RESARC: RESIDUAL-AWARE AUTOREGRESSIVE CODING FOR ULTRA-LOW BITRATE IMAGE COMPRESSION

Anonymous authors
Paper under double-blind review

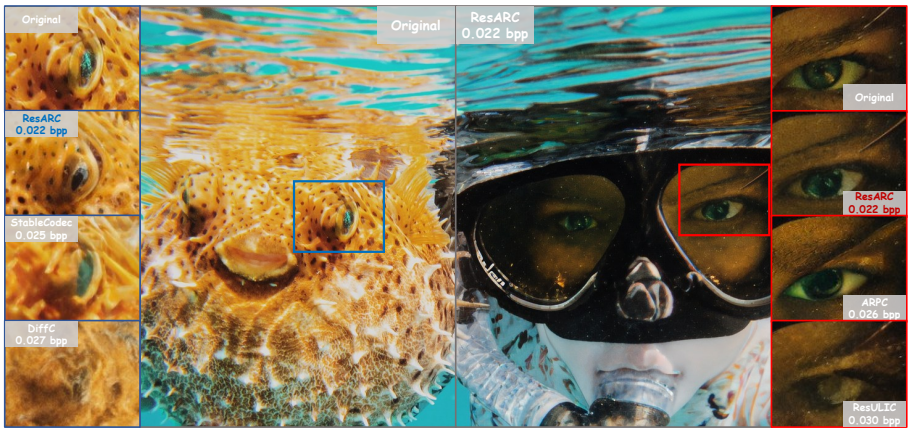

Figure 1: **Qualitative comparison with baselines at comparable ultra-low bitrates.** ResARC better preserves fine-grained textures and image structures than competing generative codecs.

ABSTRACT

Progressive autoregressive image codecs provide an appealing paradigm for generative compression by quantizing continuous latents into discrete tokens, transmitting coarse-to-fine prefix tokens and generating the remaining suffix tokens at the decoder. However, their reconstruction quality is fundamentally limited by two residuals introduced along this pipeline: the *quantization residual*, arising from information loss during discrete tokenization, and the *generation residual*, resulting from imperfect autoregressive generation of the suffix tokens. To address these limitations, we introduce **ResARC**, a residual-aware autoregressive codec that explicitly compensates for both residuals at the decoder. Specifically, we generate the quantization residual with a diffusion transformer conditioned on the autoregressive decoding context, while requiring no additional side information. In parallel, we compute the generation residual at the encoder and employ a learned Generation Residual Codec to efficiently compress and transmit it for decoder-side correction. The recovered residuals are then integrated with the reconstructed latent representation and decoded through an adapted VAE decoder. Extensive experiments demonstrate that ResARC achieves competitive perceptual similarity while substantially improving distributional fidelity over leading generative codecs across the ultra-low bitrate regime.

1 INTRODUCTION

The rapid growth of visual data has made efficient image compression increasingly important, particularly under limited bandwidth and storage. At ultra-low bitrates, the limited transmitted information makes it difficult to preserve fine textures and image structures, often leading distortion-oriented codecs (Ballé et al., 2017; Ballé et al., 2018) to produce overly smooth reconstructions. Generative codecs (Agustsson et al., 2019; Mentzer et al., 2020; Careil et al., 2024) address this by leveraging generative priors to synthesize plausible details from compressed representations, substantially improving perceptual quality at low bitrates. Early generative codecs mainly rely on GAN priors (Agustsson et al., 2019; Mentzer et al., 2020), and more recent approaches adopt diffusion models for high-quality generative reconstruction (Theis et al., 2022). However, these approaches generally lack native progressive bitrate control and tight integration between entropy modeling and the generative prior.

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.

Autoregressive generative codecs provide an alternative by exploiting the shared autoregressive prior for both entropy modeling and token generation (Mao et al., 2024; Xue et al., 2024). More recently, Visual Auto-Regressive (VAR) models (Tian et al., 2024; Han et al., 2025) generate visual tokens in a coarse-to-fine manner across multiple scales, providing a natural foundation for progressive generative compression. Building on this paradigm, ARPC (Zhang et al., 2026b) quantizes continuous latents into discrete tokens, transmits a coarse-to-fine prefix of tokens, and autoregressively generates the remaining suffix at the decoder. By varying the transmitted prefix depth, a single model can operate at multiple bitrates.

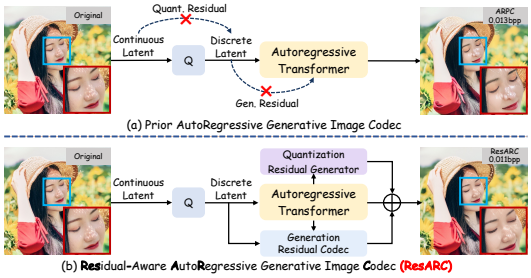

Figure 2: **Residual-aware design of ResARC.** (a) Prior autoregressive codecs introduce *quantization and generation residuals*. (b) ResARC generates the former and transmits the latter.

Despite their scalability and flexibility, we observe that the reconstruction fidelity of autoregressive generative codecs remains limited by two residuals arising along the pipeline. As illustrated in Figure 2, discrete tokenization converts the continuous latent into transmissible tokens, but inevitably discards information, including fine-grained details important for reconstruction. We define the resulting difference between the original continuous latent and its quantized reconstruction as the *quantization residual*. A second discrepancy arises when the suffix tokens omitted from transmission are replaced by autoregressively synthesized predictions, which may deviate from their ground-truth counterparts. We define this discrepancy as the *generation residual*, which further degrades reconstruction fidelity. These observations raise a central question: ***How should quantization and generation residuals be explicitly compensated in autoregressive compression?***

To answer this question, we introduce **ResARC**, the first residual-aware autoregressive image codec that explicitly compensates for both residuals according to their distinct characteristics. Specifically, the *quantization residual* is predominantly associated with fine-grained details discarded during discrete tokenization, while the reconstructed tokens and autoregressive decoding context available at the decoder provide strong cues for recovering this missing information. We therefore employ a **Quantization Residual Generator**, which uses a conditional diffusion transformer to generate the quantization residual from the decoding context without transmitting additional side information. For the *generation residual*, the discrepancy between the ground-truth and generated suffix may affect both structural content and fine details. Since this residual represents the specific correction required to align the generated suffix with its ground-truth counterpart, we compute it at the encoder and employ a **Generation Residual Codec** to efficiently compress it into a bitstream transmitted for decoder-side correction. Finally, the recovered quantization and generation residuals are fused with the reconstructed latent and decoded via an adapted VAE decoder, improving reconstruction fidelity while preserving progressive bitrate control.

Quantitative evaluations on DIV2K (Agustsson & Timofte, 2017) and CLIC2020 (Toderici et al., 2020) demonstrate that ResARC achieves strong perceptual similarity and substantially improved distributional fidelity compared with leading autoregressive and diffusion-based generative codecs in the ultra-low bitrate regime. Qualitative comparisons further show that ResARC better preserves fine-grained textures and image structures at comparable or lower bitrates. Moreover, ablation studies validate the benefits of both residual branches and the effectiveness of their respective compensation strategies.

Our principal contributions are summarized as follows:

- We identify two distinct residuals in autoregressive generative codecs: the *quantization residual* introduced by discrete tokenization, and the *generation residual* arising from imperfect autoregressive suffix generation.
- We propose **ResARC**, the first residual-aware autoregressive codec that generates the quantization residual without additional transmitted bits and compresses the generation residual into a compact bitstream transmitted for decoder-side correction.
- Extensive experiments on DIV2K and CLIC2020 demonstrate improved perceptual similarity and distributional fidelity in the ultra-low bitrate regime, with ablations further validating the complementary contributions of the two residual branches.

2 Related Work

Generative image compression. Image compression has traditionally relied on hand-crafted transforms and coding tools (Bellard, 2018), whereas modern neural codecs (Ballé et al., 2017) jointly learn representations, entropy models, and reconstruction in an end-to-end manner. While these methods primarily optimize the conventional rate-distortion trade-off, producing perceptually realistic reconstructions remains challenging, particularly in the ultra-low bitrate regime. Recent advances in visual generative modeling (Rombach et al., 2022) have therefore motivated generative codecs that leverage learned generative priors to improve perceptual reconstruction quality. GAN-based codecs established this paradigm for low bitrate perceptual compression (Agustsson et al., 2019; Mentzer et al., 2020), with later approaches further improving statistical fidelity (Muckley et al., 2023). More recently, diffusion-based codecs have conditioned generation on compressed representations (Yang & Mandt, 2023; Hoogeboom et al., 2023), semantic or visual cues (Careil et al., 2024; Körber et al., 2024), and semantic-space residuals (Ke et al., 2025). Subsequent works have improved efficiency and flexibility through shortened or one-step sampling (Relic et al., 2024; Zhang et al., 2025; Xue et al., 2025b; Shi et al., 2026), multi-rate coding (Guo et al., 2025), lightweight architectures (Zhang et al., 2026a; Jia et al., 2026a), compression-oriented pretraining (Jia et al., 2026b), video-diffusion decoding (Chen et al., 2026), and content-adaptive coding (Sheng et al., 2026). Beyond diffusion-based approaches, GLC (Jia et al., 2024) and DLF (Xue et al., 2025a) explore generative latent compression, while RDVQ (Jiang et al., 2026) improves rate-distortion optimization for vector-quantized representations. Despite these advances, existing generative codecs generally lack native progressive bitrate control and tight integration between entropy modeling and the generative prior.

Autoregressive generative codecs. Autoregressive modeling over discrete visual tokens (van den Oord et al., 2017; Esser et al., 2021; Lee et al., 2022; Yu et al., 2024; Zhao et al., 2025) provides a natural way to address the above limitations by sharing a generative prior for both entropy modeling and sequential generation, and the autoregressive prior further enables progressive coding. Early autoregressive codecs (Mao et al., 2024; Xue et al., 2024) explored this direction with conventional next-token prediction. More recently, Visual Auto-Regressive (VAR) models (Tian et al., 2024; Han et al., 2025) generate token maps in a coarse-to-fine hierarchy, providing a natural basis for progressive generative compression, while HART (Tang et al., 2025) further enhances this paradigm with continuous residual diffusion to improve image synthesis. Building on this coarse-to-fine autoregressive generation, Phung et al. (2025) first explored its application to image compression through preliminary experiments. ARPC (Zhang et al., 2026b) further develops this paradigm by transmitting coarse-to-fine prefix tokens and autoregressively generating the remaining suffix tokens, with the transmitted prefix depth controlling the bitrate within a single model. ProGVC (Li et al., 2026) extends progressive autoregressive coding to generative video compression. Despite these advances, existing autoregressive codecs remain limited by two distinct residuals: discrete tokenization discards information from continuous latents, yielding a *quantization residual*, while imperfect autoregressive suffix generation introduces a *generation residual*. ResARC explicitly models and compensates for both residuals to improve reconstruction fidelity.

3 Background: Multi-Scale Autoregressive Image Compression

We first review the progressive autoregressive codec ARPC (Zhang et al., 2026b) based on Infinity (Han et al., 2025), which underlies our method. Given an input image $x \in \mathbb{R}^{H\times W\times 3}$, a VAE encoder first extracts a continuous latent representation $h = \mathcal{E}(x)$. A multi-scale residual quantizer (Zhao et al., 2025) $\mathcal{Q}$ then converts h into a sequence of discrete token maps $\mathcal{T} = (T_1, \ldots, T_K) = \mathcal{Q}(h)$, where successive scales progressively refine the latent representation from coarse structures to fine details. The complete source token sequence is aggregated through multi-scale summation to obtain the quantized latent $h_q = \mathcal{S}(\mathcal{T}) = \sum_{i=1}^{K} U_i(T_i)$. Here, U_i maps the scale-i tokens to latent features at the common spatial resolution of h. For a given prefix depth k, the codec transmits the first k token scales $T_{\le k}$ and leaves the remaining suffix tokens $T_{>k}$ to be synthesized at the decoder. The autoregressive model ψ is utilized for both entropy coding and suffix generation: its predicted probabilities are used to entropy-code the transmitted prefix and to autoregressively sample the omitted suffix scales.

Specifically, given the text condition ℓ, the autoregressive model estimates the prefix token bitrate in bits per pixel (bpp) as

$$R^{(k)}_{\text{prefix}} = -\frac{1}{HW}\sum_{i=1}^{k}\log_2 p_\psi(T_i \mid T_{<i}, \ell). \tag{1}$$

After entropy decoding the prefix scales from the bitstream, the same autoregressive transformer predicts the conditional distribution for each remaining suffix scale and samples the corresponding tokens as

$$\hat{T}_i \sim p_\psi \left(T_i \mid T_{\leq k}, \hat{T}_{k+1:i-1}, \ell \right), \qquad i = k+1, \ldots, K. \tag{2}$$

The transmitted prefix and generated suffix are then combined to form the complete multi-scale token sequence $\hat{\mathcal{T}}^{(k)} = (T_1, \ldots, T_k, \hat{T}_{k+1}, \ldots, \hat{T}_K)$, which is aggregated and sent to the decoder for reconstruction:

$$\hat{h}_q^{(k)} = \mathcal{S}\left(\hat{\mathcal{T}}^{(k)}\right), \qquad \hat{x}^{(k)} = \mathcal{D}\left(\hat{h}_q^{(k)}\right). \tag{3}$$

By varying the prefix depth k, the codec changes the number of transmitted token scales and therefore supports progressive bitrate control within a single model.

This progressive coding paradigm reconstructs images from quantized token representations and autoregressively generated suffixes, naturally introducing two distinct residuals that motivate the residual-aware design of ResARC.

4 ResARC: Residual-aware Autoregressive Codec

4.1 Residual-Aware Formulation

ResARC builds on the progressive autoregressive coding framework introduced above. As discussed, this coding pipeline introduces two distinct discrepancies. The first arises from discrete tokenization. To obtain transmissible discrete symbols, the continuous latent h extracted by the VAE encoder is quantized into multi-scale token maps, which are aggregated to form the quantized latent h_q. Since discrete tokenization cannot preserve all information in h, the quantized latent h_q generally differs from the original continuous latent. We define the resulting discrepancy as the *quantization residual*:

$$r_q = h - h_q = h - \mathcal{S}\left(\mathcal{Q}(h)\right). \tag{4}$$

This residual represents information lost during tokenization, including fine-grained details that are vital for reconstruction.

The second discrepancy arises from autoregressive suffix generation. For a prefix depth k, only the prefix scales $T_{\leq k}$ are transmitted, while the omitted suffix scales are autoregressively synthesized as $\hat{T}_{>k}$ at the decoder. Aggregating the decoded prefix and generated suffix yields the reconstructed latent

$$\hat{h}_q^{(k)} = \mathcal{S}\left(\hat{\mathcal{T}}^{(k)}\right) = \mathcal{S}\left(T_{\leq k}, \hat{T}_{>k}\right). \tag{5}$$

Since the generated suffix may deviate from its ground-truth counterpart, $\hat{h}_q^{(k)}$ can differ from the quantized latent h_q reconstructed from the complete ground-truth token sequence. We define this discrepancy as the *generation residual*:

$$r_g^{(k)} = h_q - \hat{h}_q^{(k)} = \mathcal{S}(\mathcal{T}) - \mathcal{S}\left(\hat{\mathcal{T}}^{(k)}\right). \tag{6}$$

This generation residual captures deviations introduced by autoregressive suffix generation and may affect both local details and image structures. Overall, the discrepancy between the continuous latent h and the autoregressively reconstructed latent $\hat{h}_q^{(k)}$ decomposes into the two residuals:

$$h - \hat{h}_q^{(k)} = (h - h_q) + (h_q - \hat{h}_q^{(k)}) = r_q + r_g^{(k)}. \tag{7}$$

This decomposition motivates ResARC to compensate for the two residuals with dedicated strategies: generating the quantization residual from decoder-side context and explicitly transmitting a compact correction for the generation residual.

4.2 Quantization Residual Generation

The *quantization residual* r_q represents information lost when the continuous latent is quantized into discrete multi-scale token maps. The reconstructed tokens and autoregressive decoding features available at the decoder provide rich cues for estimating this residual. We therefore generate the residual at the decoder without transmitting additional bits. To this end, we introduce the **Quantization Residual Generator**, implemented as a conditional Diffusion Transformer (DiT) (Peebles & Xie, 2023). As illustrated in Figure 3, the DiT conditioned on hidden features produced during autoregressive decoding iteratively generates the quantization residual from Gaussian noise. The

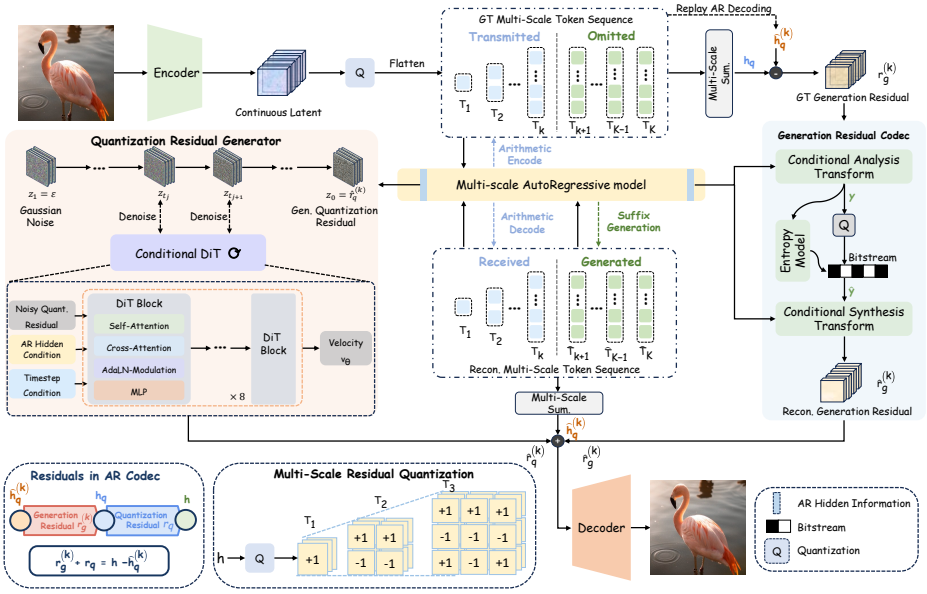

Figure 3: **Overview of the ResARC architecture.** ResARC decomposes the discrepancy in the autoregressive codec into a quantization residual and a generation residual. The quantization residual is synthesized at the decoder without additional bits, while the generation residual is compressed and transmitted for decoder-side correction.

autoregressive features are injected through cross-attention, while the flow matching timestep modulates the DiT blocks via Adaptive LayerNorm (AdaLN).

To learn this conditional generation process, we train the conditional DiT using flow matching (Lipman et al., 2023; Liu et al., 2023). Let $z_0 = r_q$ denote the target residual and $z_1 = \epsilon$ denote Gaussian noise, where $\epsilon \sim \mathcal{N}(0, I)$. For $t \sim \mathcal{U}(0, 1)$, we define the linear interpolation $z_t = (1-t)z_0 + tz_1$ whose target velocity is $v = z_1 - z_0$; the training objective is

$$\mathcal{L}_{\text{FM}} = \mathbb{E}\Big[\|v_\theta(z_t, t; c_{\text{quant}}) - v\|_2^2\Big]. \tag{8}$$

Here, c_{quant} denotes the autoregressive decoding features obtained at prefix depth k. At inference, starting from Gaussian noise at $t = 1$, we integrate the learned velocity field toward $t = 0$ to obtain the predicted quantization residual $\hat{r}_q^{(k)} = z_0$.

Since the generator relies only on autoregressive features already available in the decoder, quantization residual generation does not require additional transmitted payload.

4.3 Generation Residual Coding

The *generation residual* captures the mismatch between the generated suffix and the ground-truth token sequence. The decoder can reproduce the generated suffix, but the corresponding ground-truth suffix tokens are only available at the encoder. We therefore compute the generation residual at the encoder and transmit a compact correction for decoder-side recovery.

Specifically, as illustrated in Figure 3, the encoder replays the same autoregressive suffix-generation process as the decoder to reproduce $\hat{h}_q^{(k)}$, and computes the generation residual $r_g^{(k)}$ according to Equation (6). We compress $r_g^{(k)}$ using the proposed **Generation Residual Codec**, whose architecture is detailed in Appendix B.

The codec incorporates two complementary conditioning signals. First, we embed the prefix depth k as e_k and obtain a prefix-dependent residual scale s_k and gain g_k. These prefix-dependent parameters adapt the codec to different bitrates, since the statistics of $r_g^{(k)}$ vary with the number of omitted suffix scales. Second, we combine autoregressive decoding features with $\hat{h}_q^{(k)}$ to construct a multi-scale context pyramid $C^{(k)} = \{c_0^{(k)}, c_1^{(k)}, c_2^{(k)}, c_3^{(k)}\}$, which is injected into the corresponding stages of the codec. We first normalize the generation residual $r_g^{(k)}$ by the prefix-dependent scale s_k and feed it into the analysis transform:

$$y = g_a\Big(r_g^{(k)} \oslash s_k; \hat{h}_q^{(k)}, C^{(k)}, e_k\Big), \tag{9}$$

where $\oslash$ denotes element-wise division.

Following prior works (Li et al., 2023; Sheng et al., 2025), we employ a four-pass conditional entropy model parameterized by η. The analysis latent elements are first modulated by the prefix-dependent gain g_k and then partitioned into four sequential coding groups, where the quantization step map is predicted once from $c_3^{(k)}$ and e_k and remains fixed across the four passes. Each pass predicts the conditional mean and Laplace scale using these conditions and the latent values reconstructed in previous passes. The quantized latent is represented by an integer-valued symbol tensor $\ddot{y}$, whose conditional probability mass function p_η is used to estimate the bitrate during training:

$$R_g^{(k)} = \frac{-\log_2 p_\eta(\ddot{y} \mid c_3^{(k)}, e_k)}{HW}. \tag{10}$$

At the decoder, the same causal context is used to reconstruct the quantized latent $\hat{y}$ from $\ddot{y}$. The synthesis transform then maps $\hat{y}$ to the reconstructed generation residual:

$$\hat{r}_g^{(k)} = s_k \odot g_s\left(\hat{y}; C^{(k)}, e_k\right), \tag{11}$$

where $\odot$ denotes element-wise multiplication.

4.4 Residual Fusion and Reconstruction

The two residual branches provide complementary information missing from the autoregressively reconstructed latent $\hat{h}_q^{(k)}$. Following the decomposition in Equation (7), we obtain the compensated latent as

$$\hat{h}_c^{(k)} = \hat{h}_q^{(k)} + \hat{r}_q^{(k)} + \hat{r}_g^{(k)}. \tag{12}$$

The adapted VAE decoder $\tilde{\mathcal{D}}$ then maps the compensated latent to the reconstructed image $\hat{x}^{(k)} = \tilde{\mathcal{D}}\left(\hat{h}_c^{(k)}\right)$.

The two residual branches incur different bitrate costs. The quantization residual is synthesized entirely from information already available at the decoder and therefore requires no additional transmitted bits, whereas the generation residual is entropy encoded and explicitly transmitted. For the image $x \in \mathbb{R}^{H\times W\times 3}$, the total bitrate is calculated from the measured bit lengths as

$$R_{\text{total}}^{(k)} = \frac{B_{\text{text}} + B_{\text{prefix}}^{(k)} + B_g^{(k)}}{HW}, \tag{13}$$

where B_{text}, $B_{\text{prefix}}^{(k)}$, and $B_g^{(k)}$ denote the measured bit lengths of the text condition, the transmitted token prefix, and the generation residual bitstream, respectively. Detailed rate accounting is provided in Section D.

4.5 Training Pipeline

We train ResARC in multiple stages. Detailed training settings are provided in Section C.

- **Backbone fine-tuning for compression.** We first fine-tune the Infinity backbone (Han et al., 2025) on our compression training data to adapt the autoregressive model and VAE decoder from their original generative setting to the target compression setting. To adapt the VAE decoder for quantization residual-compensated latents, we train it with the quantized latent h_q and the continuous latent h with equal probability, rather than only using h_q as in the original backbone. The adapted backbone is then frozen for subsequent residual learning.
- **Quantization residual generator training.** We next train the Quantization Residual Generator using the flow-matching objective in Equation (8), together with perceptual supervision $\mathcal{L}_{\text{LPIPS}}$ (Zhang et al., 2018) to improve recovery of fine-grained details.
- **Generation residual codec training.** We then optimize the Generation Residual Codec with a rate-distortion objective that balances the residual bitrate and reconstruction fidelity:

$$\mathcal{L}_g = \lambda_R R_g^{(k)} + \lambda_{\text{res}}\mathcal{L}_{\text{res}} + \lambda_{\text{D}}\mathcal{L}_{\text{DISTS}}, \tag{14}$$

  where $\mathcal{L}_{\text{res}}$ is the latent-space Smooth L1 loss, $\mathcal{L}_{\text{DISTS}}$ is the image-space perceptual DISTS loss (Ding et al., 2022), and λ_R, λ_{res}, and λ_{D} weight the bitrate and reconstruction terms.
- **Residual-aware VAE decoder adaptation.** Finally, we freeze both residual branches and adapt only the VAE decoder using the compensated latent $\hat{h}_c^{(k)}$. This stage adapts the decoder to the latent distribution produced after residual compensation and improves its utilization of the recovered residual information.

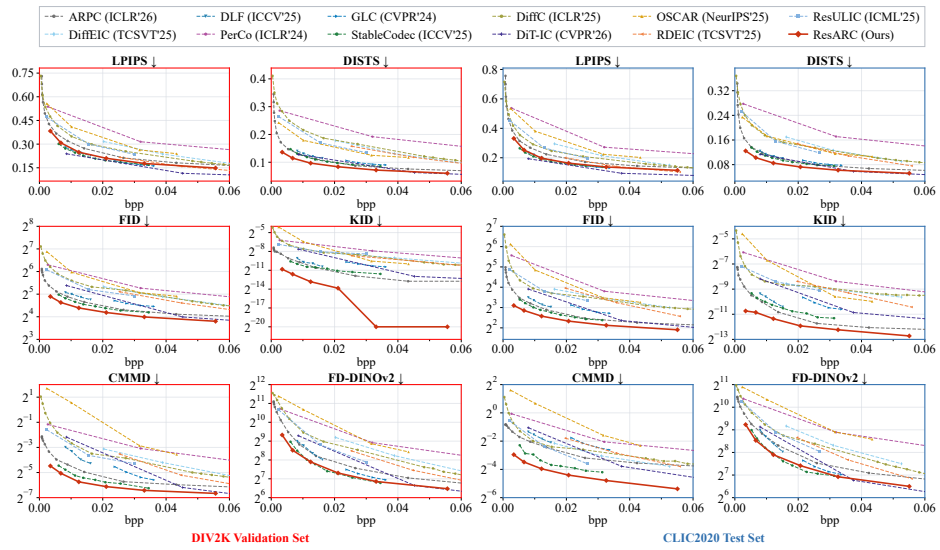

Figure 4: **Rate-perception comparison on DIV2K (left) and CLIC2020 (right).** ResARC achieves strong overall performance across perceptual similarity and distributional fidelity metrics in the ultra-low bitrate regime.

5 Experiments

5.1 Implementation

Datasets. We train on 1,000,000 filtered text-image pairs from COYO-700M (Byeon et al., 2022). We remove low-resolution, unsafe, watermarked, low-aesthetic, and text-heavy samples. Autoregressive and residual learning use 1024×1024 crops, while the initial VAE decoder fine-tuning uses 512×512 crops. For evaluation, we use the DIV2K validation set (Agustsson & Timofte, 2017) with 100 images and the CLIC2020 test set (Toderici et al., 2020) with 428 images. Following the preprocessing protocol of ARPC (Zhang et al., 2026b), all evaluation images are center-cropped to 1024×1024. We generate a caption for each evaluation image using Florence-2 (Xiao et al., 2024).

Training details. We initialize the VAE and autoregressive model from Infinity-2B (Han et al., 2025), with the VAE encoder frozen throughout training. The autoregressive transformer is fine-tuned for 2,000 iterations with a batch size of 64 and a learning rate of 6×10^{-5}. We fine-tune the VAE decoder for 35,000 iterations by sampling continuous and quantized latents with equal probability. The Quantization Residual Generator and Generation Residual Codec are then trained for 5,000 and 7,300 iterations, respectively. Finally, with both residual branches frozen, we adapt the VAE decoder to the compensated latents for another 1,000 iterations. All stages use AdamW (Loshchilov & Hutter, 2019). The training details can be found in Section C.

Metrics. We focus on perceptual quality in the ultra-low bitrate regime from two complementary perspectives. For paired perceptual similarity, we report LPIPS (Zhang et al., 2018) and DISTS (Ding et al., 2022), which compare each reconstruction with its corresponding original image in learned perceptual feature spaces, with DISTS further emphasizing structural and textural similarity. For distributional fidelity, we report FID (Heusel et al., 2017), KID (Bińkowski et al., 2018), CMMD (Jayasumana et al., 2024), and FD-DINOv2 (Stein et al., 2023; Oquab et al., 2023), which measure the discrepancy between the distributions of reconstructed and original images in complementary feature spaces. Lower values indicate better performance for all six metrics.

Baselines. We compare ResARC with eleven leading open-source generative image codecs at their released bitrate points. These baselines include the autoregressive codec ARPC (Zhang et al., 2026b); diffusion-based codecs DiffEIC (Li et al., 2024), DiffC (Vonderfecht & Liu, 2025), DiT-IC (Shi et al., 2026), OSCAR (Guo et al., 2025), PerCo (Careil et al., 2024), RDEIC (Li et al., 2025), ResULIC (Ke et al., 2025), and StableCodec (Zhang et al., 2025); and generative latent codecs DLF (Xue et al., 2025a) and GLC (Jia et al., 2024). For all methods, the reported bitrate includes all transmitted components specified by the corresponding codec.

5.2 Quantitative Comparisons

Figure 4 compares ResARC with leading generative codecs on the DIV2K validation set and CLIC2020 test set. Across both datasets, ResARC achieves a favorable rate-perception trade-off throughout the ultra-low bitrate regime. Compared with the autoregressive codec ARPC, ResARC consistently improves all six evaluated metrics over the evaluated bitrate range. Across all com-

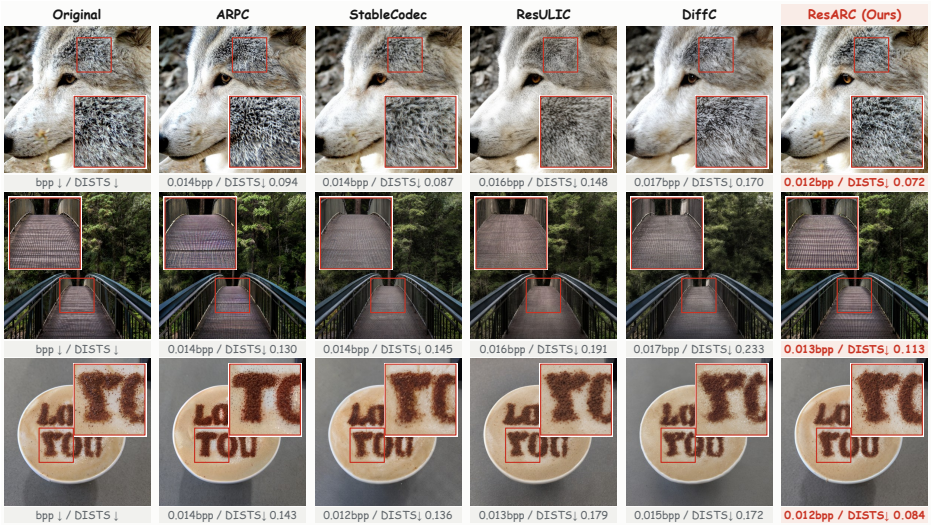

Figure 5: **Qualitative comparison with other generative codecs on DIV2K and CLIC2020.**

pared methods, ResARC achieves the strongest overall performance under DISTS and the distributional fidelity metrics FID, KID, CMMD, and FD-DINOv2, while remaining competitive under LPIPS. These results demonstrate that compensating for both quantization and generation residuals substantially improves the reconstruction fidelity of autoregressive generative compression. The complete signed BD-rate comparison is provided in Appendix G.2, further validating the consistent improvements of ResARC across metrics and datasets.

5.3 Qualitative Comparisons

Figure 5 compares ResARC with leading generative codecs at similar ultra-low bitrates. The autoregressive codec ARPC (Zhang et al., 2026b) produces rich textures but may alter image structures, such as fur patterns, repeated lines on the bridge deck, and lettering on the coffee surface, likely due to imperfect suffix generation. In contrast, diffusion-based methods such as StableCodec (Zhang et al., 2025) and ResULIC (Ke et al., 2025) tend to smooth these local details, reducing their similarity to the original images. ResARC produces less over-smoothed reconstructions and better preserves fine details and image structures, including the arrangement of wolf fur, repeated bridge-deck patterns, and the shape of the lettering. These qualitative results are consistent with our residual-aware design, where quantization residual generation helps recover fine-grained details, while generation residual correction helps preserve image structures and textures. Additional qualitative comparisons are provided in Section H.2.

5.4 Ablation Studies

We ablate three aspects: the complementary contributions of the two residuals, the staged training pipeline, and the compensation strategy for each residual. All experiments are evaluated on the DIV2K validation set. Tables report signed BD-rate (%) relative to the specified reference, where lower values are better. Additional ablations are provided in Section F.

Contribution of the two residuals. Starting from the autoregressive baseline, we separately add the generation residual branch and the quantization residual branch, and then combine both in full ResARC. As shown in Table 1, either residual branch improves rate-perception performance over the autoregressive baseline, while combining both yields the best overall performance. Specifically, with only the generation residual branch, the DISTS and FID BD-rates decrease from $+9.80\%$ and $+26.57\%$ to $+1.90\%$ and $+8.03\%$, respectively; with only the quantization residual branch, they decrease to $+5.38\%$ and $+23.99\%$. Full ResARC further improves all four metrics, demonstrating the complementary benefits of the two residual branches. Additional rate-perception curves and qualitative results which further illustrate the contributions of the two residuals are provided in Section F.1.

Fine-tuning the autoregressive prior. To isolate the effect of adapting the autoregressive prior to the compression task, we compare the pretrained Infinity-2B transformer (Han et al., 2025) with our model fine-tuned on the compression training data. As shown in Table 2, fine-tuning consistently improves all four metrics on DIV2K, yielding BD-rate reductions of 6.42% under DISTS and 11.29% under FID. These results show that adapting the autoregressive prior provides a stronger foundation for subsequent training of the residual branches.

Table 1: **Ablation study on the contribution of the quantization and generation residuals.**

Configuration	r_q	r_g	LPIPS↓	DISTS↓	FID↓	KID↓
Autoregressive Baseline	✗	✗	+7.78	+9.80	+26.57	+340.22
w/ Generation Residual Branch	✗	✓	+2.42	+1.90	+8.03	+67.91
w/ Quantization Residual Branch	✓	✗	+3.46	+5.38	+23.99	+263.50
ResARC (Both Residual Branches)	✓	✓	**0.00**	**0.00**	**0.00**	**0.00**

Table 2: **Effect of fine-tuning the AR prior.**

Configuration	LPIPS↓	DISTS↓	FID↓	KID↓
w/o AR fine-tuning	0.00	0.00	0.00	0.00
w/ AR fine-tuning	**-4.56**	**-6.42**	**-11.29**	**-29.67**

Table 3: **Effect of VAE decoder adaptation.**

Configuration	LPIPS↓	DISTS↓	FID↓	KID↓
w/o Adapt VAE Dec.	0.00	0.00	0.00	0.00
w/ Adapt VAE Dec.	**-4.31**	**-7.19**	**-4.24**	**-24.20**

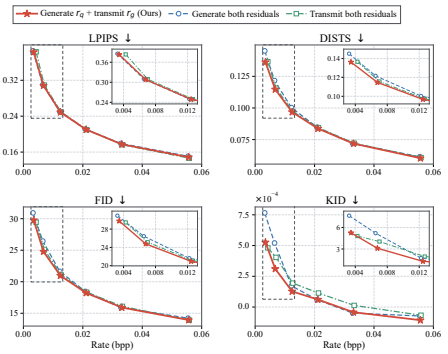

Figure 6: **Rate-perception curves for residual compensation strategies.**

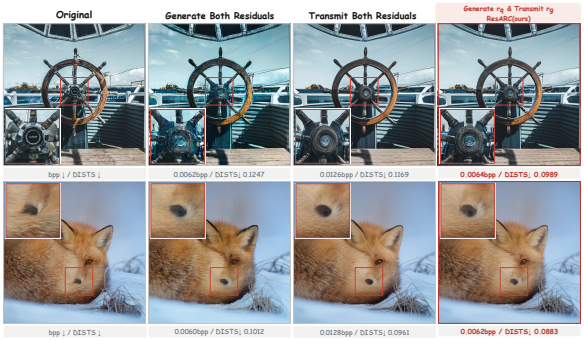

Figure 7: **Visual comparison of residual compensation strategies.**

Residual-aware VAE decoder adaptation. We evaluate the effect of adapting the VAE decoder to compensated latents by comparing perceptual quality before and after adaptation. As shown in Table 3, the residual-aware adaptation improves all four metrics on DIV2K, including BD-rate reductions of 4.31% under LPIPS and 7.19% under DISTS. These results demonstrate that adapting the VAE decoder to the compensated latent improves perceptual reconstruction quality.

Residual compensation strategies. We compare our residual-specific strategy, which generates the quantization residual r_q with a conditional DiT and transmits the generation residual r_g with a learned codec, against two uniform strategies that either generate or transmit both residuals. As shown in Figure 6, transmitting both residuals yields consistently worse rate-perception trade-offs across all four metrics, while generating both slightly improves LPIPS but degrades DISTS, FID, and KID. These results support the residual-specific design of ResARC: the quantization residual can be effectively inferred from decoder-side autoregressive context without additional transmitted bits, whereas the generation residual depends on source suffix information unavailable at the decoder and therefore benefits from explicit transmission. The corresponding BD-rate comparison is reported in Appendix F.2.

We further provide a qualitative comparison of the three strategies in Figure 7. ResARC better preserves local structures and fine details, such as the fox's nose and the central hub of the ship's wheel, further supporting the effectiveness of our residual-specific compensation design.

6 Conclusion

We identify two residuals inherent to autoregressive generative compression: the *quantization residual* introduced by discrete tokenization and the *generation residual* arising from imperfect autoregressive suffix generation. To recover both residuals, we introduce **ResARC**, the first residual-aware autoregressive codec that generates the quantization residual with a Quantization Residual Generator based on DiT and compresses the generation residual with a dedicated Generation Residual Codec for transmission. Extensive experiments demonstrate the effectiveness of this residual-aware design, with ResARC achieving strong rate-perception performance compared with leading generative codecs. These results highlight explicit modeling and compensation of codec residuals as an effective design principle for ultra-low bitrate generative image compression.

USE OF LARGE LANGUAGE MODELS

We used Qwen3.8 Max and GPT-6 Astra solely to refine the language in the method and experimental sections, focusing on clarity, grammar, and academic style. All technical content, including the conceptual formulation and experimental work, is our own. Their assistance was confined to improving the wording and stylistic consistency of the text.

REFERENCES

Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW)*, pp. 1122–1131. IEEE, 2017.

Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 221–231, 2019.

Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rJxdQ3jeg.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkcQFMZRb.

Fabrice Bellard. Bpg image format. https://bellard.org/bpg/, 2018.

Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1lUOzWCW.

Minwoo Byeon, Beomhee Park, Haecheon Kim, Sungjun Lee, Woonhyuk Baek, and Saehoon Kim. COYO-700M: Image–text pair dataset. GitHub repository, 2022. URL https://github.com/kakaobrain/coyo-dataset.

Marlene Careil, Matthew J. Muckley, Jakob Verbeek, and Stéphane Lathuilière. Towards image compression with perfect realism at ultra-low bitrates. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ktdETU9JBg.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9630–9640. IEEE, 2021.

Pierre Charbonnier, Laure Blanc-Feraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st international conference on image processing*, volume 2, pp. 168–172. IEEE, 1994.

Yunuo Chen, Chuqin Zhou, Jiangchuan Li, Xiaoyue Ling, Bing He, Jincheng Dai, Li Song, and Guo Lu. Next-frame decoding for ultra-low-bitrate image compression with video diffusion priors. *arXiv preprint arXiv:2603.15129*, 2026.

Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. PaddleOCR 3.0 technical report. *arXiv preprint arXiv:2507.05595*, 2025. URL https://arxiv.org/abs/2507.05595.

Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2022.

Jarek Duda. Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540*, 2013.

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.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Jinpei Guo, Yifei Ji, Zheng Chen, Kai Liu, Min Liu, Wang Rao, Wenbo Li, Yong Guo, and Yulun Zhang. Oscar: One-step diffusion codec across multiple bit-rates. *Advances in Neural Information Processing Systems*, 38:85267–85286, 2025.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15733–15744. IEEE, 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. *Advances in neural information processing systems*, 30, 2017.

Emiel Hoogeboom, Eirikur Agustsson, Fabian Mentzer, Luca Versari, George Toderici, and Lucas Theis. High-fidelity image compression with score-based generative models. *arXiv preprint arXiv:2305.18231*, 2023.

Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9307–9315. IEEE, 2024.

Zhaoyang Jia, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Generative latent coding for ultra-low bitrate image compression. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26088–26098. IEEE, 2024.

Zhaoyang Jia, Naifu Xue, Zihan Zheng, Jiahao Li, Bin Li, Xiaoyi Zhang, Zongyu Guo, Yuan Zhang, Houqiang Li, and Yan Lu. Cod-lite: Real-time diffusion-based generative image compression. *arXiv preprint arXiv:2604.12525*, 2026a.

Zhaoyang Jia, Zihan Zheng, Naifu Xue, Jiahao Li, Bin Li, Zongyu Guo, Xiaoyi Zhang, Houqiang Li, and Yan Lu. Cod: A diffusion foundation model for image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38420–38429, 2026b.

Shiyin Jiang, Wei Long, Minghao Han, Zhenghao Chen, Ce Zhu, and Shuhang Gu. Differentiable vector quantization for rate-distortion optimization of generative image compression. *arXiv preprint arXiv:2604.10546*, 2026.

Anle Ke, Xu Zhang, Tong Chen, Ming Lu, Chao Zhou, Jiawen Gu, and Zhan Ma. Ultra lowrate image compression with semantic residual coding and compression-aware diffusion. *arXiv preprint arXiv:2505.08281*, 2025.

Nikolai Körber, Eduard Kromer, Andreas Siebert, Sascha Hauke, Daniel Mueller-Gritschneder, and Björn Schuller. Perco (SD): Open perceptual compression. In *Workshop on Machine Learning and Compression, NeurIPS 2024*, 2024. URL https://openreview.net/forum?id=8xvygfdRWy.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11513–11522. IEEE, 2022.

Daowen Li, Ruixiao Dong, Ying Chen, Kai Li, Ding Ding, and Li Li. Progvc: Progressive-based generative video compression via auto-regressive context modeling. *arXiv preprint arXiv:2603.17546*, 2026.

Jiahao Li, Bin Li, and Yan Lu. Neural video compression with diverse contexts. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22616–22626. IEEE, 2023.

Zhiyuan Li, Yanhui Zhou, Hao Wei, Chenyang Ge, and Jingwen Jiang. Toward extreme image compression with latent feature guidance and diffusion prior. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(1):888–899, 2024.

Zhiyuan Li, Yanhui Zhou, Hao Wei, Chenyang Ge, and Ajmal Mian. Rdeic: Accelerating diffusion-based extreme image compression with relay residual diffusion. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(11):11540–11552, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Qi Mao, Tinghan Yang, Yinuo Zhang, Zijian Wang, Meng Wang, Shiqi Wang, Libiao Jin, and Siwei Ma. Extreme image compression using fine-tuned vqgans. In *2024 Data Compression Conference (DCC)*, pp. 203–212. IEEE, 2024.

Fabian Mentzer, George D Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression. *Advances in neural information processing systems*, 33:11913–11924, 2020.

Matthew J Muckley, Alaaeldin El-Nouby, Karen Ullrich, Hervé Jégou, and Jakob Verbeek. Improving statistical fidelity for neural image compression with implicit local likelihood models. In *International Conference on Machine Learning*, pp. 25426–25443. PMLR, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Huu-Tai Phung, Yu-Hsiang Lin, Yen-Kuan Ho, and Wen-Hsiao Peng. Exploring autoregressive vision foundation models for image compression. In *2025 Picture Coding Symposium (PCS)*, pp. 1–5. IEEE, 2025.

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 *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Lucas Relic, Roberto Azevedo, Markus Gross, and Christopher Schroers. Lossy image compression with foundation diffusion models. In *European Conference on Computer Vision*, pp. 303–319. Springer, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Xihua Sheng, Li Li, Dong Liu, and Shiqi Wang. Bi-directional deep contextual video compression. *IEEE Transactions on Multimedia*, 27:5632–5646, 2025.

Xihua Sheng, Lingyu Zhu, Tianyu Zhang, Dong Liu, Shiqi Wang, and Jing Wang. Cadc: Content adaptive diffusion-based generative image compression. *arXiv preprint arXiv:2602.21591*, 2026.

Junqi Shi, Ming Lu, Xingchen Li, Anle Ke, Ruiqi Zhang, and Zhan Ma. Dit-ic: Aligned diffusion transformer for efficient image compression. *arXiv preprint arXiv:2603.13162*, 2026.

George Stein, Jesse Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L Caterini, Eric Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. *Advances in Neural Information Processing Systems*, 36:3732–3784, 2023.

Haotian Tang, Yecheng Wu, Shang Yang, Enze Xie, Junsong Chen, Junyu Chen, Zhuoyang Zhang, Han Cai, Yao Lu, and Song Han. HART: Efficient visual generation with hybrid autoregressive transformer. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=q5sOv4xQe4.

Lucas Theis, Tim Salimans, Matthew D Hoffman, and Fabian Mentzer. Lossy compression with gaussian diffusion. *arXiv preprint arXiv:2206.08889*, 2022.

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.

George Toderici, Wenzhe Shi, Radu Timofte, Lucas Theis, Johannes Balle, Eirikur Agustsson, Nick Johnston, and Fabian Mentzer. Workshop and challenge on learned image compression (clic2020). In *CVPR*, 2020.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Jeremy Vonderfecht and Feng Liu. Lossy compression with pretrained diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=raUnLe0Z04.

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *Proceedings of the Thirty-Seventh Asilomar Conference on Signals, Systems and Computers*, volume 2, pp. 1398–1402. Ieee, 2003.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4818–4829. IEEE, 2024.

Naifu Xue, Qi Mao, Zijian Wang, Yuan Zhang, and Siwei Ma. Unifying generation and compression: Ultra-low bitrate image coding via multi-stage transformer. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2024.

Naifu Xue, Zhaoyang Jia, Jiahao Li, Bin Li, Yuan Zhang, and Yan Lu. Dlf: Extreme image compression with dual-generative latent fusion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19227–19236. IEEE, 2025a.

Naifu Xue, Zhaoyang Jia, Jiahao Li, Bin Li, Yuan Zhang, and Yan Lu. One-step diffusion-based image compression with semantic distillation. *Advances in neural information processing systems*, 38:37108–37144, 2025b.

Ruihan Yang and Stephan Mandt. Lossy image compression with conditional diffusion models. *Advances in Neural Information Processing Systems*, 36:64971–64995, 2023.

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 *International Conference on Learning Representations*, volume 2024, pp. 765–783, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Tianyu Zhang, Xin Luo, Li Li, and Dong Liu. Stablecodec: Taming one-step diffusion for extreme image compression. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17379–17389. IEEE, 2025.

Tianyu Zhang, Dong Liu, and Chang Wen Chen. Ultra-low bitrate perceptual image compression with shallow encoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12118–12128, 2026a.

Ziyuan Zhang, Yichong Xia, Bin Chen, Tianwei Zhang, Hao Wang, and Han Qiu. Autoregressive-based progressive coding for ultra-low bitrate image compression. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. In *International Conference on Learning Representations*, volume 2025, pp. 90844–90868, 2025.

A PSEUDO-CODE OF RESARC

Algorithm 1 and Algorithm 2 summarize the ResARC encoding-decoding pipeline and quantization residual generation, respectively.

A.1 ENCODING AND DECODING

During encoding, the encoder reproduces the same autoregressive suffix generation process used at the decoder from the transmitted prefix $T_{\leq k}$ and text condition ℓ. This reconstructs $\hat{h}_q^{(k)}$, from which the generation residual $r_g^{(k)} = h_q - \hat{h}_q^{(k)}$ is computed and compressed by the Generation Residual Codec. The resulting bitstream contains the encoded text condition, transmitted prefix scales, and entropy-coded generation residual.

During decoding, the text condition and prefix scales are first recovered and used to regenerate the same suffix and reconstruct $\hat{h}_q^{(k)}$. The autoregressive decoding process also provides the condition c_{quant} for synthesizing the quantization residual and context $C^{(k)}$ for the generation residual. To ensure identical autoregressive states at the encoder and decoder, GENERATESUFFIX uses the same sampling realization at both endpoints, while residual entropy coding follows the same causal symbol order. Finally, the decoder generates the quantization residual $\hat{r}_q^{(k)}$, decodes the transmitted generation residual $\hat{r}_g^{(k)}$, and combines both with $\hat{h}_q^{(k)}$ to form the compensated latent $\hat{h}_c^{(k)}$. The adapted VAE decoder $\tilde{\mathcal{D}}$ then maps $\hat{h}_c^{(k)}$ to the reconstructed image.

Algorithm 1 ResARC encoding and decoding

Require: Shared models and fixed sampling settings in encoding and decoding

procedure ENCODE(x, ℓ, k)
  $h \leftarrow \mathcal{E}(x); \mathcal{T} \leftarrow \mathcal{Q}(h); h_q \leftarrow \mathcal{S}(\mathcal{T})$
  $\mathcal{B}_{\text{text}} \leftarrow$ ENCODETEXT(ℓ)
  $\mathcal{B}_{\text{prefix}} \leftarrow$ ENCODEPREFIX($T_{\leq k}, \ell; p_\psi$)
  $\hat{T}_{>k} \leftarrow$ GENERATESUFFIX($T_{\leq k}, \ell; p_\psi$)
  $\hat{h}_q^{(k)} \leftarrow \mathcal{S}(T_{\leq k}, \hat{T}_{>k})$
  Construct $C^{(k)}$ from the autoregressive model
  Construct e_k, s_k, g_k from k
  $r_g^{(k)} \leftarrow h_q - \hat{h}_q^{(k)}$
  $y \leftarrow g_a(r_g^{(k)} \oslash s_k; \hat{h}_q^{(k)}, C^{(k)}, e_k)$
  $\ddot{y} \leftarrow \mathcal{Q}_\eta(g_k \odot y \mid c_3^{(k)}, e_k)$
  $\mathcal{B}_g \leftarrow$ ENTROPYENCODE($\ddot{y}; p_\eta(\ddot{y} \mid c_3^{(k)}, e_k)$)
  return $\mathcal{B} \leftarrow$ ASSEMBLE($k, \mathcal{B}_{\text{text}}, \mathcal{B}_{\text{prefix}}, \mathcal{B}_g$)
end procedure

procedure DECODE($\mathcal{B}$)
  $(k, \mathcal{B}_{\text{text}}, \mathcal{B}_{\text{prefix}}, \mathcal{B}_g) \leftarrow$ PARSE($\mathcal{B}$)
  $\ell \leftarrow$ DECODETEXT($\mathcal{B}_{\text{text}}$)
  $T_{\leq k} \leftarrow$ DECODEPREFIX($\mathcal{B}_{\text{prefix}}, \ell; p_\psi$)
  $\hat{T}_{>k} \leftarrow$ GENERATESUFFIX($T_{\leq k}, \ell; p_\psi$)
  $\hat{h}_q^{(k)} \leftarrow \mathcal{S}(T_{\leq k}, \hat{T}_{>k})$
  Construct c_{quant} and $C^{(k)}$ from the autoregressive model
  Construct e_k, s_k, g_k from k
  $\ddot{y} \leftarrow$ ENTROPYDECODE($\mathcal{B}_g; p_\eta(\ddot{y} \mid c_3^{(k)}, e_k)$)
  $\hat{u} \leftarrow \mathcal{R}_\eta(\ddot{y} \mid c_3^{(k)}, e_k)$
  $\hat{y} \leftarrow \hat{u} \oslash g_k$
  $\hat{r}_g^{(k)} \leftarrow s_k \odot g_s(\hat{y}; C^{(k)}, e_k)$
  Draw $\epsilon \sim \mathcal{N}(0, I)$
  $\hat{r}_q^{(k)} \leftarrow$ GENERATEQUANTIZATIONRESIDUAL($c_{\text{quant}}, \epsilon, N$)
  $\hat{h}_c^{(k)} \leftarrow \hat{h}_q^{(k)} + \hat{r}_q^{(k)} + \hat{r}_g^{(k)}$
  return $\hat{x}^{(k)} \leftarrow \tilde{\mathcal{D}}(\hat{h}_c^{(k)})$
end procedure

Algorithm 2 Quantization residual generation

Require: Diffusion Transformer v_θ

procedure GENERATEQUANTIZATIONRESIDUAL($c_{\text{quant}}, \epsilon, N$)
  $z \leftarrow \epsilon$
  for $j = 0, \ldots, N-1$ **do**
    $t_j \leftarrow 1 - j/N$; $t_{j+1} \leftarrow 1 - (j+1)/N$
    $\Delta t \leftarrow t_{j+1} - t_j$
    $v \leftarrow v_\theta(z, t_j; c_{\text{quant}})$
    $\tilde{z} \leftarrow z + \Delta t\, v$
    if $j < N-1$ **then**
      $\tilde{v} \leftarrow v_\theta(\tilde{z}, t_{j+1}; c_{\text{quant}})$
      $z \leftarrow z + \frac{\Delta t}{2}(v + \tilde{v})$
    else
      $z \leftarrow \tilde{z}$
    end if
  end for
  return $\hat{r}_q^{(k)} \leftarrow z$
end procedure

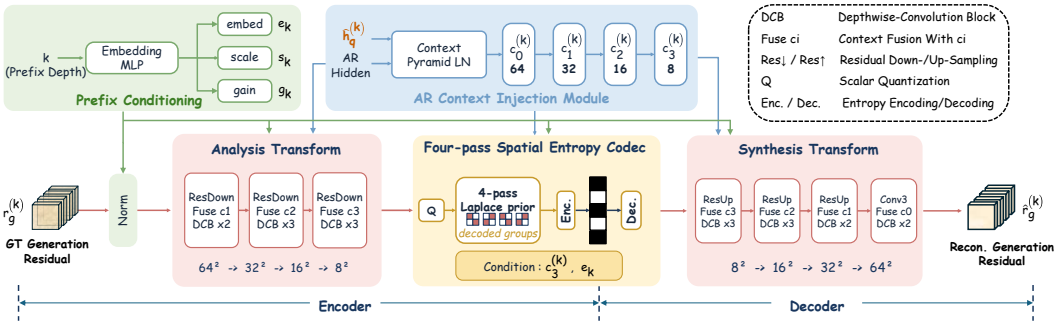

Figure 8: **Architecture of the Generation Residual Codec.** Prefix-depth conditioning and autoregressive context guide the analysis, entropy modeling, and synthesis stages of generation residual coding.

A.2 Quantization Residual Generation

Algorithm 2 summarizes the sampling procedure of the Quantization Residual Generator. Starting from Gaussian noise, we integrate the conditional velocity field from $t = 1$ to $t = 0$ using Heun updates with an Euler update on the final step. At each step, the DiT predicts the velocity conditioned on the autoregressive decoding context c_{quant}, and the final sample z_0 is the predicted quantization residual $\hat{r}_q^{(k)}$.

B Generation Residual Codec

Architecture. Figure 8 illustrates the Generation Residual Codec introduced in Section 4.3. The statistics of the generation residual $r_g^{(k)}$ vary with the transmitted prefix depth k, since different numbers of suffix scales are generated autoregressively. To adapt the codec across different prefix depths, we introduce a prefix-depth embedding e_k, together with a residual scale s_k and latent gain g_k. The embedding e_k conditions the analysis, entropy, and synthesis transforms, while s_k normalizes the generation residual and g_k modulates the codec latent before quantization. In addition, the autoregressively reconstructed latent $\hat{h}_q^{(k)}$ and the hidden features of the autoregressive model provide context for modeling $r_g^{(k)}$. We aggregate these features into a multi-scale context pyramid $C^{(k)}$, which is injected into the corresponding stages of the codec. Conditioned on e_k and $C^{(k)}$, the Generation Residual Codec consists of an analysis transform, a four-pass conditional entropy model, and a synthesis transform for compressing $r_g^{(k)}$ and reconstructing $\hat{r}_g^{(k)}$.

Four-pass entropy model. Following prior works (Li et al., 2023; Sheng et al., 2025), we adopt a four-pass channel-spatial entropy model for the analysis latent y. The channels of y are first divided into four equal groups. Within each channel group, spatial positions are further partitioned according to the row-column parity masks M_0, M_1, M_2, M_3, corresponding to $(0,0)$, $(0,1)$, $(1,0)$, and $(1,1)$ under zero-based coordinates. The four groups follow different mask orders across the four passes:

Pass	Group 0	Group 1	Group 2	Group 3
1	M_0	M_1	M_2	M_3
2	M_3	M_2	M_1	M_0
3	M_2	M_3	M_0	M_1
4	M_1	M_0	M_3	M_2

This schedule assigns each latent element to exactly one coding pass while allowing all elements within the same pass to be processed in parallel.

Before quantization, the analysis latent y is modulated by the prefix-dependent gain g_k. A conditional prior predicts a positive quantization-step map q, together with the initial mean μ and Laplace scale σ, from the autoregressive context $c_3^{(k)}$ and prefix embedding e_k. The quantization-step map q remains fixed across the four passes. For later passes, the mean and scale parameters μ, σ are further updated using the latent values reconstructed in previous passes, while positions that have not yet been decoded are masked to zero.

These predicted parameters determine the quantization and reconstruction of each latent element. Let $\mathcal{I}_j$ denote the index set of elements in y assigned to coding pass j. For $i \in \mathcal{I}_j$, the integer symbol $\ddot{y}_i$ and corresponding reconstructed latent value $\hat{y}_i$ are given by

$$\ddot{y}_i = Q\left(\frac{g_{k,i} y_i}{q_i} - \mu_{j,i}\right), \qquad \hat{y}_i = \frac{q_i}{g_{k,i}} \left(\ddot{y}_i + \mu_{j,i}\right), \tag{15}$$

where $Q(v) = \mathrm{round}(v)$ denotes the quantizer, $g_{k,i}$ denotes the i-th element of the prefix-dependent gain g_k, and $\mu_{j,i}$ denotes the conditional mean predicted for pass j. The resulting integer-valued tensor $\ddot{y}$ is entropy-coded for transmission. Causal context updates use the reconstructed normalized values $\ddot{y}_i + \mu_{j,i}$, whereas the rescaled latent $\hat{y}$ is passed to the synthesis transform.

For entropy modeling, each symbol is modeled with a discretized zero-mean Laplace distribution centered by the predicted conditional mean μ. For element i coded in pass j, the probability mass assigned to integer symbol n is

$$P_{j,i}(n) = F_{\sigma_{j,i}}\left(n + \tfrac{1}{2}\right) - F_{\sigma_{j,i}}\left(n - \tfrac{1}{2}\right), \tag{16}$$

where F_σ denotes the CDF of a zero-mean Laplace distribution with predicted scale $\sigma_{j,i}$.

The conditional probability of the complete symbol tensor $\ddot{y}$ is then factorized according to the four-pass causal coding order:

$$p_\eta\left(\ddot{y} \mid c_3^{(k)}, e_k\right) = \prod_{j=1}^{4} \prod_{i \in \mathcal{I}_j} P_{j,i}(\ddot{y}_i), \tag{17}$$

where the parameters of $P_{j,i}$ for each pass are conditioned on $c_3^{(k)}$, e_k, and the symbols reconstructed in previous passes. Accordingly, the generation-residual bitrate during training is estimated as

$$R_g^{(k)} = \frac{-\log_2 p_\eta\left(\ddot{y} \mid c_3^{(k)}, e_k\right)}{HW}. \tag{18}$$

At the decoder, the same pass order and causal context updates are used to recover $\hat{y}$ from the decoded symbols $\ddot{y}$. The synthesis transform then reconstructs the generation residual as

$$\hat{r}_g^{(k)} = s_k \odot g_s\left(\hat{y}; C^{(k)}, e_k\right). \tag{19}$$

Training overview. Beyond latent-space residual reconstruction, we also optimize the perceptual quality of the reconstructed image. Specifically, we combine $\hat{r}_g^{(k)}$ with the autoregressively reconstructed latent $\hat{h}_q^{(k)}$ and decode

$$\hat{x}_g^{(k)} = \mathcal{D}_0\left(\hat{h}_q^{(k)} + \hat{r}_g^{(k)}\right), \tag{20}$$

which is used for image-space perceptual supervision. Here, $\mathcal{D}_0$ denotes the VAE decoder obtained by mixed-latent fine-tuning and kept frozen during residual-branch training. Based on both latent-space and image-space supervision, we train the codec in three stages: latent-space rate-distortion training, image-space perceptual refinement, and joint fine-tuning. Training details for the Generation Residual Codec are provided in Section C.

C Training Details

We train ResARC in stages: first fine-tuning the backbone, then optimizing the two residual branches separately, and finally adapting the VAE decoder to the compensated latent representation. The VAE encoder remains frozen throughout the entire training pipeline. Table 4 summarizes the optimization settings for all stages, which are detailed in the following subsections.

Training data. We filter COYO-700M (Byeon et al., 2022) to retain images with a shorter side greater than $1,024$ pixels, a LAION aesthetic score of at least 5.0, and a watermark score no greater than 0.1. We further remove samples rejected by caption or domain filters, as well as images whose available NSFW score from either OpenNSFW2 or GantMan exceeds 0.1. Using PP-OCRv5 mobile text detection (Cui et al., 2025) with a confidence threshold of 0.5, we discard images if the total detected text area exceeds 5% of the image, any single text region exceeds 3.5%, or at least 20 text regions are detected.

Selecting prefix depth. For residual-branch training and residual-aware VAE decoder adaptation, we use prefix depths $k \in \{5, 6, 7, 8, 9, 10\}$. We follow a balanced deterministic schedule over prefix depths across training steps.

C.1 Autoregressive Prior Fine-Tuning

The Infinity-2B transformer (Han et al., 2025) is fine-tuned for 2,000 iterations on 1024×1024 images using AdamW (Loshchilov & Hutter, 2019), with a global batch size of 64 and a learning rate of 6×10^{-5}. We optimize the autoregressive model with the cross-entropy objective

$$\mathcal{L}_{\mathrm{AR}} = -\frac{1}{|\Omega|} \sum_{(i,d)\in\Omega} \left[b_{i,d} \log p_{i,d} + (1 - b_{i,d}) \log(1 - p_{i,d}) \right], \tag{21}$$

where Ω indexes valid token dimensions and $p_{i,d}$ denotes the predicted probability associated with $b_{i,d}$, the d-th dimension of token i, conditioned on the preceding scales and text. The fine-tuned autoregressive model is then frozen for all subsequent training stages.

C.2 Mixed-Latent VAE Decoder Fine-Tuning

We further fine-tune the VAE decoder on our compression training data. Recovering the quantization residual shifts the decoder input from the quantized latent h_q toward the continuous latent h. To adapt the decoder to this shift, we expose it to both latent distributions during training. Specifically, for each image, we sample $z = h$ with probability 0.5 and $z = h_q$ otherwise, and reconstruct $x_0 = \mathcal{D}_0(z)$. This balanced sampling improves the decoder's compatibility with the compensated latents used in ResARC without over-specializing to either latent distribution. We then optimize the VAE decoder under this mixed-latent setting for $35,000$ steps at a resolution of 512×512 using the following objective:

$$\mathcal{L}_{\mathrm{rec}} = \mathcal{L}_{\mathrm{MSE}}(x_0, x) + 2\mathcal{L}_{\mathrm{LPIPS}}(x_0, x), \tag{22}$$

$$\mathcal{L}_{\mathrm{ft-dec}} = \mathcal{L}_{\mathrm{rec}} + \lambda_{\mathrm{adv}} \mathcal{L}_{\mathrm{adv}}^{G}, \qquad \mathcal{L}_{\mathrm{adv}}^{G} = -\mathbb{E}[\mathcal{F}(x_0)], \tag{23}$$

where $\mathcal{L}_{\text{LPIPS}}$ denotes the LPIPS-VGG loss (Zhang et al., 2018), and $\mathcal{L}_{\text{adv}}^{G}$ is the adversarial loss enabled after $10,000$ steps. The discriminator $\mathcal{F}$ consists of a frozen DINO feature extractor (Caron et al., 2021) and trainable discriminator heads. Following the adaptive weighting strategy used in perceptual autoencoder training (Rombach et al., 2022), we set the adversarial weight λ_{adv} as

$$\lambda_{\text{adv}} = 0.5 \ \text{sg}\left[\text{clip}_{[0,10^4]}\left(\frac{\|\nabla_W \mathcal{L}_{\text{rec}}\|_2}{\|\nabla_W \mathcal{L}_{\text{adv}}^{G}\|_2 + 10^{-4}}\right)\right], \tag{24}$$

where W denotes the weights of the final convolutional layer of the decoder and $\text{sg}(\cdot)$ denotes the stop-gradient operator. Starting from step $10,000$, the discriminator heads are jointly optimized using the hinge loss

$$\mathcal{L}_{\mathcal{F}} = \tfrac{1}{2}\mathbb{E}\left[\max(0, 1 - \mathcal{F}(x)) + \max(0, 1 + \mathcal{F}(\text{sg}(x_0)))\right]. \tag{25}$$

The discriminator heads are optimized with AdamW using a learning rate of 5×10^{-5}, $\beta = (0.5, 0.9)$, and zero weight decay.

C.3 Quantization Residual Generator Training

We train the Quantization Residual Generator in two stages: flow-matching pretraining and image-space perceptual refinement. The conditional DiT consists of 8 blocks with width 640, 8 attention heads, and a latent patch size of 2×2. It predicts 32 residual channels conditioned on a 2048-channel autoregressive decoding feature c_{quant} from the final-scale hidden features of the penultimate layer of the autoregressive model, without condition dropout or additional text conditioning. A fixed normalization is applied to the quantization residual before training and inverted after sampling.

Flow-matching pretraining. We first train the conditional DiT for $3,000$ steps using the flow-matching objective defined in Section 4.2. The timestep is clipped to $[10^{-5}, 1 - 10^{-5}]$ for numerical stability.

Image-space perceptual refinement. Flow-matching pretraining directly supervises the latent residual but does not optimize for the perceptual quality of the final reconstruction. We therefore further train the generator for $2,000$ steps with image-space perceptual supervision. Specifically, with $\hat{x}_q^{(k)} = \mathcal{D}_0(\hat{h}_q^{(k)} + \hat{r}_q^{(k)})$, we optimize the generator using the training objective

$$\mathcal{L}_q^{\text{image}} = 0.25\mathcal{L}_{\text{FM}} + \lambda_{\text{LPIPS}}(s)\, \mathcal{L}_{\text{LPIPS}}(\hat{x}_q^{(k)}, x). \tag{26}$$

The perceptual weight $\lambda_{\text{LPIPS}}(s)$ increases linearly from 0 to 1 during the first 300 optimization steps of this stage and is maintained at 1 thereafter. This gradual ramp-up introduces perceptual supervision smoothly while retaining the flow-matching objective.

C.4 Generation Residual Codec Training

Latent-space rate-distortion training. We first train the complete codec for $5,000$ steps using a latent-space rate-distortion objective. Specifically, the generation residual reconstruction is supervised with Smooth L1 loss,

$$\mathcal{L}_{\text{res}} = \mathcal{L}_{\text{SmoothL1}}\left(\hat{r}_g^{(k)}, r_g^{(k)}\right), \tag{27}$$

where the transition parameter of the Smooth L1 loss is set to $\beta_{\text{SL1}} = 0.02$. The entropy model provides the differentiable rate estimate $R_g^{(k)}$ defined in Equation (18). Combining these, the overall training objective is

$$\mathcal{L}_g^{\text{latent}} = R_g^{(k)} + \mathcal{L}_{\text{res}}. \tag{28}$$

Image-space perceptual refinement. Latent-space reconstruction supervision alone does not directly optimize the perceptual quality of the reconstructed image. We therefore freeze the analysis transform and entropy model and refine only the synthesis transform for 300 steps using

$$\mathcal{L}_g^{\text{synthesis}} = 0.05\mathcal{L}_{\text{res}} + 0.8\mathcal{L}_{\text{DISTS}}\left(\hat{x}_g^{(k)}, x\right). \tag{29}$$

This stage encourages the decoded generation residual to better preserve structural and textural information that is important for perceptual reconstruction.

Table 4: **Training hyperparameters across stages.**

Stage	Steps	Global batch	LR
1. Autoregressive prior fine-tuning	$2,000$	64	6×10^{-5}
2. Mixed-latent VAE decoder fine-tuning	$35,000$	16	2×10^{-5}
3. Quantization residual generator training			
Flow-matching pretraining	$3,000$	64	1×10^{-4}
Image-space perceptual refinement	$2,000$	32	1×10^{-5}
4. Generation residual codec training			
Latent-space rate-distortion training	$5,000$	8	1×10^{-4}
Image-space perceptual refinement	300	8	1×10^{-5}
Joint fine-tuning	$2,000$	8	1×10^{-5}
5. Residual-aware VAE decoder adaptation	$1,000$	16	1×10^{-6}

Joint fine-tuning. Finally, we jointly fine-tune the complete Generation Residual Codec for $2,000$ steps with rate, latent reconstruction, and perceptual supervision:

$$\mathcal{L}_g^{\text{full}} = R_g^{(k)} + 0.05\mathcal{L}_{\text{res}} + 0.8\mathcal{L}_{\text{DISTS}}\left(\hat{x}_g^{(k)}, x\right). \tag{30}$$

C.5 Residual-Aware VAE Decoder Adaptation

After training the two residual branches, we freeze them and adapt only the VAE decoder for $1,000$ iterations using the compensated latent representation

$$\hat{h}_c^{(k)} = \hat{h}_q^{(k)} + \hat{r}_q^{(k)} + \hat{r}_g^{(k)}. \tag{31}$$

Given the reconstruction $\hat{x}^{(k)} = \tilde{\mathcal{D}}(\hat{h}_c^{(k)})$, we optimize

$$\mathcal{L}_{\text{adapt}} = 0.05\mathcal{L}_{\text{Charb}} + \mathcal{L}_{\text{LPIPS}}(\hat{x}^{(k)}, x) + 0.8\mathcal{L}_{\text{DISTS}}(\hat{x}^{(k)}, x), \tag{32}$$

where $\mathcal{L}_{\text{Charb}}$ denotes the Charbonnier reconstruction loss (Charbonnier et al., 1994) with $\epsilon_{\text{Charb}} = 10^{-3}$. This final stage adapts the decoder to the latent distribution produced after the quantization and generation residual compensation.

C.6 Optimization Settings

All training stages use AdamW (Loshchilov & Hutter, 2019) with linear warm-up followed by cosine learning-rate decay and gradient-norm clipping at 1. Training the quantization residual generator uses $\beta = (0.9, 0.999)$, zero weight decay, and a final learning-rate ratio of 0.2, while training the generation residual codec uses the same β, a weight decay of 10^{-4}, and a final learning-rate ratio of 0.1. Both VAE decoder fine-tuning and residual-aware decoder adaptation use $\beta = (0.9, 0.95)$ with zero weight decay. Stage-specific training iterations, global batch sizes, and learning rates are summarized in Table 4.

D Bitstream and Rate Accounting

The two residual branches incur different rate costs: quantization residual generation requires no additional payload, whereas generation residual coding introduces an additional transmitted bitstream. We report the measured bitrate contributions of the text condition, token prefix, and generation-residual bitstream. Specifically, when compressing an $H \times W$ image, the total bitrate is

$$R_{\text{total}}^{(k)} = \frac{B_{\text{text}} + B_{\text{prefix}}^{(k)} + B_g^{(k)}}{HW}, \tag{33}$$

where B_{text}, $B_{\text{prefix}}^{(k)}$, and $B_g^{(k)}$ denote the measured bit lengths of the text condition, transmitted token prefix, and generation-residual bitstream, respectively. The generation residual latent is entropy-coded using byte-rANS (Duda, 2013), and $B_g^{(k)}$ is measured from the complete encoded residual stream. The text rate is computed from its UTF-8 byte length.

As reported in Table 5, the generation-residual bitstream occupies only 1.98×10^{-4} - 3.14×10^{-4} bpp across the evaluated prefix depths. Its fraction of the total bitrate decreases from 6.64% at $k = 5$ to 0.36% at $k = 10$, as the transmitted token prefix increasingly dominates the overall rate. Figure 9 further visualizes the average bitrate composition across the prefix depths on DIV2K. These results show that generation residual coding incurs only a small fraction of the overall transmitted rate.

E Evaluation Protocol

We evaluate saved 8-bit RGB reconstructions at a fixed resolution of 1024×1024 on the DIV2K validation set and CLIC2020 test set, containing 100 and 428 images, respectively. Reconstruction quality is evaluated from three complementary perspectives: distortion-oriented fidelity, perceptual similarity, and distributional fidelity.

Evaluation settings. We evaluate transmitted prefix depths $k \in \{5, 6, 7, 8, 9, 10\}$ within the 13-scale token hierarchy. Autoregressive suffix generation uses multinomial sampling with temperature 0.5, top-2, top-$p = 0.97$, and classifier-free guidance scale 3.0 applied to the logits. The Quantization Residual Generator uses $N = 4$ sampling steps in Algorithm 2, with unit noise temperature and no classifier-free guidance.

Distortion-oriented fidelity metrics. We report PSNR and MS-SSIM (Wang et al., 2003) to evaluate conventional reconstruction fidelity. Both metrics are computed on full-resolution RGB images in $[0, 1]$. PSNR is averaged over images in dB, while MS-SSIM uses the default five-scale setting of `pytorch-msssim`.

Perceptual similarity. We report LPIPS (Zhang et al., 2018), DISTS (Ding et al., 2022), and CLIP image-to-image similarity (Radford et al., 2021) to measure perceptual similarity between each reconstruction and its corresponding original image. LPIPS uses AlexNet v0.1 with inputs scaled to $[-1, 1]$, while DISTS is evaluated on images scaled to $[0, 1]$; both are computed at the original resolution without resizing. CLIP image-to-image similarity is computed as 100 times the cosine similarity between ℓ_2-normalized ViT-B/32 image embeddings using standard CLIP preprocessing.

Distributional fidelity. We evaluate the fidelity between the distributions of reconstructed and source images using FID (Heusel et al., 2017), KID (Bińkowski et al., 2018), CMMD (Jayasumana et al., 2024), and FD-DINOv2 (Oquab et al., 2023). FID and KID are computed with `torchmetrics` using 2048-dimensional Inception features. To obtain sufficient samples for distribution estimation, each 1024×1024 image is partitioned into two 256×256 patch grids with offsets $(0, 0)$ and $(128, 128)$ and stride 256, yielding 25 valid patches per image. This results in 2,500 patches for DIV2K and 10,700 patches for CLIC2020. FID is computed from feature means and covariances, while KID uses the polynomial kernel $\kappa(u, v) = (u^\top v/2048 + 1)^3$ and averages 100 estimates, each computed from 1,000 randomly sampled feature vectors with a fixed random seed. CMMD is computed from one ℓ_2-normalized CLIP ViT-L/14@336 embedding per image using an RBF kernel with $\sigma = 10$ and the diagonal-inclusive MMD estimator scaled by 1000. FD-DINOv2 uses one 1024-dimensional DINOv2 ViT-L/14 CLS embedding (Oquab et al., 2023) per image without ℓ_2 normalization, with inputs resized to 224×224 and normalized using ImageNet statistics.

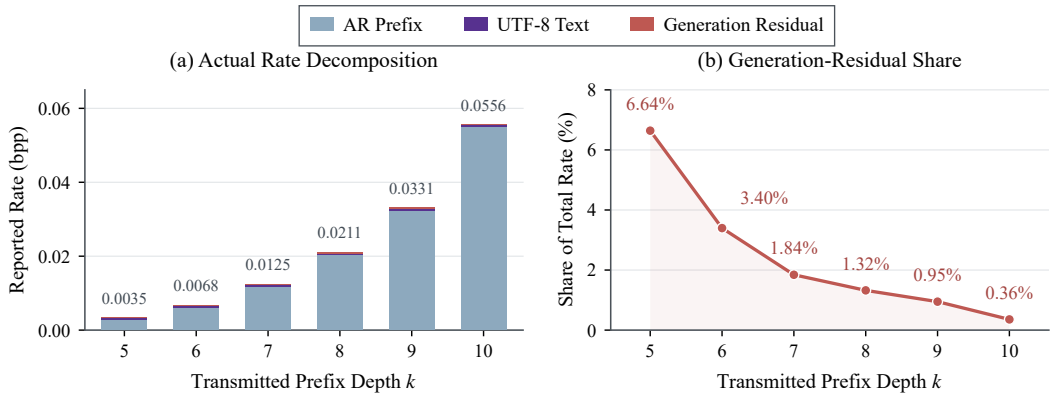

Figure 9: **Rate allocation on DIV2K.** The left panel shows the bitrate of each transmitted component across prefix depths, while the right panel shows the fraction of the total bitrate contributed by the generation-residual stream.

Table 5: **Rate decomposition on DIV2K.** Rates are reported in bpp and averaged over the DIV2K validation set.

k	Text	AR Prefix	Gen. Residual	Total
5	3.75×10^{-4}	2.85×10^{-3}	2.30×10^{-4}	3.46×10^{-3}
6	3.75×10^{-4}	6.16×10^{-3}	2.30×10^{-4}	6.76×10^{-3}
7	3.75×10^{-4}	1.18×10^{-2}	2.29×10^{-4}	1.25×10^{-2}
8	3.75×10^{-4}	2.05×10^{-2}	2.79×10^{-4}	2.11×10^{-2}
9	3.75×10^{-4}	3.24×10^{-2}	3.14×10^{-4}	3.31×10^{-2}
10	3.75×10^{-4}	5.51×10^{-2}	1.98×10^{-4}	5.56×10^{-2}

Signed KID estimates. Table 6 reports the raw KID values of ResARC on DIV2K across different prefix depths. The reported `std` is computed across the 100 random subsets used in the main KID evaluation. For $k = 8, 9, 10$, we additionally repeat the evaluation with ten random seeds and report the range of the resulting mean KID values. As k increases, KID consistently decreases and becomes negative at $k = 9$ and $k = 10$. Although the population KID is non-negative, its unbiased finite-sample estimator may take negative values due to sampling variance (Bińkowski et al., 2018; Gretton et al., 2012). For BD-rate computation, we retain the raw signed KID values. For visualization only, values below 2^{-20} are clipped to 2^{-20} in the logarithmic KID plot in Figure 4.

Table 6: **ResARC KID on DIV2K.** All KID entries are in 10^{-4} units.

k	bpp	KID mean $\pm$ std	10-seed range
5	0.00346	5.477 ± 1.056	-
6	0.00676	3.177 ± 0.946	-
7	0.01245	1.380 ± 0.830	-
8	0.02110	0.675 ± 0.854	[0.487, 0.835]
9	0.03312	-0.397 ± 0.846	[−0.501, −0.243]
10	0.05562	-1.140 ± 0.782	[−1.191, −0.903]

F Additional Ablation Studies

We provide supplementary results on the contributions and compensation strategies of the two residuals, together with additional ablations on the training objectives for the two residual branches and the VAE decoder input distributions used during mixed-latent fine-tuning. All experiments are evaluated on DIV2K. We report signed BD-rate (%) relative to the reference configuration specified in each experiment, where lower values indicate better rate-quality performance. Dashes denote cases with insufficient quality overlap for reliable computation.

F.1 Supplementary Results for Residual Compensation

We further provide results on the contribution of the two residual branches through rate-perception curves. For all configurations, we follow the same training stages and optimization settings, while only adding the corresponding residual branch to the autoregressive baseline. As shown in Figure 10, adding either the generation residual branch or the quantization residual branch consistently

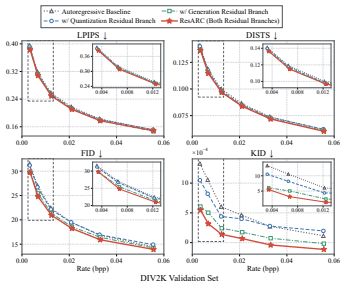

Figure 10: **Rate-perception curves for the contribution of the quantization and generation residuals on DIV2K.**



Figure 11: **Qualitative illustration of the complementary contributions of the two residual branches.**

Table 7: **Ablation of residual compensation strategies.** BD-rate (%) relative to ResARC (Ours).

Configuration	LPIPS↓	DISTS↓	FID↓	KID↓
Generate both residuals	**-0.75**	+10.46	+11.11	+27.34
Transmit both residuals	+4.67	+6.29	+6.99	+35.65
Generate r_q + transmit r_g (Ours)	0.00	**0.00**	**0.00**	**0.00**

improves performance across all four metrics. Moreover, combining both residual branches in full ResARC yields further improvements and achieves the best overall performance, demonstrating the complementary benefits of the two residuals.

Figure 11 further illustrates the complementary roles of the two residual branches. The enlarged regions show differences in the reconstructed feather details of the penguin and the mouth contours of the portrait. Combining both branches yields reconstructions that better preserve local details and image structures, and achieves the lowest DISTS in the displayed examples.

F.2 Supplementary Results for Residual Compensation Strategies

We further provide the BD-rate comparison results for the three residual compensation strategies on DIV2K using signed BD-rate. All experiments use the same two residual targets, autoregressive backbone, and sampling settings. As shown in Table 7, relative to ResARC, generating both residuals incurs 10.46% and 11.11% higher BD-rates under DISTS and FID, respectively, despite a slight improvement under LPIPS, while transmitting both residuals increases BD-rate across all four metrics. These results support the residual-specific compensation design of ResARC, where the quantization residual r_q is generated from decoder-side context without additional transmitted bits, while r_g is explicitly transmitted to provide source-dependent correction.

F.3 Training Objectives for the Residual Branches

We study the perceptual objectives used to train the two residual branches. For each branch, we vary only the perceptual loss while keeping the remaining training objective fixed: flow matching for the quantization residual generator, and rate together with Smooth L1 loss for the generation residual codec.

Table 8 reports BD-rates relative to the objective used in ResARC for each branch. For the Quantization Residual Generator, using the LPIPS loss yields the best LPIPS, FID, and KID BD-rates among the tested objectives. However, adding DISTS supervision improves the DISTS BD-rate by only 0.65%, while increasing the LPIPS, FID, and KID BD-rates by 1.70%, 1.27%, and 12.10%, respectively. For the Generation Residual Codec, adding the LPIPS loss to the DISTS objective reduces the LPIPS BD-rate by 6.40%, but increases the DISTS, FID, and KID BD-rates by 9.69%, 19.50%, and 79.17%, respectively. These results support using the LPIPS objective for the quantization residual generator and DISTS loss for the generation residual codec.

F.4 Mixed-Latent VAE Decoder Fine-Tuning

We further investigate the input distribution used to fine-tune the VAE decoder. Quantization residual compensation shifts the decoder input from the quantized latent h_q toward the continuous latent h. We therefore compare three fine-tuning settings: quantized latents only, continuous latents only,

Table 8: **Training-objective ablations on DIV2K.** Signed BD-rate (%) is reported relative to the objective used in ResARC (Ours) within each branch. FM denotes flow matching; - denotes no common quality interval.

Training objective	LPIPS↓	DISTS↓	FID↓	KID↓
Quantization residual generator				
FM	+3.38	+2.60	+7.62	+45.37
FM + DISTS	+2.81	−0.25	+4.33	+32.50
FM + LPIPS + DISTS	+1.70	**−0.65**	+1.27	+12.10
FM + LPIPS (Ours)	**0.00**	0.00	**0.00**	**0.00**
Generation residual codec				
Rate + Smooth L1	+64.27	+155.82	+278.69	-
Rate + Smooth L1 + LPIPS	+62.97	+194.08	+233.80	+787.71
Rate + Smooth L1 + LPIPS + DISTS	**−6.40**	+9.69	+19.50	+79.17
Rate + Smooth L1 + DISTS (Ours)	0.00	**0.00**	**0.00**	**0.00**

Table 9: **Ablation of decoder input distributions during fine-tuning.** BD-rate (%) is reported relative to mixed fine-tuning ($p = 0.5$).

Type	LPIPS↓	DISTS↓	FID↓	KID↓
Quantized ($p = 0$)	+0.06	+7.42	+4.03	+0.73
Continuous ($p = 1$)	+4.45	+31.99	+52.80	+204.28
Mixed ($p = 0.5$)	**0.00**	**0.00**	**0.00**	**0.00**

and a mixed setting with $p = 0.5$, where p denotes the probability of sampling h. As shown in Table 9, the mixed setting achieves the best overall BD-rate across all four metrics. Compared with mixed-latent fine-tuning, using only quantized latents incurs 7.42% and 4.03% higher BD-rates under DISTS and FID, respectively, while using only continuous latents leads to substantially higher BD-rates across all four metrics. These results support mixed-latent fine-tuning for adapting the VAE decoder to compensated latent representations.

G Additional Rate-Quality Assessment

G.1 Additional Rate-Quality Curves

We additionally report PSNR, MS-SSIM (Wang et al., 2003), and CLIP image-to-image similarity (Radford et al., 2021) on the DIV2K validation and CLIC2020 test sets. As shown in Figure 12, ResARC achieves competitive CLIP similarity at low bitrates, while maintaining comparable PSNR and MS-SSIM performance. These results complement the main perceptual evaluation in Figure 4 by characterizing pixel-level and structural reconstruction fidelity.

G.2 BD-rate Comparison

Table 10 summarizes the signed BD-rate results across the six perceptual quality metrics. Using ARPC (Zhang et al., 2026b) as the reference, ResARC achieves the lowest reported BD-rates under DISTS and all four distributional fidelity metrics on both datasets, while remaining competitive under LPIPS. Specifically, compared with the autoregressive codec ARPC, ResARC achieves BD-rate reductions of 42.94% and 50.98% on DIV2K under DISTS and FID, respectively, and 46.51% and 51.22% on CLIC2020. ResARC also exhibits strong performance under KID, CMMD, and FD-DINOv2 while remaining competitive under LPIPS. Compared with diffusion-based codecs such as StableCodec, ResARC achieves lower DISTS and better overall distributional fidelity at comparable bitrates. These results further validate the effectiveness of explicitly compensating for both residuals in autoregressive generative codecs.

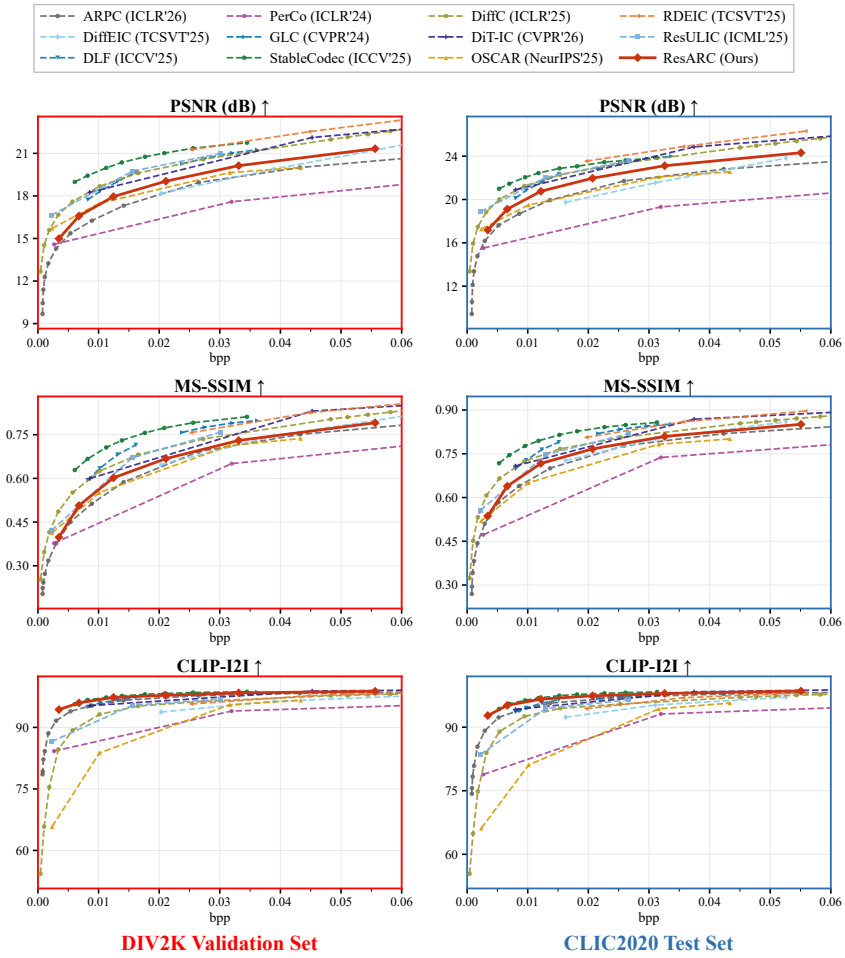

Figure 12: **Additional rate-quality comparison with leading generative codecs on the DIV2K validation and CLIC2020 test sets.**

H More Visual Results

H.1 Visual Effect of Prefix Depth

To inspect progressive rate control, Figure 13 compares the same five images as the transmitted prefix depth increases from $k = 5$ to $k = 10$. As more ground-truth prefix tokens are transmitted, the bitrate increases and fewer suffix scales need to be autoregressively generated, leading to progressively improved recovery of fine-grained details and image structures, including butterfly markings, mushroom textures, railing patterns, and facial details.

H.2 Additional Visual Comparisons

We provide additional qualitative comparisons with diffusion-based and autoregressive generative codecs in Figures 14 and 15. ResARC consistently preserves finer details and image structures, including facial details, headlight contours, glass boundaries, plate rims, and petal textures. It also achieves the lowest full-image DISTS in every displayed example. Together with Figure 5, these results further demonstrate the effectiveness of ResARC at ultra-low bitrates. The improvements are consistent with our residual-aware design: quantization residual generation recovers fine-grained information lost during tokenization, while generation residual compensation reduces inaccuracies introduced by autoregressive suffix generation.

Table 10: **Signed BD-rate (%) relative to ARPC.** Bold pink and blue denote the lowest and second-lowest reported values for each metric and dataset.

Methods	LPIPS ↓		DISTS ↓		FID ↓	
	DIV2K	CLIC2020	DIV2K	CLIC2020	DIV2K	CLIC2020
ARPC (ICLR'26) (Zhang et al., 2026b)	0.00	0.00	0.00	0.00	0.00	0.00
DiffEIC (TCSVT'25) (Li et al., 2024)	+106.16	+104.11	+520.97	+350.87	+423.82	+466.27
DLF (ICCV'25) (Xue et al., 2025a)	-32.50	-31.46	+22.63	+3.75	+42.75	+42.42
PerCo (ICLR'24) (Careil et al., 2024)	+190.35	+243.61	+639.33	+567.67	+770.14	+1207.55
GLC (CVPR'24) (Jia et al., 2024)	-34.72	-33.83	+43.01	+24.56	+76.14	+95.65
StableCodec (ICCV'25) (Zhang et al., 2025)	**−39.81**	**−39.78**	+1.83	-7.92	**−16.06**	**−12.71**
DiffC (ICLR'25) (Vonderfecht & Liu, 2025)	+31.40	+17.23	+359.73	+161.03	+417.41	+293.57
DiT-IC (CVPR'26) (Shi et al., 2026)	**−55.33**	**−57.20**	**−8.88**	**−19.24**	+99.10	+132.96
OSCAR (NeurIPS'25) (Guo et al., 2025)	+144.50	+178.24	+224.40	+236.37	+751.72	+876.05
RDEIC (TCSVT'25) (Li et al., 2025)	-7.45	-1.43	+387.95	+229.43	+272.37	+402.97
ResULIC (ICML'25) (Ke et al., 2025)	+35.52	+43.55	+297.70	+217.00	+236.68	+261.12
ResARC (Ours)	-28.88	-31.53	**−42.94**	**−46.51**	**−50.98**	**−51.22**

Methods	KID ↓		CMMD ↓		FD-DINOv2 ↓	
	DIV2K	CLIC2020	DIV2K	CLIC2020	DIV2K	CLIC2020
ARPC (ICLR'26) (Zhang et al., 2026b)	0.00	**0.00**	0.00	0.00	0.00	0.00
DiffEIC (TCSVT'25) (Li et al., 2024)	+639.93	+736.06	+634.54	+39.11	+204.14	+136.24
DLF (ICCV'25) (Xue et al., 2025a)	+28.85	+55.91	+204.20	+115.17	+5.94	-1.91
PerCo (ICLR'24) (Careil et al., 2024)	+2936.40	+1902.80	+1519.05	+556.87	+194.30	+268.23
GLC (CVPR'24) (Jia et al., 2024)	+164.57	+191.31	+91.73	+166.07	-21.18	-16.97
StableCodec (ICCV'25) (Zhang et al., 2025)	**−8.53**	+21.16	**−32.59**	**−61.17**	**−31.57**	**−34.50**
DiffC (ICLR'25) (Vonderfecht & Liu, 2025)	+792.74	+430.30	+479.16	+78.25	+128.05	+97.77
DiT-IC (CVPR'26) (Shi et al., 2026)	+464.09	+353.40	+583.35	+153.67	+52.70	+26.33
OSCAR (NeurIPS'25) (Guo et al., 2025)	+894.84	+1012.69	+1857.37	+839.67	+460.83	+573.09
RDEIC (TCSVT'25) (Li et al., 2025)	+538.40	+700.73	+380.34	+114.42	+97.88	+64.99
ResULIC (ICML'25) (Ke et al., 2025)	+801.30	+407.62	+312.75	+45.41	+47.39	+68.55
ResARC (Ours)	**−73.11**	**−42.84**	**−57.75**	**−82.74**	**−40.15**	**−38.63**

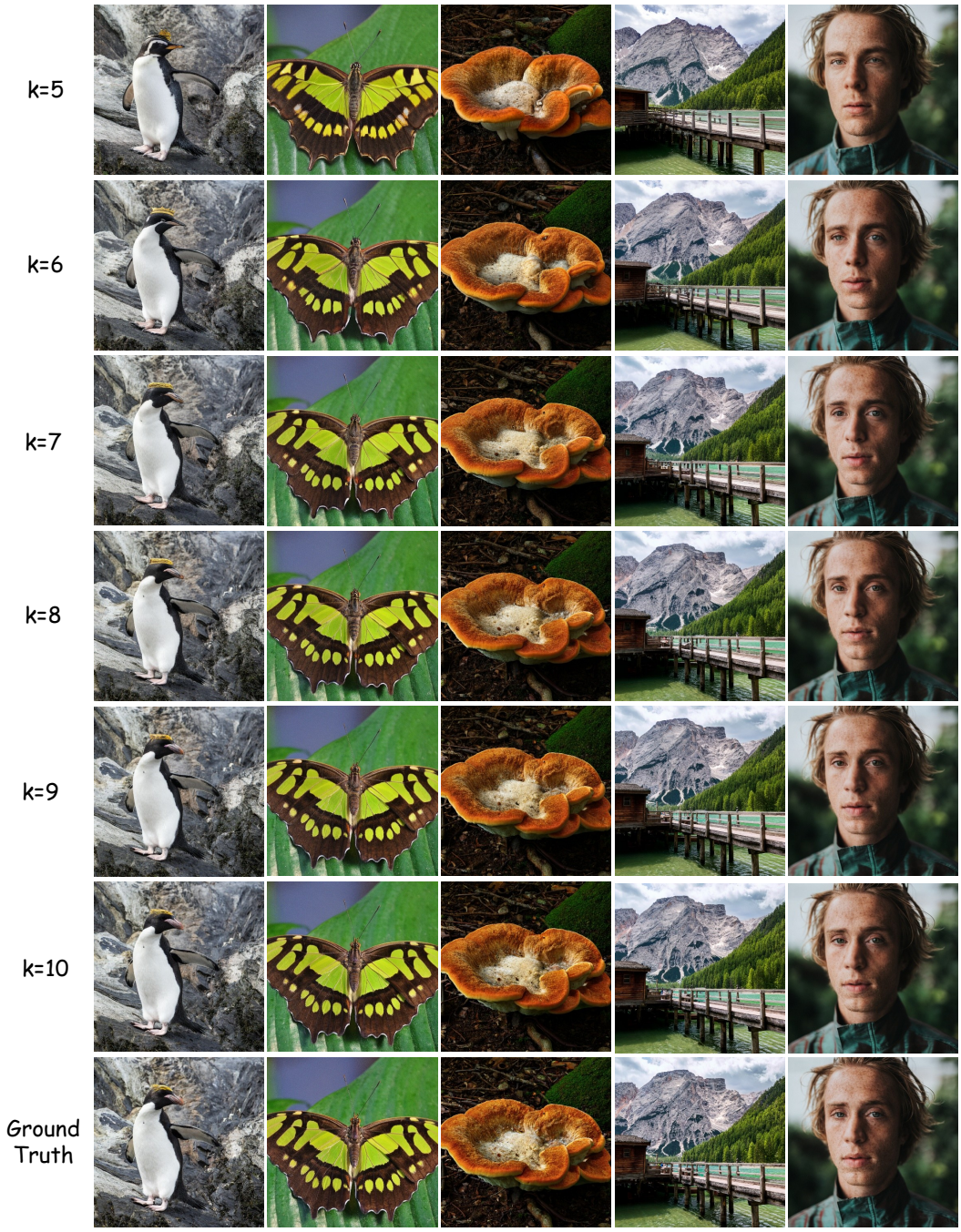

Figure 13: **Progressive reconstruction across different prefix depths.** Increasing k transmits more ground-truth prefix scales and progressively improves reconstruction quality.

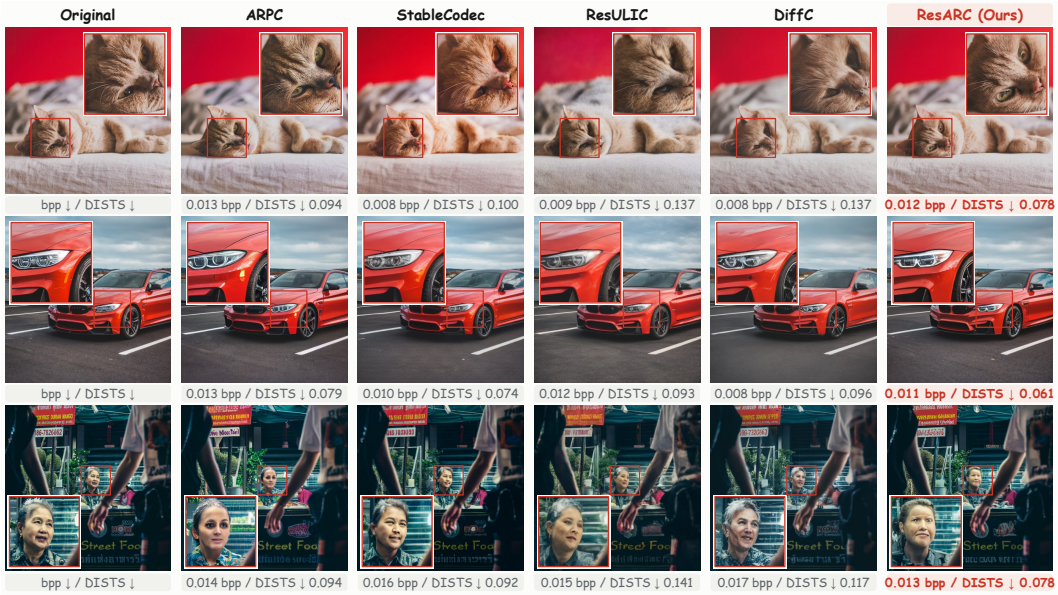

Figure 14: **More visual comparisons on CLIC2020.** Labels report per-image bitrate (bpp) and DISTS.



Figure 15: **More visual comparisons on DIV2K.** Labels report per-image bitrate (bpp) and DISTS.