0.562 bpp it reaches an rFID of 0.29 against the 0.20 of the original SD3 latent (4 bpp) it quantizes. This suggests that most of the information the continuous VAE latent carries for reconstruction can be retained at a bitrate far below its nominal 4 bpp.

We now summarise our contributions. **i) Variational analysis of latent recovery.** We derive a two-latent ELBO whose recovery term exposes how quantization error can decrease through either improved recovery or changes to the continuous target, motivating LABQ's use of a fixed pretrained representation. **ii) Latent-anchored binary quantization (LABQ):** We introduce a discrete tokenizer that learns binary codes and a recovery map against the latents of a frozen pretrained VAE encoder, avoiding representation collapse and improving reconstruction over the trainable-encoder ablation without adversarial loss. **iii) VAE-encoder discriminator:** A pretrained VAE encoder

achieves lower rFID than PatchGAN and DINO discriminators under the same ten-epoch adversarial fine-tuning schedule.

2 Related Works

Discrete tokenization begins with VQ-VAE (Van Den Oord et al., 2017), and VQGAN (Esser et al., 2021) established the adversarially trained tokenizer as the primary driver of two-stage generation, adopted across autoregressive, masked, and next-scale decoders (Chang et al., 2022; Yu et al., 2023a; Tian et al., 2024) and across backbone changes (Yu et al., 2021; Cao et al., 2023). The recipe trains both sides of the quantization interface, the encoder toward its codes and the codes toward the encoder, and its documented failures track this chase. Entries that lose their assignments receive no gradient while the encoder drifts away, producing the collapse and underutilization that later work patches with stochastic assignment (Zhang et al., 2023; Wang et al., 2024a) and with codebooks seeded from frozen CLIP features (Zhu et al., 2024; Radford et al., 2021). VQGAN-LC is the sharpest evidence, restoring near-full utilization at 100k codes precisely by pinning one side of the chase. In our framework these are damping mechanisms for a non-stationary target (Section 3.1).

Lookup-free quantization removes the codebook instead (Yu et al., 2023b; Mentzer et al., 2023), binarizing the latent so the vocabulary grows exponentially, and the family now spans MaskBit (Weber et al., 2024), Open-MAGVIT2 (Luo et al., 2024), group-wise WeTok (Zhuang et al., 2025), and the lattice codes of Λ_{24}-SQ (Zhao et al., 2026), with BSQ (Zhao et al., 2024) bounding the per-dimension gap through hypersphere projection. These designs fix the recovery at the code lattice, so the quantization error is bounded or entropy-shaped but never minimized through recovery, and reconstruction gains have come from more bits, data, and capacity while the error remains the untreated variable.

A third line anchors tokenization to pretrained continuous representations. TokenBridge (Wang et al., 2025) quantizes a frozen VAE latent post hoc on a per-channel grid, learning neither assignment nor recovery and thus confined to near-lossless rates. CODA (Liu et al., 2025) learns residual quantization over a continuous VAE but LoRA-adapts the backbone, so its codes describe a co-adapted space rather than the original latent, whereas LABQ learns binary codes and a deep recovery map against fixed VAE latents. Our latent alignment likewise differs from semantic alignment to vision encoders for generation (Yao et al., 2025), which shapes what the latent represents rather than recovering it from its code.

3 Method

This section presents our tokenizer in two parts. We first formalize tokenization as a two-latent-variable generative model and derive its ELBO, showing that the discrete-root factorization makes continuous-latent recovery a canonical training objective whose naive end-to-end realization fails (Appendix D). We then describe our tokenizer architecture and training procedure, which realize this objective stably by aligning a learned discrete bottleneck to the latents of a frozen pretrained continuous VAE.

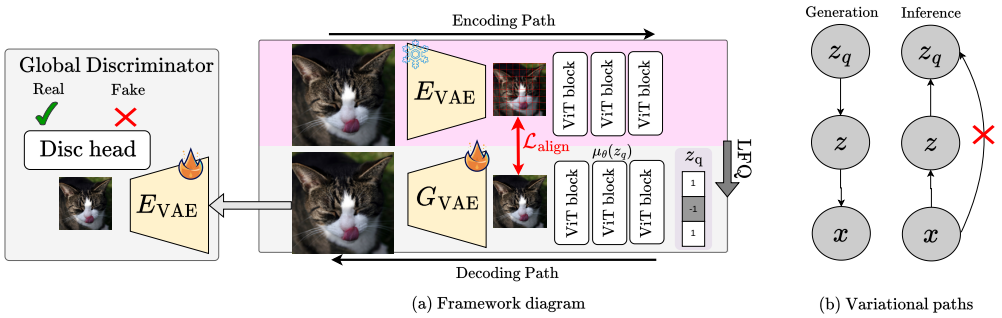
3.1 Two-state Evidence Lower Bound

Two-latent generative model. We formalize tokenization as a generative model with two latent variables, a continuous latent $z \in \mathbb{R}^d$ encoding the observation $x \in \mathcal{X}$ and a discrete code $z_q \in \mathcal{Z}_q$ at the bottleneck. With joint $p_\theta(x, z, z_q)$ and approximate posterior $q_\phi(z, z_q \mid x)$, the log-evidence admits the variational lower bound

$$\log p_\theta(x) \;\geq\; \mathcal{L}(\theta, \phi; x) \;:=\; \mathbb{E}_{q_\phi(z, z_q \mid x)}[\log p_\theta(x, z, z_q) - \log q_\phi(z, z_q \mid x)], \tag{1}$$

whose derivation follows the single-latent case.

Discrete-root ELBO. The latent pair admits two hierarchies, a continuous-root form in which the latent generates the code and a discrete-root form in which the code generates the latent. Only the latter matches the operation of a discrete tokenizer, where the code alone persists and the continuous representation must be regenerated from it, so we adopt the discrete-root factorization



(a) Framework diagram (b) Variational paths

Figure 2: **Anchored tokenizer framework.** (a) The frozen encoder E_{VAE} maps the image to the latent z, which the ViT tokenizer binarizes into the code z_q via LFQ. A symmetric ViT decoder recovers $\mu_\theta(z_q)$ under $\mathcal{L}_{\text{align}}$ against the frozen latent, and the fine-tuned G_{VAE} decodes it to pixels, refined by a discriminator built on a separate trainable copy of the VAE encoder. Snowflake and flame icons mark frozen and trainable modules. (b) Generation and inference paths of the two-latent model, where the crossed edge marks the code as conditionally independent of the image given the latent.

$p_\theta(x \mid z, z_q)\, p_\theta(z \mid z_q)\, p(z_q)$ with posterior $q_\phi(z_q \mid x)\, q_\phi(z \mid z_q, x)$, deferring the shared derivation and the continuous-root form to Appendix A. The bound becomes

$$\begin{aligned}\mathcal{L}_{\text{disc}}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z_q|x)\, q_\phi(z|z_q,x)}[\log p_\theta(x \mid z, z_q)] - \mathrm{KL}(q_\phi(z_q \mid x) \,\|\, p(z_q)) \\ - \mathbb{E}_{q_\phi(z_q|x)}[\mathrm{KL}(q_\phi(z \mid z_q, x) \,\|\, p_\theta(z \mid z_q))]\,. \end{aligned} \tag{2}$$

The first term is reconstruction, and the second is constant for our deterministic quantizer under the uniform code prior (Appendix A). The third is the conditional KL between the target $q_\phi(z \mid z_q, x)$ and the recovery $p_\theta(z \mid z_q)$, and under Gaussian conditionals with shared variance it reduces to a squared distance in z-space, the quantization error of Section 1. This interpretation provides a common basis for comparing how existing quantizers define the target and its recovery, motivating our choice of a fixed pretrained target and a learned recovery map.

Realizations of the KL. Under Gaussian conditionals with target mean $\bar{z}(x)$ and recovery mean $\mu(z_q)$, the conditional KL in equation 2 reduces to $\|\bar{z}(x) - \mu(z_q)\|^2$ up to constants. VQ (Van Den Oord et al., 2017) sets $\bar{z} = E(x)$ and $\mu(z_q) = e_{z_q}$, both trained, and its stop-gradient splits this one term into the commitment and codebook losses, $\beta\|E(x) - \mathrm{sg}(e_{z_q})\|^2 + \|\mathrm{sg}(E(x)) - e_{z_q}\|^2$. LFQ (Yu et al., 2023b) fixes the recovery at the code, $\mu(z_q) = z_q$, leaving the one-sided pull $\beta\|E(x) - \mathrm{sg}(z_q)\|^2$. BSQ (Zhao et al., 2024) keeps $\mu(z_q) = z_q/\sqrt{D}$ fixed and instead constrains the target, $\bar{z} = E(x)/\|E(x)\|$, so the same gap is bounded by construction. All three either train the target or bend it onto a fixed lattice, and none pairs a stationary, informative target with a recovery map free to meet the data, the configuration we construct next.

Anchored alignment with a frozen continuous VAE. We construct the remaining configuration. The target is fixed at $\bar{z}(x) = E_{\text{VAE}}(x)$, the latent of a frozen pretrained continuous VAE encoder, stationary because no gradient reaches it and informative because it is the optimum of the continuous reconstruction objective. The recovery is a learnable deep map $\mu_\theta : \mathcal{Z}_q \to \mathbb{R}^d$ with parameters shared across all codes, free to meet the data where a codebook entry or the code itself is not. Under these choices the reduction instantiates as

$$\mathrm{KL}(q_\phi(z \mid z_q, x) \,\|\, p_\theta(z \mid z_q)) = \frac{1}{2\sigma^2}\,\|\bar{z}(x) - \mu_\theta(z_q)\|^2 + \text{const}, \tag{3}$$

the quantization error against a fixed source, whose minimization is the stable, classical problem of fitting a quantizer to a stationary distribution. Section 3.2 realizes μ_θ and shows that reconstruction through the decoder pretrained on this latent space enforces the objective implicitly, with the explicit term retained as a guardrail (Table 4). Appendix D confirms the construction is necessary. Realizing this with a trainable target instead leads to destabilized training and code collapse at large weights.

Table 1: Reconstruction at 256×256 on ImageNet and COCO. Best discrete results are in **bold**, second-best are underlined; the SD3 continuous reference is blue. Rates are nominal bpp (Appendix B). ‡ CODA counts all ten residual scales: 680 MAR or 2,240 FLUX tokens per image.

bpp	Method	Dim	Size	Tokens	ImageNet				COCO			
					LPIPS↓	rFID↓	PSNR↑	SSIM↑	LPIPS↓	rFID↓	PSNR↑	SSIM↑
0.039	VQGAN Esser et al. (2021)	256	1024	256	-	8.30	19.51	0.61	-	16.95	19.08	0.61
0.055	VQGAN Esser et al. (2021)	256	16384	256	-	4.99	20.00	0.63	-	12.29	19.57	0.63
0.039	MaskGIT Chang et al. (2022)	256	1024	256	-	2.28	-	-	-	-	-	-
0.125	VQGAN Esser et al. (2021)	4	256	1024	-	1.44	22.63	0.74	-	6.58	22.29	0.74
0.219	VQGAN Esser et al. (2021)	4	16384	1024	-	1.19	23.38	0.76	-	5.89	23.08	0.77
0.203	ViT-VQGAN Yu et al. (2021)	32	8192	1024	-	1.28	-	-	-	-	-	-
0.219	LlamaGen Sun et al. (2024)	8	16384	1024	-	0.59	24.45	0.81	-	4.19	24.20	0.82
0.070	Flowmo-Lo Sargent et al. (2025)	18	2^{18}	256	0.113	0.95	22.07	0.65	-	-	-	-
0.219	Flowmo-Hi Sargent et al. (2025)	14	2^{14}	1024	0.073	0.56	24.93	0.79	-	-	-	-
0.281	BSQ-18 Zhao et al. (2024)	18	2^{18}	1024	0.076	1.14	25.36	0.76	0.074	5.81	25.08	0.77
0.5625	BSQ-36 Zhao et al. (2024)	36	2^{36}	1024	0.043	0.41	**27.88**	0.84	0.041	3.42	**27.64**	0.85
0.145	CODA-MAR‡ Liu et al. (2025)	16	16384	680	-	1.43	22.0	0.59	-	-	-	-
0.166	CODA-MAR‡ Liu et al. (2025)	16	65536	680	-	1.34	22.2	0.60	-	-	-	-
0.547	CODA-FLUX‡ Liu et al. (2025)	16	65536	2240	-	0.43	25.9	0.77	-	-	-	-
0.275	Λ_{24}-SQ Zhao et al. (2026)	24	196560	1024	0.062	0.83	26.37	0.79	0.063	5.15	26.00	0.80
0.070	**Ours-S**	18	2^{18}	256	0.212	1.13	23.19	0.63	0.231	6.44	21.84	0.73
0.219	**Ours-M**	14	2^{14}	1024	0.073	0.83	24.05	0.75	0.073	5.09	23.98	0.75
0.5625	**Ours-L**	36	2^{36}	1024	**0.038**	**0.29**	27.31	**0.85**	**0.037**	**2.84**	27.55	**0.86**
4	SD3-VAE Esser et al. (2024)	16	-	1024	0.021	0.20	29.09	0.89	0.019	1.67	29.52	0.89
4	WAN-VAE Wan et al. (2025)	16	-	1024	0.032	1.03	31.42	0.89	0.027	3.49	31.24	0.90
1	SD-VAE Rombach et al. (2022)	4	-	1024	0.098	1.35	21.99	0.63	0.099	5.94	21.68	0.64
1	SDXL-VAE Podell et al. (2023)	4	-	1024	0.067	0.72	25.38	0.73	-	4.07	25.76	0.85

3.2 TOKENIZER DESIGN AND OBJECTIVE

Design. We instantiate the configuration of Section 3.1 on a pretrained SD3 VAE (Esser et al., 2024) whose encoder is frozen and whose decoder is fine-tuned. Let $x \in \mathbb{R}^{3\times H\times W}$ be an input image and $z = E_{\text{VAE}}(x) \in \mathbb{R}^{C\times H_z\times W_z}$ (omitted $\bar{z}$ since KL collapsed to a deterministic MSE) its latent at spatial stride s, with $C = 16$, $H_z = H/s$, and $W_z = W/s$. A learnable ViT encoder E_{ViT} patchifies z into n pre-quantization tokens in $\mathbb{R}^{n\times D}$, which the sign quantizer maps to the binary code

$$z_q \;=\; \operatorname{sign}\big(E_{\text{ViT}}(z)\big) \;\in\; \{-1,+1\}^{n\times D}. \tag{4}$$

A symmetric ViT decoder returns the code to the latent space as $G_{\text{ViT}}(z_q) \in \mathbb{R}^{C\times H_z\times W_z}$, and the pretrained decoder reconstructs $\hat{x} = G_{\text{VAE}}(G_{\text{ViT}}(z_q))$.

Objective. The conditional KL yields the latent alignment objective. In practice, we additionally reconstruct the image from the recovered latent, allowing image-space supervision to train the recovery map. Reconstruction at the recovered latent mean serves as a deterministic surrogate for the ELBO's reconstruction expectation over the target posterior. This supervision uses the pixel loss $\mathcal{L}_{\text{recon}}$ and perceptual loss $\mathcal{L}_{\text{vgg}}$. Dropping constants from equation 3 gives the latent alignment loss

$$\mathcal{L}_{\text{align}} \;=\; \|z - G_{\text{ViT}}(z_q)\|^2, \tag{5}$$

kept at a small weight to seal the one channel reconstruction leaves open, the co-adaptation of the fine-tuned decoder to a drifted recovery. Table 4 confirms the effect the each, with the explicit term changing converged quality only marginally. With an adversarial term $\mathcal{L}_{\text{GAN}}$ and the standard lookup-free commitment and entropy terms $\mathcal{L}_{\text{quant}}$, the full objective is

$$\mathcal{L} \;=\; \beta_{\text{align}}\mathcal{L}_{\text{align}} + \beta_{\text{recon}}\mathcal{L}_{\text{recon}} + \beta_{\text{vgg}}\mathcal{L}_{\text{vgg}} + \beta_{\text{GAN}}\mathcal{L}_{\text{GAN}} + \mathcal{L}_{\text{quant}}, \tag{6}$$

with all weights, including those inside $\mathcal{L}_{\text{quant}}$, in Appendix C.

Discriminator. The adversarial loss $\mathcal{L}_{\text{GAN}}$ uses a discriminator built on a separate trainable copy of the pretrained SD3 VAE encoder in place of the standard PatchGAN (Isola et al., 2017; Esser et al., 2021). The backbone maps the image to its $H/8 \times W/8 \times 32$ pre-sampling feature map, which a learnable linear head reduces to a scalar logit after global average pooling. Backbone and head are trained jointly with the tokenizer, and Section 4.3 compares this choice against PatchGAN and DINO discriminators.

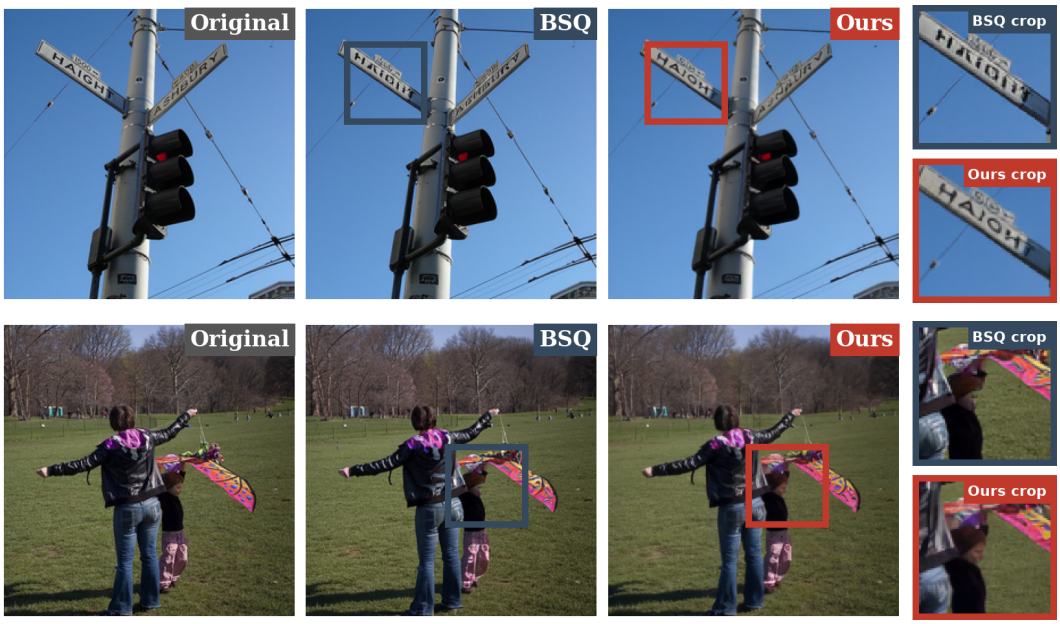


Figure 3: Qualitative comparison on MS-COCO between BSQ and our tokenizer at the matched bitrate. Our method demonstrates lower visual degradation on reconstruction with texts, and faces retaining integrity.

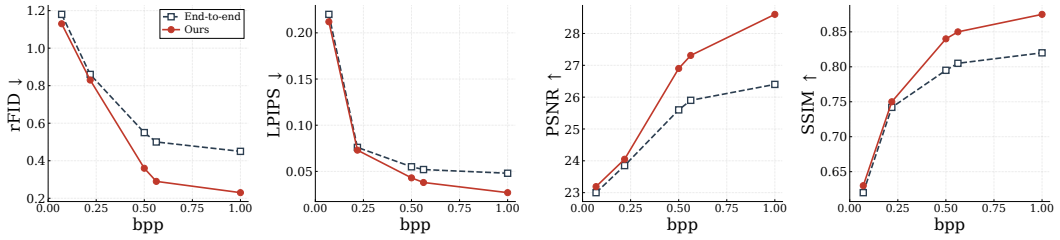


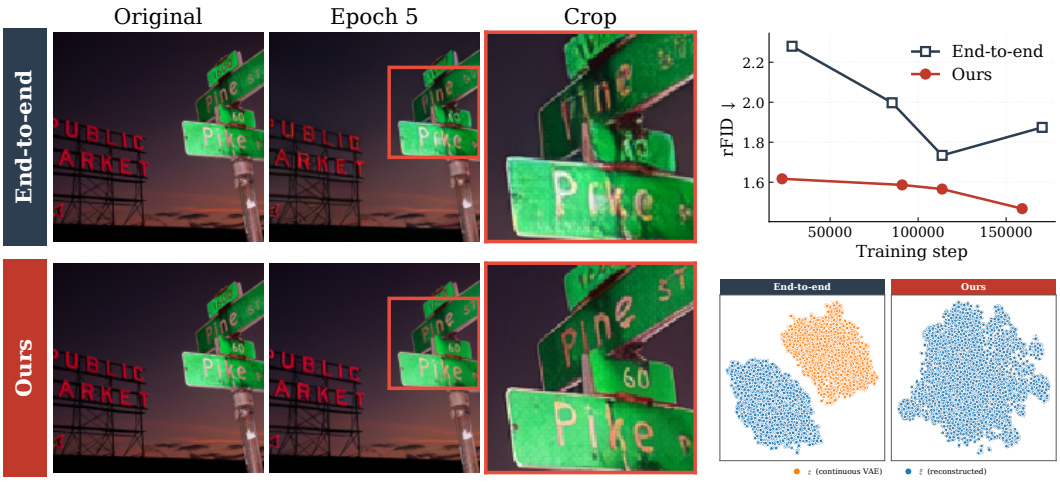
Figure 4: Reconstruction quality as a function of bpp for our method and a matched end-to-end baseline trained without latent alignment. The performance gap widens monotonically with bpp.

4 Experiments

Our primary objective in this experimental campaign is to demonstrate that the discrete tokenizer attains reconstruction quality approaching the continuous VAE reference while operating at substantially lower bits per pixel, narrowing the reconstruction gap between discrete and continuous image tokenizers. As a secondary check, we conduct ImageNet generation at 128×128 following the protocols of BSQ (Zhao et al., 2024) and Flowmo (Sargent et al., 2025) solely to verify whether any part of the objectives over-specialize the bottleneck for reconstruction at the expense of generation; we do not aim to compete with state-of-the-art image synthesis and therefore omit comparisons with generators built on representation autoencoders.

Benchmarks and baselines. We train on the ImageNet-1K training split (Russakovsky et al., 2015), roughly 1.28M images, resized to a shortest side of 256 and center-cropped to 256×256, and evaluate on the ImageNet-1K validation set and, zero-shot, on MS-COCO val2017 (Lin et al., 2014), matching baselines. We compare against discrete tokenizers at matched bpp, with BSQ and FlowMo as the direct competitors, VQGAN, MaskGIT, and LlamaGen as VQ-based references, and the SD, SDXL, SD3, and WAN VAEs as continuous references at their nominal 16 bits per channel.

Implementation and Training. Here the backbone is the pretrained SD3 VAE (Esser et al., 2024), a $C = 16$ channel latent at $8\times$ stride, with the encoder frozen and the 42M decoder fine-tuned. The



(a) (b)

Figure 5: Accelerated convergence compared to end-to-end, with adversarial refinement disabled in both models. **(a)** Reconstructions at epoch 5 for a standard end-to-end LFQ tokenizer (top) and our method (bottom) under the same architecture; the crop highlights the sharper text and corrected color our method recovers early in training. **(b)** Our method converges faster and to a lower rFID (top), and reconstructs the continuous latent structure that end-to-end training fails to preserve (bottom).

ViT encoder and decoder use FLUX transformer blocks (Labs, 2024; Labs et al., 2025) with the text-conditioning paths removed, matching FlowMo (Sargent et al., 2025), in a small configuration of hidden dimension 384, 6 layers per side, and MLP ratio 4, for 21M parameters, with the sign quantizer of Section 3.2 at the bottleneck. All experiments run on 16 NVIDIA H200 GPUs.

4.1 Image Compression

Table 1 reports ImageNet-1K and MS-COCO validation performance at 256×256. We train one tokenizer per operating point, a low rate of 0.070, a medium rate of 0.219, and a high rate of 0.5625 bpp, sharing the training framework, frozen SD3 encoder, and fine-tuned SD3 decoder, and differing only in token layout, with 256×18, 1024×14, and 1024×36 tokens$\times$bits respectively. Each rate matches the strongest baselines at that point, FlowMo-Lo at the low rate, LlamaGen-32 and FlowMo-Hi at the medium rate with the same 1024 tokens and 2^{14} codes, and BSQ-36 at the high rate, with CODA-FLUX (0.547) nearby and Λ_{24}-SQ (0.275) adjacent to BSQ-18. Even the high rate is approximately $7.1\times$ smaller than the nominal 4 bpp of SD3-VAE and WAN-VAE. We refer to these as the low, medium, and high rate.

Comparisons. At the high rate, our tokenizer attains the best LPIPS, rFID, and SSIM among the discrete methods in Table 1 on both datasets. Against BSQ-36 this is an rFID reduction of 0.12 on ImageNet and 0.58 on COCO (29% and 17% relative), with LPIPS gains of 0.005 and 0.004 and SSIM gains of 0.01 on both. BSQ-36 keeps a PSNR lead of 0.57 dB on ImageNet, narrowing to 0.09 dB on COCO, the usual perceptual-versus-pixel tradeoff under strong perceptual and adversarial supervision. The nearly rate-matched CODA-FLUX trails on every reported metric. CODA adapts its VAE backbone, whereas we freeze the encoder and fine-tune the decoder. Measured against the SD3 reference of 0.20 rFID, the high rate model closes 57% of the gap from BSQ-36, supporting discrete tokens that approach continuous fidelity at a seventh of the nominal rate. At the medium rate, our model ties FlowMo-Hi on LPIPS and trails FlowMo-Hi and LlamaGen-32 on rFID and PSNR, and Λ_{24}-SQ matches its rFID at 0.275 bpp with stronger pixel metrics. At the low rate, we lead FlowMo-Lo on PSNR but trail on the perceptual metrics. Both regimes follow the capacity analysis below, where the code's shrinking capacity to recover the SD3 latent saturates recovery at the irreducible quantization error and the method reduces to end-to-end training.

Monotonic improvement. The gain over end-to-end training grows monotonically with bpp (Figure 4). The mechanism is structural. At low bpp the code lacks the capacity to represent the frozen target, so recovery saturates at the irreducible quantization error and our method becomes indistinguishable from the end-to-end baseline. As bpp grows the code can approximate the target ever more faithfully, and stationarity lets the recovery pressure accumulate against a fixed source rather than dissipate against a drifting one.

Accelerated convergence. Finally, we isolate the effect of the frozen target on convergence, comparing two tokenizers of identical architecture without adversarial refinement, a standard end-to-end LFQ tokenizer and our anchored model. The baseline has no incentive to keep its pre-quantization tokens near their codes, incurring a $8.4\times$ larger quantization error (Figure 1(b) as defined in Appendix B, and correspondingly biased straight-through gradients. Anchoring removes this mismatch. The perceptual loss converges $25.6\times$ faster and settles lower (Figure 1(c)), in-training rFID follows (Figure 5b), by epoch 5 our method recovers sharp text and faithful color where the baseline still blurs (Figure 5a), and a final-step t-SNE places our reconstructions on the continuous targets, the baseline's in a separate cluster (b, bottom).

4.2 Discrete Image Generation

To assess whether the tokenizer's reconstruction-focused design affects downstream generative capability, we train a class-conditional MaskGIT-style generator with the procedure explained in Section E on ImageNet at 128×128 resolution. We compare against BSQ-18 (Zhao et al., 2024) at the matched 18-bit code width, the only setting for which BSQ reports a generation result, and additionally include our 36-bit configuration to assess scaling behavior. Sampling uses 32 decoding steps with a MaskGIT temperature of 1.5 and a classifier-free guidance scale of 4.0 for both code widths. Table 2 reports these results.

Table 2: Class-conditional generation on ImageNet 128×128 using MaskGIT generator with the detail outlined in Appendix E. gFID-50K is computed against the full ImageNet training distribution. † Results taken from (Zhao et al., 2024). Best in **bold**, second-best with underline.

Tokenizer	Bits	gFID↓	IS↑
Diffusion (ADM)†	-	5.91	93.3
VQ†	-	9.4	-
BSQ-18† Zhao et al. (2024)	18	<u>5.44</u>	139.60
Ours	18	**4.98**	<u>148.61</u>
Ours	36	6.64	**185.91**

Our tokenizer achieves competitive generation quality with prior discrete tokenizers such as BSQ. At the matched 18-bit configuration, the reconstruction improvements established in Table 1 translate into corresponding gains on generation; our model surpasses BSQ-18 on both gFID and IS under an equal bit budget. At the larger 36-bit configuration, however, we observe the reconstruction-generation dilemma reported in prior work (Yao et al., 2025). While IS continues to improve as the generator gains access to richer per-token information, gFID degrades because the substantially larger discrete code space makes the underlying generative modeling task harder to fit at our fixed training budget.

4.3 Discriminator effect

To isolate the effect of the SD3 discriminator, we conduct a controlled comparison against PatchGAN (Isola et al., 2017) and a DINO-based discriminator (Sauer et al., 2023; Wang et al., 2024b) under a matched training protocol. The DINO discriminator mirrors the architectural pattern of our SD3 discriminator, with a learnable linear head sitting on top of the pretrained backbone and the backbone itself made trainable during adversarial finetuning. Since both SD3 and DINO inherit pretrained weights, we adopt a two-phase schedule for them. The tokenizer is first trained for 20 epochs with the

Table 3: Discriminator ablation. *Joint* trains 30 epochs from scratch. *Two-phase* pretrains the autoencoder for 20 epochs, then enables GAN finetuning for 10 epochs. gFID on ImageNet 128×128 generation.

Disc type	LPIPS↓	rFID↓	gFID↓
No GAN	0.0632	1.5658	-
PatchGAN (joint)	**0.038**	<u>0.401</u>	-
PatchGAN (two-phase)	**0.038**	0.4122	-
DINO	0.0586	1.4126	11.23
SD3 (ours)	**0.038**	**0.29**	6.64

non-adversarial terms of Equation 6; the relevant discriminator is then enabled for an additional 10 epochs. The VAE encoder remains frozen while the ViT tokenizer, VAE decoder, and discriminator are optimized. PatchGAN, which has no pretrained initialization, is run under the same two-phase protocol and additionally under a single-phase protocol with adversarial training enabled throughout 30 epochs, following the joint-training approach of prior tokenizers (Esser et al., 2021; Yu et al., 2021). Results are reported in Table 3.

4.4 Ablation studies

To distinguish each component's contribution, we ablate Ours-L at 256×256 and 0.562 bpp, removing contributions in reverse order of introduction (Section 3). Two transitions in Table 4 account for nearly all of the gain.

Adversarial refinement. Enabling the GAN sharply improves perceptual quality (LPIPS $0.063 \rightarrow 0.038$, rFID $1.57 \rightarrow 0.30$) at a small cost in PSNR and SSIM. This stage is standard across tokenizers; what distincts ours is the SD3 discriminator that drives the gain, studied in Section 4.3.

Pre-trained initialization is not responsible for gains. Freezing the SD3 encoder improves LPIPS from 0.084 to 0.062 and rFID from 1.76 to 1.57. With the encoder frozen, the post-quantization latent must recover the continuous manifold for the SD3 decoder to reconstruct stably; this pressure drives down the STE error and produces both the faster convergence shown above and the jump in reconstruction quality. Since a randomly initialized encoder matches a pretrained one once both are trainable (final two rows), we can conclusively attribute these gains to our framework rather than to the pretrained initialization.

Table 4: Component ablation on ImageNet reconstruction at 256×256 and 0.562 bpp (the Ours-L operating point). Each row cumulatively removes one component from the row above, in reverse order of introduction (Section 3).

Configuration	LPIPS↓	rFID↓	PSNR↑	SSIM↑
Ours-L (full)	0.038	0.29	27.31	0.85
− SD3 discriminator	0.063	1.57	28.15	0.87
− $\mathcal{L}_{\text{align}}$	0.062	1.57	28.22	0.87
− frozen encoder	0.083	1.75	27.28	0.85
− pretrained init	0.084	1.73	26.87	0.84

5 Discussion & Conclusion

In this work, we addressed the quantization error at the bottleneck of discrete visual tokenizers. Casting tokenization as a two-latent generative model, we derived an ELBO whose discrete-root form isolates the error as a single conditional KL, unified the commitment, codebook, and structural mechanisms of prior tokenizers as its partial instantiations, and identified the configuration the term requires, a stationary and informative target with a learned recovery. We constructed this configuration by anchoring to the frozen latent of a pretrained SD3 encoder, reducing normalized quantization error by $8.4\times$ and reaching the end-to-end baseline's final training perceptual loss in $25.6\times$ fewer training steps. Separately, the reported epoch budgets imply approximately $6.7\times$ fewer tokenizer-training image exposures than BSQ. Our high-rate tokenizer achieves the best rFID, LPIPS, and SSIM among the discrete methods compared on ImageNet and MS-COCO, while supporting MaskGIT generation through grouped classification heads. It reaches within 0.09 rFID of the SD3 reference at 0.5625 bpp, approximately $7.1\times$ below the reference's nominal 4 bpp storage rate. This is a lossy reconstruction operating point, not a measurement of the continuous latent's entropy or a lossless compression bound.

Limitations. LABQ remains fragile at very low bitrates, where limited binary capacity makes it difficult to preserve the frozen VAE latents and small latent errors can degrade reconstruction. Reconstruction also depends on the information retained by the frozen encoder. Further improvements that require details absent from this representation may therefore require additional training of the underlying VAE, including its encoder, to retain more information relevant to reconstruction.

REFERENCES

Shiyue Cao, Yueqin Yin, Lianghua Huang, Yu Liu, Xin Zhao, Deli Zhao, and Kaigi Huang. Efficient-vqgan: Towards high-resolution image generation with efficient vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7368–7377, 2023.

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.

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*, pp. 12873–12883, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15733–15744, 2025.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space, 2025. URL `https://arxiv.org/abs/2506.15742`.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Zeyu Liu, Zanlin Ni, Yeguo Hua, Xin Deng, Xiao Ma, Cheng Zhong, and Gao Huang. Coda: Repurposing continuous vaes for discrete tokenization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18906–18916, 2025.

Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-magvit2: An open-source project toward democratizing auto-regressive visual generation. *arXiv preprint arXiv:2409.04410*, 2024.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

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*, pp. 10684–10695, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Kyle Sargent, Kyle Hsu, Justin Johnson, Li Fei-Fei, and Jiajun Wu. Flow to the mode: Mode-seeking diffusion autoencoders for state-of-the-art image tokenization. *arXiv preprint arXiv:2503.11056*, 2025.

Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In *International conference on machine learning*, pp. 30105–30118. PMLR, 2023.

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.

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.

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

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Hanyu Wang, Saksham Suri, Yixuan Ren, Hao Chen, and Abhinav Shrivastava. Larp: Tokenizing videos with a learned autoregressive generative prior. In *The Thirteenth International Conference on Learning Representations*, 2024a.

Jiangtao Wang, Zhen Qin, Yifan Zhang, Vincent Tao Hu, Björn Ommer, Rania Briq, and Stefan Kesselheim. Scaling image tokenizers with grouped spherical quantization. *arXiv preprint arXiv:2412.02632*, 2024b.

Yuqing Wang, Zhijie Lin, Yao Teng, Yuanzhi Zhu, Shuhuai Ren, Jiashi Feng, and Xihui Liu. Bridging continuous and discrete tokens for autoregressive visual generation. *arXiv preprint arXiv:2503.16430*, 2025.

Mark Weber, Lijun Yu, Qihang Yu, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. Maskbit: Embedding-free image generation via bit tokens. *arXiv preprint arXiv:2409.16211*, 2024.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15703–15712, 2025.

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 vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, et al. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10459–10469, 2023a.

Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023b.

Jiahui Zhang, Fangneng Zhan, Christian Theobalt, and Shijian Lu. Regularized vector quantization for tokenized image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18467–18476, 2023.

Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. *arXiv preprint arXiv:2406.07548*, 2024.

Yue Zhao, Hanwen Jiang, Zhenlin Xu, Chutong Yang, Ehsan Adeli, and Philipp Krähenbühl. Spherical leech quantization for visual tokenization and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12913–12923, 2026.

Lei Zhu, Fangyun Wei, Yanye Lu, and Dong Chen. Scaling the codebook size of vq-gan to 100,000 with a utilization rate of 99%. *Advances in Neural Information Processing Systems*, 37:12612–12635, 2024.

Shaobin Zhuang, Yiwei Guo, Canmiao Fu, Zhipeng Huang, Zeyue Tian, Xiaohui Li, Fangyikang Wang, Ying Zhang, Chen Li, and Yali Wang. Wetok: Powerful discrete tokenization for high-fidelity visual reconstruction. *arXiv preprint arXiv:2508.05599*, 2025.

A THE TWO-LATENT ELBO AND ITS HIERARCHIES

This appendix derives the bound of Equation 1 and instantiates it under the two factorizations of Section 3.1. Both hierarchies follow from a single root-conditional derivation, which also yields the continuous-root form omitted from the main text.

The general bound. For any approximate posterior $q_\phi(z, z_q \mid x)$,

$$\log p_\theta(x) \;=\; \mathcal{L}(\theta, \phi; x) \;+\; \mathrm{KL}\big(q_\phi(z, z_q \mid x) \,\big\|\, p_\theta(z, z_q \mid x)\big), \tag{7}$$

which follows from writing $p_\theta(x) = p_\theta(x, z, z_q)/p_\theta(z, z_q \mid x)$ and taking the expectation under q_ϕ. Non-negativity of the KL term gives Equation 1, exactly as in the single-latent case.

Root-conditional factorizations. Let (r, c) denote an ordering of the latent pair into a root and a child, so that $(r, c) = (z_q, z)$ in the discrete-root hierarchy and $(r, c) = (z, z_q)$ in the continuous-root hierarchy. Both factorizations of Section 3.1 share the form

$$p_\theta(x, z, z_q) \;=\; p_\theta(x \mid z, z_q)\, p_\theta(c \mid r)\, p(r), \qquad q_\phi(z, z_q \mid x) \;=\; q_\phi(r \mid x)\, q_\phi(c \mid r, x). \tag{8}$$

Substituting into Equation 1 and pairing each posterior factor with its model counterpart gives

$$\mathcal{L} \;=\; \underbrace{\mathbb{E}_{q_\phi}[\log p_\theta(x \mid z, z_q)]}_{\text{reconstruction}} \;-\; \underbrace{\mathrm{KL}\big(q_\phi(r \mid x) \,\big\|\, p(r)\big)}_{\text{root prior matching}} \;-\; \underbrace{\mathbb{E}_{q_\phi(r|x)}\, \mathrm{KL}\big(q_\phi(c \mid r, x) \,\big\|\, p_\theta(c \mid r)\big)}_{\text{child recovery}}. \tag{9}$$

The two hierarchies therefore share one bound and differ only in which latent each term binds.

Discrete root. With $(r, c) = (z_q, z)$, Equation 9 is Equation 2. The sign quantizer makes $q_\phi(z_q \mid x)$ a point mass, so the root term equals $-\log p(\hat{z}_q(x))$, which is the constant $nD \log 2$ under the uniform prior. The child term compares two distributions over the continuous latent, and under the Gaussian parameterization of Section 3.1 the identity $\mathrm{KL}\big(\mathcal{N}(a, \sigma^2 I) \,\|\, \mathcal{N}(b, \sigma^2 I)\big) = \|a - b\|^2 / 2\sigma^2$ yields Equation 3. The hierarchy thus places the quantization error itself inside the bound.

Continuous root. With $(r, c) = (z, z_q)$, Equation 9 becomes

$$\mathcal{L}_{\mathsf{cont}} \;=\; \mathbb{E}_{q_\phi(z|x)\, q_\phi(z_q|z,x)}[\log p_\theta(x \mid z, z_q)] - \mathrm{KL}\big(q_\phi(z \mid x) \,\big\|\, p(z)\big) - \mathbb{E}_{q_\phi(z|x)}\, \mathrm{KL}\big(q_\phi(z_q \mid z, x) \,\big\|\, p_\theta(z_q \mid z)\big). \tag{10}$$

The first two terms are the standard continuous-VAE objective. The third compares the code assignment with a learned code predictor $p_\theta(z_q \mid z)$ and, for a deterministic quantizer, reduces to the cross-entropy of that predictor at the assigned code, a term familiar from entropy-coding priors. No term in Equation 10 measures how well the continuous latent can be recovered from the code, so this hierarchy maintains no continuous representative per code and exerts no control over the quantization error. The distinction is also operational. Sampling from a hierarchy proceeds from the root, so the continuous-root form describes pipelines that sample a continuous prior and treat the code as an annotation of the latent, whereas a tokenizer stores only the code and must regenerate the latent from it, which is precisely the discrete-root order $z_q \sim p(z_q)$ followed by $z \sim p_\theta(z \mid z_q)$.

B METRIC DEFINITIONS AND CONVERGENCE PROTOCOL

This appendix defines the quantization error for arbitrary tokenizers, the measurement behind Figure 1(b), and the convergence speedups of Section 4.1.

Quantization error. For any tokenizer, let u denote the continuous representation produced by the encoder at the interface severed by quantization, and let $r(z_q)$ denote the continuous value the subsequent decoder consumes in its place. The quantization error is the relative mean squared gap

$$\varepsilon_Q \;=\; \frac{\mathbb{E}_x \big\| u - r(z_q) \big\|_F^2}{\mathbb{E}_x \big\| u \big\|_F^2}, \tag{11}$$

the empirical, scale-invariant counterpart of the squared distance between target and recovery in Section 3.1. The definition instantiates across training schemes. In VQ the decoder consumes the

codebook entry, $r(z_q) = e_{z_q}$, and ε_Q is the normalized commitment distance. In LFQ and BSQ the decoder consumes the code itself, $r(z_q) = z_q$, with u projected onto the unit hypersphere in BSQ so that ε_Q is bounded by construction. In our pipeline the decoder G_{VAE} consumes the recovered latent, so the interface is the latent space itself, with $u = z$ and $r(z_q) = G_{\text{ViT}}(z_q)$, and ε_Q is the normalized alignment loss of Equation 5. The sign step inside the bottleneck is the local cut the straight-through estimator bridges, and the latent-interface error is the pipeline-level distortion this local approximation accumulates into.

Measurement protocol. Figure 1(b) reports the latent-interface error for the anchored model and the end-to-end baseline, each measured in its own latent space, between what its encoder produced and what its VAE decoder received. The relative form of Equation 11 makes the two comparable despite the baseline's latent scale being free, since each error is normalized by the second moment of its own target. The two models share architecture, data, batch size, optimizer, and schedule. Each value is estimated on training batches and averaged over the final logging windows of each run, and the end-to-end error exceeds the anchored error by $8.4\times$. We corroborate the connection between this distortion and biased straight-through training through the convergence comparisons below rather than measuring the gradient bias directly.

Within-paper convergence speedup. The two runs being compared share architecture, data, batch size, optimizer, and hardware, with adversarial refinement disabled, the VAE decoder trainable, and the quantization terms of Equation 12 active at identical weights in both. The runs differ only in whether the target encoder is frozen. Let $\ell(t)$ denote the smoothed training perceptual loss at step t, and let τ be the value reached by the end-to-end baseline at the end of its full schedule. The convergence step of a run is the first t with $\ell(t) \le \tau$. The baseline never reaches the anchored model's converged loss within its schedule, so τ is the most demanding threshold at which both runs are measurable, and the reported speedup is a lower bound on the advantage. The baseline attains τ only at the end of its $\sim$180k-step schedule while the anchored run attains it within its first $\sim$7k steps, giving the reported $25.6\times$ (Figure 1(c)). The in-training rFID comparison of Figure 5 follows the same protocol with rFID in place of the perceptual loss.

Cross-paper comparison with BSQ. Comparisons with BSQ cannot share hardware or code, so we normalize by training exposure, defined as the number of images processed, computed as optimization steps multiplied by global batch size. BSQ-ViT trains its image tokenizer for 1M steps at global batch size 256, reported as 200 epochs over ImageNet-1K (Zhao et al., 2024), a total of $\approx$ 256M exposures, while our full schedule processes 38.4M exposures, corresponding to 30 epochs over the 1.28M ImageNet-1K training images. The reported roughly $6\times$ is the ratio of these budgets, $256/38.4 \approx 6.7$, at each method's reported final reconstruction quality. This figure compares published training budgets rather than reproduced convergence curves, since we do not retrain BSQ.

C Training hyperparameters

We train the tokenizer with the objective of Equation 6. Table 5 lists all weights and the discriminator configuration. The pixel and perceptual terms carry unit weight and supply the primary reconstruction signal, while the adversarial term is kept small at 0.05, since the SD3 discriminator refines perceptual quality rather than driving reconstruction, under a hinge loss.

The quantization term groups the standard lookup-free commitment and entropy components. With pre-quantization tokens $u = E_{\text{ViT}}(z)$ and $q(z_q \mid x)$ the per-dimension soft assignment induced by u,

$$\mathcal{L}_{\text{quant}} \;=\; \beta_{\text{commit}} \left\| u - \text{sg}(z_q) \right\|^2 \;+\; \beta_{\text{ent}}\, \mathbb{E}_x \big[H\big(q(z_q \mid x)\big) \big] \;-\; \beta_{\text{use}}\, H\big(\mathbb{E}_x[q(z_q \mid x)]\big), \tag{12}$$

following the entropy objective of LFQ (Yu et al., 2023b). The per-sample term sharpens code confidence, and the usage term drives the aggregate code distribution toward the uniform prior $p(z_q)$ of Section 3.1, encouraging the binary code to span its range.

The latent alignment term serves the guardrail role described in Section 3.2. Most of the recovery pressure comes from reconstruction through the decoder pretrained on the frozen latent space, and the explicit term prevents the fine-tuned decoder from co-adapting to a drifted recovery in late-stage

Table 5: Loss weights, discriminator, and training configuration. The weights β_* correspond to Equations 6 and 12.

Group	Hyperparameter	Value
Alignment	β_{align}	0.1
Reconstruction	β_{recon}	1.0
Perceptual	β_{vgg}	1.0
Adversarial	β_{GAN}	0.05
Quantization ($\mathcal{L}_{\text{quant}}$)	β_{commit}	0.25
	β_{ent} (per-sample entropy)	0.1
	β_{use} (codebook usage)	0.5
	entropy group size	12
Discriminator	type	SD3
	loss	hinge
	start epoch	20
	reinitialized at start	yes
Optimization	optimizer	AdamW
	learning rate	3×10^{-4}
	(β_1, β_2)	$(0.9, 0.999)$
	epsilon	10^{-8}
	weight decay	0.001
	warmup	1 epoch
	gradient clipping	1.0
Training	global batch size	512
	precision	bfloat16
	epochs (reconstruction + adversarial)	$20 + 10$
	weight EMA	none
	random seed	42

training. Our ablation (Table 4) confirms the explicit term adds little to converged quality, so we set β_{align} to a small 0.1, enough to hold the anchor without competing with the terms that carry the signal.

All trainings and inference are conducted on Nvidia H200 GPUs.

D Representation collapse under joint alignment

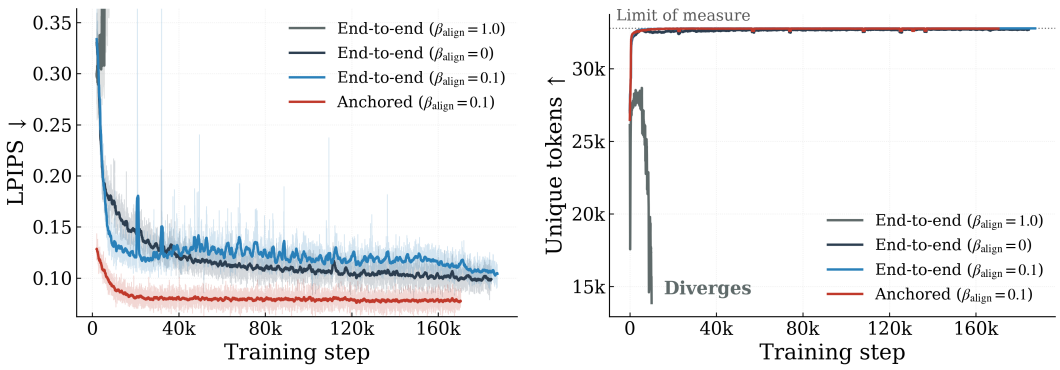
Section 3.1 identifies the conditional KL of Equation 2 as the canonical objective coupling the continuous and discrete latents. Realizing it as a training loss requires deciding which of its two arguments is learnable. This appendix substantiates the claim made in Section 3.1: the naive realization, in which both the target distribution $q_\phi(z \mid z_q, x)$ and the recovery distribution $p_\theta(z \mid z_q)$ are trained jointly from pixels, fails and the failure is attributable to the non-stationarity of the target.

Naive joint realization. In the joint setting, the frozen VAE encoder of Section 3.2 is replaced by a trainable encoder E_ψ, so the target latent $z = E_\psi(x)$ moves during training rather than being fixed at $E_{\text{VAE}}(x)$, and the alignment term becomes

$$\mathcal{L}_{\text{align}}^{\text{joint}} \;=\; \big\|z - G_{\text{ViT}}(z_q)\big\|^2, \qquad z_q = \operatorname{sign}\big(E_{\text{ViT}}(z)\big), \tag{13}$$

with all modules trained end-to-end under the full objective of Equation 6. Here $G_{\text{ViT}}(z_q)$ realizes the prior mean $\mu_\theta(z_q)$ exactly as in the anchored model; the only change is that the target z now depends on ψ.

Degenerate optima. The alignment term is symmetric in its two arguments and indifferent to the informativeness of the target: Equation 13 is minimized whenever the encoder produces latents the finite-capacity recovery map can express, and any constant encoder paired with a matching G_{ViT} attains the global minimum while transmitting zero information through the bottleneck. Only



(a) (b)

Figure 6: Joint vs. anchored realizations of the alignment objective, adversarial refinement disabled. (a) Training LPIPS. (b) Distinct codes per logging window. Joint training at β_{align}=1.0 collapses code usage and diverges; at β_{align}=0.1 it matches the β_{align}=0 baseline. The anchored model at the same weight converges fastest and lowest.

the pixel and perceptual terms oppose these solutions. Since an $n \times D$-bit code cannot track an information-rich z, joint training resolves the mismatch from whichever side is cheaper, and the target is the cheaper side: reducing the dispersion of z across inputs is always available to the encoder, whereas the recovery map is capacity-bound. Anchoring removes the target from the optimization entirely; with z produced by the frozen E_{VAE}, the term acts only on the recovery map and the code assignment, and its minimizers are exactly the configurations that best recover a fixed, informative reference. The stop-gradient operators of VQ-VAE (Van Den Oord et al., 2017) partition the same two-sided pull in the commitment loss; anchoring is the limit in which the target receives no gradient at all and is additionally guaranteed to be informative by its pretraining.

Empirical study. We compare four realizations of the objective under an otherwise identical protocol: the architecture, data, and non-adversarial losses of Section 3.2, with adversarial refinement disabled. The runs differ only in (i) whether the VAE encoder–decoder pair is trainable (*end-to-end*) or encoder frozen (*anchored*), and (ii) the alignment weight $\beta_{\text{align}} \in \{0, 0.1, 1.0\}$. Figure 6 reports training LPIPS alongside codebook usage, measured as the number of distinct binary codes observed within each logging window; the dotted line in Figure 6(b) marks the number of tokens per window, the maximum measurable count. The measurable count is a function of the number of samples processed during training across GPUs.

The three regimes in Figure 6 isolate the role of the target. At $\beta_{\text{align}} = 1.0$ the jointly trained model diverges within the first $\sim$10k steps (a). The failure is visible in the bottleneck itself, as the number of distinct codes in use collapses from near the measurement ceiling to well under half of it concurrently with the LPIPS blow-up (b), showing that the two-sided pull of the joint term degrades the information carried through the code before a useful representation can form. At $\beta_{\text{align}} = 0.1$ joint training does not diverge however experiences moments of blow-up and recovery (a). Every stable run saturates the usage ceiling almost immediately (b), as expected for a binary lookup-free bottleneck, so the LPIPS separation among stable runs is not a codebook-utilization effect and the usage collapse is specific to the unstable joint regime. The anchored model at the same weight of 0.1 reaches LPIPS ≈ 0.08 within $\sim$20k steps and remains below every end-to-end variant throughout.

There is thus no weight at which the jointly trained term is useful, since large weights destabilize training and small weights are inert to converged values. This is consistent with the component ablation of Table 4, where removing the explicit alignment term from the anchored model changes converged quality only marginally. With the encoder and decoder frozen, reconstruction through the frozen decoder already realizes the recovery pressure implicitly, and the explicit term serves as a guardrail against late-stage drift (Appendix C).

E GENERATION FRAMEWORK

Directly generating tokens over the tokenizer's full codebook is memory-prohibitive. With the 36-bit code of the high rate tokenizer, the index space spans $2^{36} \approx 6.9 \times 10^{10}$ classes, each requiring an embedding row and a dense connection to the output head. Prior work partitions the index into subgroups (Yu et al., 2023b; Zhao et al., 2024; Luo et al., 2024), and in the limit predicts individual bits, as in next-scale prediction (Han et al., 2025) and bit-masked generation (Weber et al., 2024). We adopt the intermediate point. The MaskGIT token masking is kept intact at the token level, and the token head is replaced by grouped categorical classifiers that compose the final code.

Let $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_N)$ denote the token sequence emitted by the tokenizer, where each $\mathbf{b}_i \in \{-1, +1\}^D$ is a binary code. A mask $\mathcal{M} \subseteq \{1, \ldots, N\}$ indicates masked positions, and a learned mask embedding $\mathbf{e}_{\text{mask}}$ constructs the masked sequence:

$$\tilde{\mathbf{b}}_i = \begin{cases} \mathbf{e}_{\text{mask}}, & i \in \mathcal{M}, \\ \mathbf{b}_i, & i \notin \mathcal{M}. \end{cases} \tag{14}$$

Each code is partitioned into G subgroups $\mathbf{b}_i = (\mathbf{g}_{i,1}, \ldots, \mathbf{g}_{i,G})$ with $\mathbf{g}_{i,k} \in \{-1, +1\}^{D/G}$, and each subgroup is identified with its index $c_{i,k} \in \{1, \ldots, 2^{D/G}\}$. Given the masked sequence, the model outputs a categorical distribution over each subgroup index, $p_\theta(c_{i,k} \mid \tilde{\mathbf{b}})$, and the masked prediction loss is

$$\mathcal{L}_{\text{mask}}(\theta) = \mathbb{E}_{\mathbf{b}}\, \mathbb{E}_{\mathcal{M}} \left[\sum_{i \in \mathcal{M}} \sum_{k=1}^{G} -\log p_\theta(c_{i,k} \mid \tilde{\mathbf{b}}) \right]. \tag{15}$$

We use $G = 3$ subgroups of 12 bits for the 36-bit code. This factorization sits between a full softmax over 2^D classes, intractable at $D = 36$, and fully independent bitwise prediction, retaining tractable heads while modeling the dependencies within each subgroup jointly.

At inference we follow the iterative parallel decoding of MaskGIT (Chang et al., 2022). Decoding begins with all tokens masked and proceeds in a fixed number of steps, at each step predicting the masked positions conditioned on the visible context and unmasking the most confident tokens, where the confidence of a token aggregates its subgroup predictions as the sum of the sampled indices' log-probabilities.

F ADDITIONAL RECONSTRUCTION RESULTS

We provide further qualitative reconstructions on MS-COCO, comparing the original image, BSQ-36, and our Ours-L tokenizer at the matched operating point of 0.562 bpp (Figures 7 and 8). A consistent pattern holds across samples. BSQ introduces noise-like artifacts that trace back to its harsher per-token quantization, with fine textures breaking into high-frequency speckle and smooth regions such as skies and walls developing faint blotches. Our reconstructions do not show these artifacts; textures stay coherent and flat regions remain clean, because aligning the bottleneck to the frozen continuous latent keeps the decoded prediction on the VAE manifold rather than forcing the decoder to interpret a noisy quantized code. These observations are consistent with the lower rFID and LPIPS our method achieves in Table 1.

G GENERATED IMAGES

Here we provide an uncurrated set of samples from the generative model trained on top our proposed tokenizer.

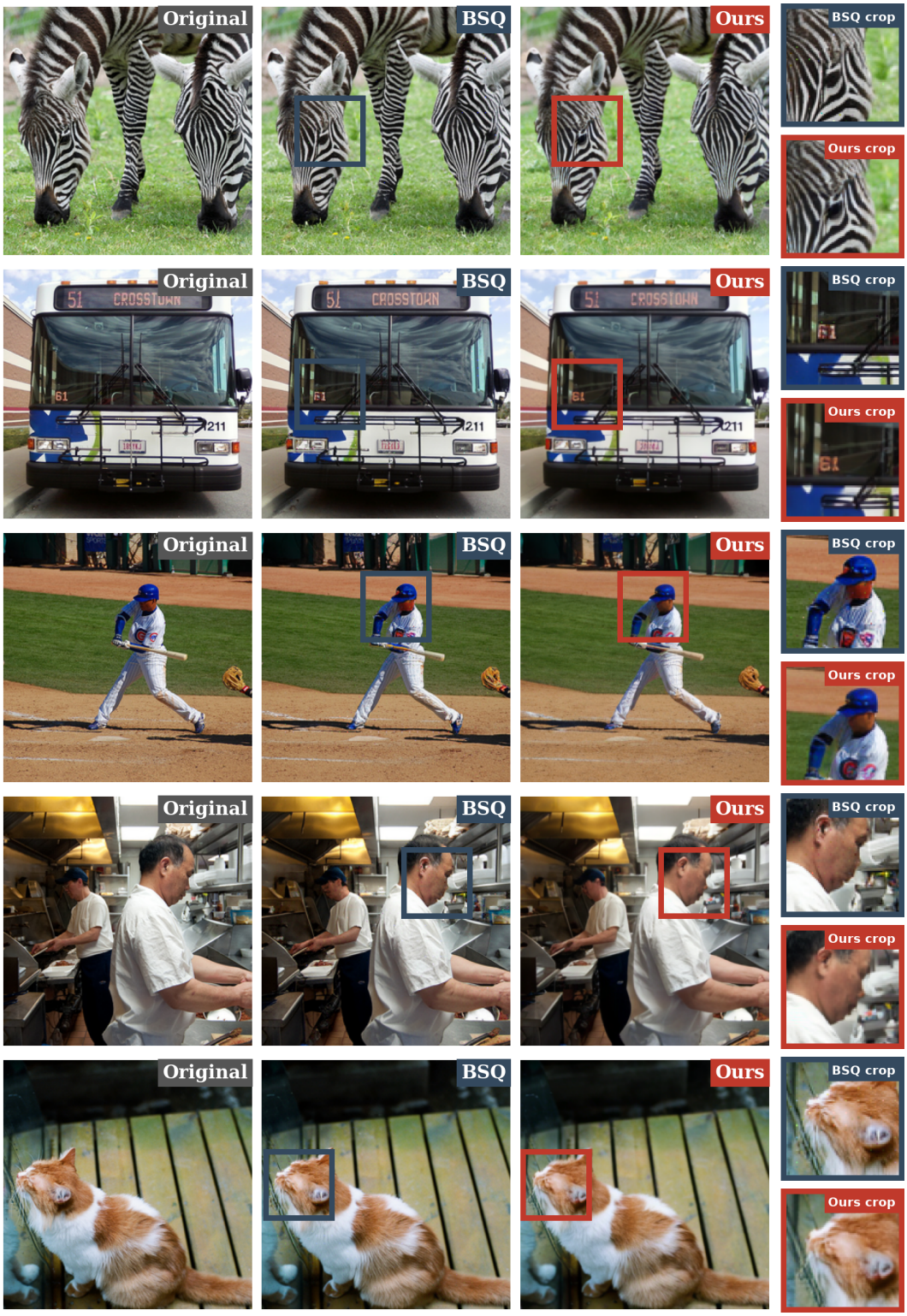


Figure 7: Additional MS-COCO reconstructions at 0.562 bpp. Each row shows, from left to right, the original image, the BSQ-36 reconstruction, and our Ours-L reconstruction. BSQ exhibits noise-like artifacts from its harsher quantization, most visible in textured and smooth regions, while our reconstructions remain clean.

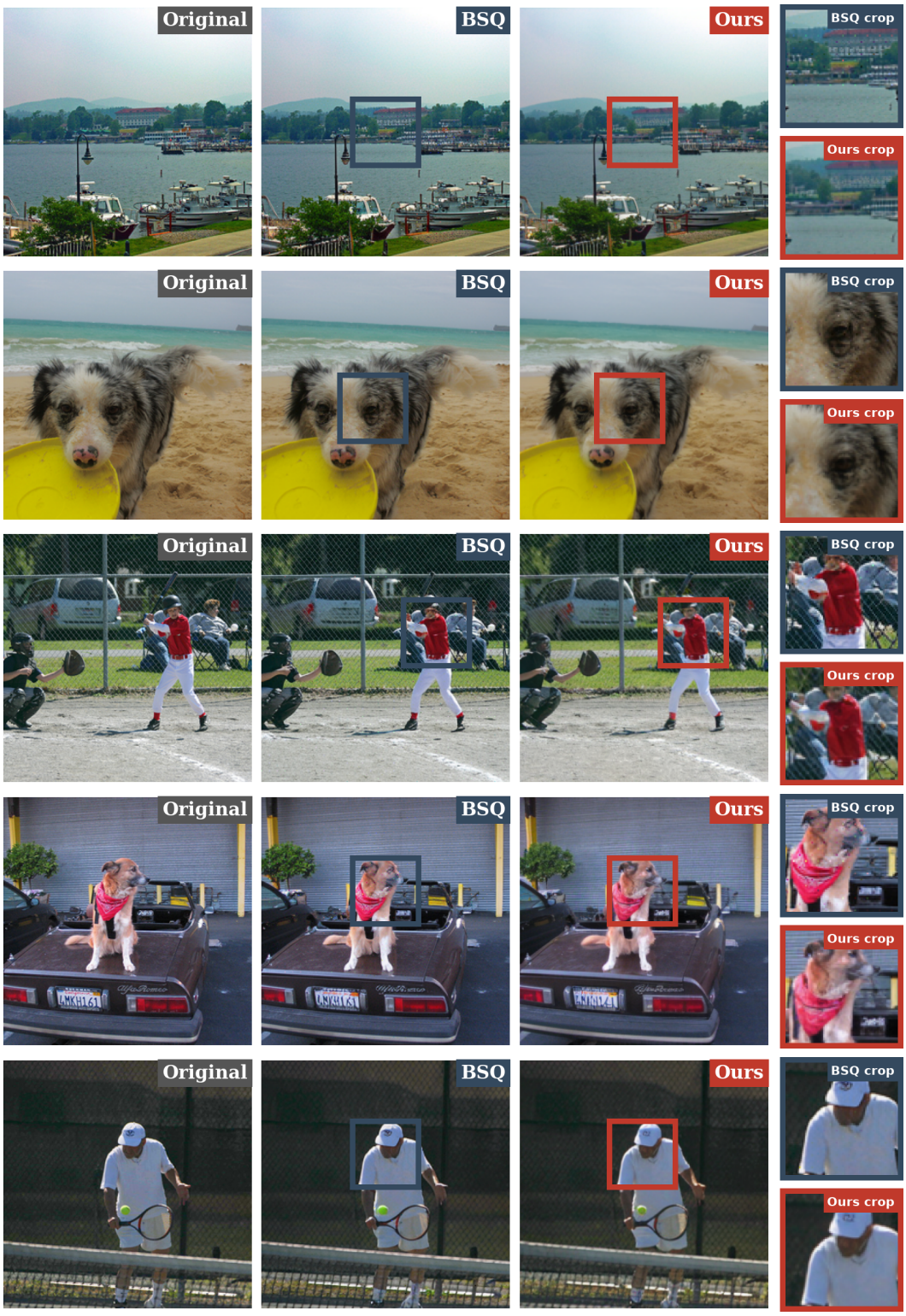


Figure 8: Additional MS-COCO reconstructions at 0.562 bpp, continued from Figure 7. Columns follow the same order (original, BSQ-36, Ours-L). The noise-like artifacts visible in the BSQ column are absent from ours.

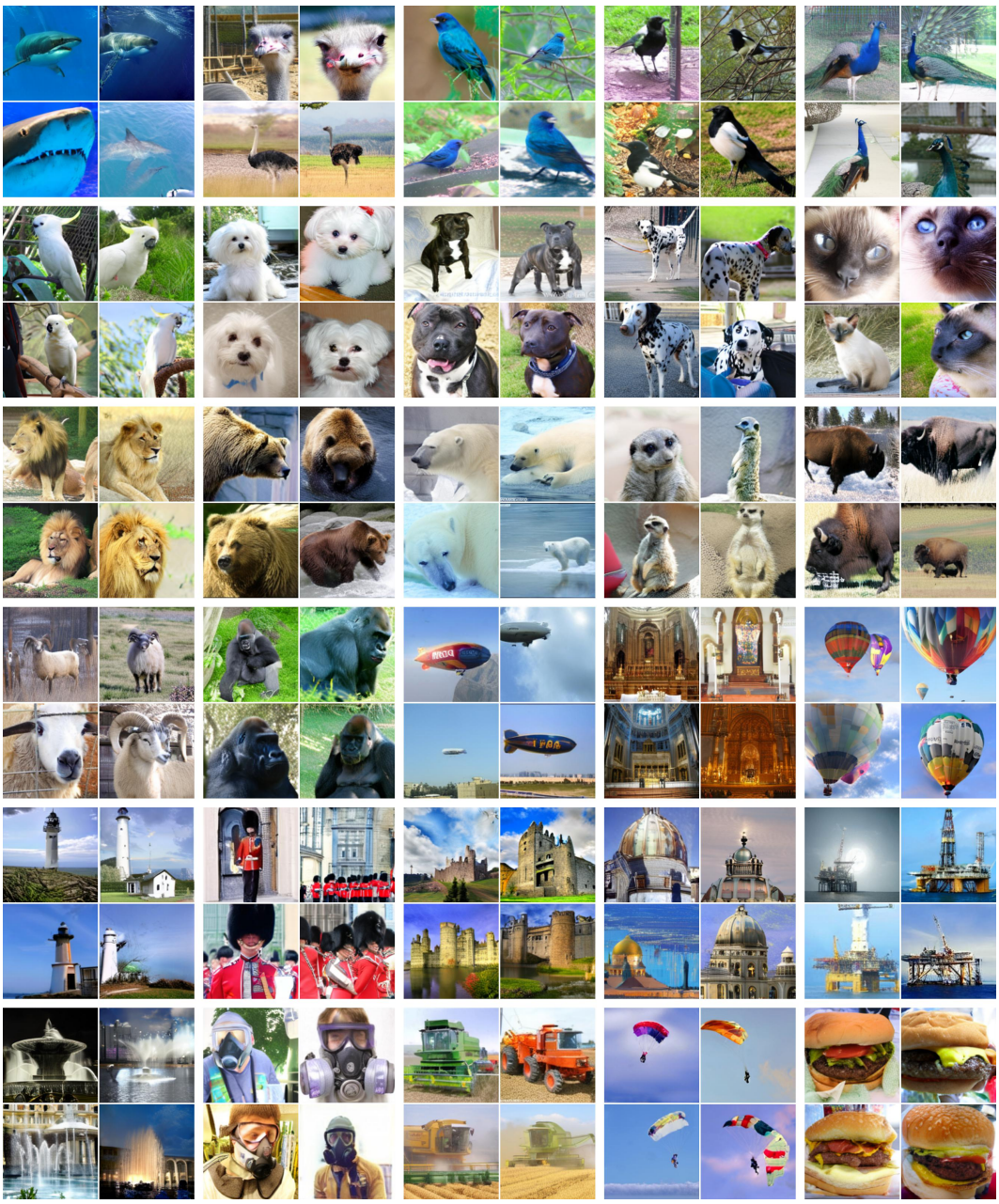


Figure 9: Uncurated samples from bitwise MaskGIT generator trained on our tokenizer (ImageNet)