PNVC: Towards Practical INR-based Video Compression

Ge Gao, Ho Man Kwan, Fan Zhang, David Bull

Visual Information Laboratory, University of Bristol Bristol, BS1 5DD, UK
{ge1.gao, hm.kwan, fan.zhang, dave.bull}@bristol.ac.uk

Abstract

Neural video compression has recently demonstrated significant potential to compete with conventional video codecs in terms of rate-quality performance. These learned video codecs are however associated with various practicality issues related to decoding complexity (for autoencoder-based methods) and/or system delays (for implicit neural representation (INR) based models), which which present major obstacles on their path to real-world applications. In this paper, targeting a practical neural video codec, we propose a novel INR-based coding framework, PNVC, which innovatively combines autoencoder-based and overfitted solutions. Our approach benefits from several design innovations, including a new structural reparameterization-based architecture, hierarchical quality control, modulation-based entropy modeling, and scale-aware positional embedding. Supporting both low delay (LD) and random access (RA) configurations, PNVC outperforms existing INR-based codecs, achieving nearly 35%+ BD-rate savings against HEVC HM 18.0 (LD) - almost 10% more compared to the state-of-the-art INR-based codec, HiNeRV, and 5% more over VTM 20.0 (LD), while maintaining 20+ FPS decoding speeds for 1080p content. This marks a significant advancement for INR-based video coding, bringing it closer to practical deployment.

Code — https://github.com/ge1-gao/PNVC
Extended version — https://arxiv.org/abs/2409.00953

1 Introduction

There is an ever-increasing requirement for higher video coding efficiency due to the significantly increased demand for high quality digital video content. Unlike standardized video codecs, such as H.265/HEVC (Sullivan et al. 2012) and H.266/VVC (Bross et al. 2021) that offer impressive performance but based on evolutions of conventional architectures, neural video compression (NVC) is enjoying much faster development cycles and rapidly increasing performance benchmarks using an optimized, data-driven end-to-end architecture. Advances in this research area have delivered a wide variety of candidate neural codecs, some of which (Xiang, Tian, and Zhang 2022; Li, Li, and Lu 2024) are reported to match or outperform compression performance of the latest state-of-the-art standard coding methods.

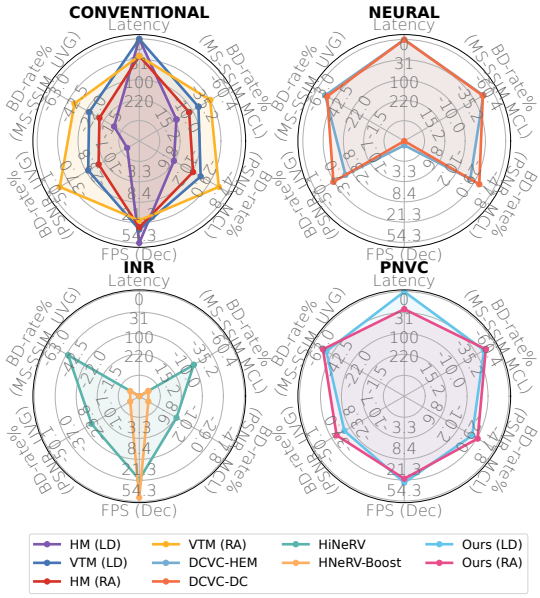

Figure 1: Radar plots illustrating the performance of proposed PNVC codec (ours) and nine other conventional and neural video codecs, in terms of coding efficiency (BD-rate measured by PSNR and MS-SSIM on UVG and MCL-JVC datasets, against HM 18.0, LD), decoding speeds (FPS) and coding latency[1]. It can be observed that PNVC demonstrates excellent performance in all these aspects.

Although showing promise in terms of coding gain, NVCs (primarily those using autoencoder-based backbones) are associated with significant complexity issues, in particular on the decoder side, making them resource-intensive and impractical for many real-world applications. Although common complexity reduction techniques such as pruning and quantization (Peng et al. 2024; Hu and Xu 2023) can alleviate these limitations, this typically results in a nontrivial reduction in coding efficiency.

More recently, implicit neural representation (INR)-based coding methods (Chen et al. 2021; He et al. 2023; Kwan et al. 2024) have attracted increasing attention as a paradigm-shifting solution to achieving high coding performance and low (decoding) complexity. This type of ap-

proach typically utilizes a lightweight neural network to overfit input video data by mapping coordinates directly to pixel values. Although the latest INR-based codecs (Kwan et al. 2024) have shown consistent coding gains over many conventional and neural video codecs, they have a major limitation in that their compression strategy to represent an entire video sequence/dataset with a single monolithic model. While this practice maximizes compression efficiency, it requires processing a large number of video frames (e.g., a few hundred to a few thousand) in each encoding session, which conflicts with commonly used coding configurations, where flexible system latency[1], e.g., Low Delay and Random Access modes in VVC VTM (Bossen et al. 2023), is often required. This issue prevents INR-based video codecs from being as performant on shorter sequences or adopted in many practical applications.

In this paper, we introduce **PNVC**, a novel (**P**ractical) I**N**R-based **V**ideo **C**ompression framework that tackles the aforementioned limitations, which enables flexible coding configurations (low latency) while still achieving competitive coding performance and low encoding/decoding complexity. The proposed PNVC is built upon a hierarchical backbone that generalizes autoregressive models (Child 2021; Lu et al. 2024) along the scale dimension and is seamlessly interchangeable with content- or modulation-based INR models (Mehta et al. 2021; Lee et al. 2024; Kwan et al. 2024). Our approach leverages a *pretrain-then-overfit* strategy, akin to an online learning setup (Tang et al. 2024; Chen et al. 2024), enabling the model to generalize across diverse content during pretraining whilst adapting to input-specific contents during overfitting. This is augmented with a novel reparameterization method, among other architectural and optimization innovations, that allows for unconstrained model capacity during training while ensuring low-complexity inference. This decoupling enables more effective optimization without sacrificing efficiency in deployment. The main contributions of this paper include:

1) We propose a novel **INR-based video codec** that integrates autoencoder-based backbone with overfitted solutions, offering competitive coding performance, relatively low encoding and decoding complexity, and a flexible coding latency configuration.
2) We design a novel **reparameterization**-based scheme, dubbed **ModMixer**, to sufficiently pretrain as well as overfitting a lightweight backbone with stronger modeling capacity and more diverse optimization directions, without incurring extra inference costs.
3) We further introduce several modifications, including **hierarchical quality parameters**, **modulation-based entropy model** with asymmetric context grouping, and **scale-aware hierarchical positional encoding** to enhance the compression performance.

The proposed PNVC demonstrates very competitive rate-distortion performance in both Low Delay and Random Access configurations (as defined in many video coding standards), whilst circumventing the latency and encoding complexity problems associated with existing INR-based video codecs. Specifically, as shown in Figure 1, on both UVG and MCL-JCV, the proposed model evidently outperforms VTM 20.0 (LD), and HiNeRV in BD-rate, measured in both PSNR and MS-SSIM. It is also associated with much lower encoding latency (system delays) compared to existing INR-based videos, and a faster decoding speed over autoencoder-based neural video coding models. We believe this work marks a meaningful step forward in INR-based video compression, moving it closer to practical adoption.

[1]Here latency is defined as the system delays in the coding process, e.g., the number of consecutive bi-directional predicted frames in a GOP (Group of Pictures).

2 Related Work

Video compression has been a long-standing research topic, evolving from hand-crafted standard codecs (Wiegand et al. 2003; Sullivan et al. 2012; Bross et al. 2021) to deep learning techniques that enhance conventional tools (Yan et al. 2018; Ma, Zhang, and Bull 2020) and end-to-end optimized frameworks (Lu et al. 2019; Agustsson et al. 2020). Recent learning-based methods have focused on improving motion estimation (Li, Li, and Lu 2023), feature space conditional coding (Hu, Lu, and Xu 2021; Li, Li, and Lu 2021), instance-adaptive overfitting (van Rozendaal, Huijben, and Cohen 2021; Yang and Mandt 2023; Yang, Oh, and Park 2024; Tang et al. 2024), and novel architectures (Ho et al. 2022; Xiang, Tian, and Zhang 2022; Mentzer et al. 2022). Despite competitive coding performance, high (decoding) computational complexity has limited practical deployment, with structured pruning approaches (Hu and Xu 2023; Peng et al. 2024) achieving only a limited reduction in complexity at the cost of significant performance drops.

Implicit Neural Representations (INRs) have been increasingly employed in recent years to represent and compress various multimedia signals, including images (Dupont et al. 2021; Strümpler et al. 2022), videos (Zhang et al. 2021; Kwan et al. 2024) and volumetric content (Ruan et al. 2024). INRs learn a coordinate-based mapping function and encode data in network parameters. Existing implicit neural video representation (NeRV) models can be categorized as: i) *index-based* methods taking frame (Chen et al. 2021), patch (Bai et al. 2023), or disentangled spatial/grid coordinates (Li et al. 2022) as input, or ii) *content-based* methods (Chen et al. 2023; Kwan et al. 2024; Kim et al. 2024; Leguay et al. 2024) with content-specific embeddings as inputs. In these cases, the video coding task is reformulated into a model compression problem, leveraging pruning, quantization, and entropy-constraint optimization (Gomes, Azevedo, and Schroers 2023). However, training NeRV models on entire video sequences or datasets results in high system latency[1], making them unsuitable for applications requiring quick response times and creating an unfair comparison with latency-constrained codecs.

3 Methods

3.1 Overview

In the proposed PNVC framework, each video sequence $\mathcal{X} = \{\mathbf{x}_t\}_{t=1}^{T},\ \mathbf{x}_t \in \mathbb{R}^{3\times H\times W}$, is segmented into Groups

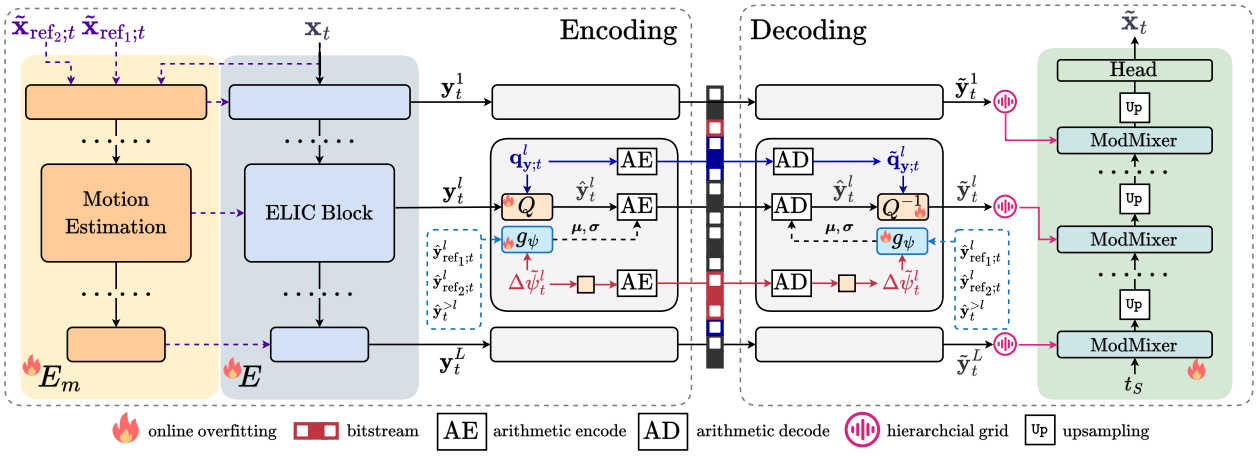

Figure 2: Illustration of the proposed PNVC framework.

of Pictures (GOPs) of length N that are independently encoded, where T, H, and W represent the length, height, and width of $\mathcal{X}$, respectively[2]. Within each GOP, frames are either intra-coded as I frames or inter-coded as P/B frames to exploit spatial and/or temporal redundancy within the video.

A patch-based INR representation is used in PNVC, i.e., each non-overlapping patch of size $k \times k$ from a frame $\mathbf{x}_t$ is parameterized by the decoder D given the patch's 3D coordinates. We use L sets of latent tokens $\{\mathbf{y}_t^l\}_{l=1}^L$ to encode the coordinate-aware, overfitted contents. For each level l, $\mathbf{y}_t^l$ consists of scale and shift parameters $\boldsymbol{\gamma}_t^l$ and $\boldsymbol{\beta}_t^l$, concatenated channel-wise, with a resolution of $(\frac{H}{k\times 2^l}, \frac{W}{k\times 2^l})$. These are generated by the content encoder E and motion encoder E_m and used to 'overfit' the decoder D by modulating its (hidden) feature distributions. The latent grids are quantized by Q, producing $\{\hat{\mathbf{y}}_t^l\}_{l=1}^L = \{\hat{\boldsymbol{\gamma}}_t^l \oplus \hat{\boldsymbol{\beta}}_t^l\}_{l=1}^L$ based on the corresponding quantization parameters $\{\mathbf{q}_{\mathbf{y};t}^l\}_{l=1}^L$, where $\oplus$ denotes the concatenate operation.

An entropy model $g_{\boldsymbol{\psi}}$ is adopted to estimate the probability mass function (PMF) of the hierarchical latents semi-autoregressively along the spatial, channel, and scale axes. The parameter of the entropy model, $\boldsymbol{\psi}$, is overfitted and updated as $\boldsymbol{\psi}_t^l \rightarrow \boldsymbol{\psi} + \Delta\tilde{\boldsymbol{\psi}}_t^l$ for level l. Here, $\boldsymbol{\psi}$ represents the pretrained parameters, and $\Delta\tilde{\boldsymbol{\psi}}_t^l$ are the (dequantized) parameter updates obtained through online overfitting. The bitstream includes $\{\hat{\mathbf{y}}_t^l, \Delta\hat{\boldsymbol{\psi}}_t^l, \tilde{\mathbf{q}}_{\mathbf{y};t}^l, \tilde{\mathbf{q}}_{\boldsymbol{\psi};t}^l\}_{l=1}^L$, where $\{\tilde{\mathbf{q}}_{\mathbf{y};t}^l\}_{l=1}^L$ and $\{\tilde{\mathbf{q}}_{\boldsymbol{\psi};t}^l\}_{l=1}^L$ are the inverse quantization parameters corresponding to latents and weight updates, respectively.

At the decoder, the entropy model is updated by $\{\tilde{\boldsymbol{\psi}}_t^l\}_{l=1}^L$ decoded from the bitstream. It is used to help entropy decode the hierarchical latent grids that are then dequantized by Q^{-1} using $\{\tilde{\mathbf{q}}_{\mathbf{y};t}^l\}_{l=1}^L$ to produce $\{\tilde{\mathbf{y}}_t^l\}_{l=1}^L$. These dequantized latents undergo positional embedding based on a scale-aware hierarchical decomposition scheme, allowing content-specific variations to be queried by spatial-temporal coordinates. It uses L stacked ModMixer blocks to progressively and conditionally map a content latent token t_S to patches of the reconstructed frame $\tilde{\mathbf{x}}_t$, with the intermediate activations of each block modulated by the corresponding latent grid.

Each P or B frame is coded following a workflow similar to that of I frames, but with an additional motion encoder E_m, as shown in Figure 2. With up to two previously reconstructed frames, $\tilde{\mathbf{x}}_{\text{ref}_1;t}$ and $\tilde{\mathbf{x}}_{\text{ref}_2;t}$, in the decoded frame buffer (they can be I or P/B frames) and the current frame $\mathbf{x}_t$, the multi-resolution latent grids $\{\mathbf{y}_t^l\}_{l=1}^L$ are generated by E but with layer-wise conditioning on the motion information extracted from E_{m}. Here, the entropy model $g_{\boldsymbol{\psi}}$ additionally takes into account the decoded reference latent tokens $\tilde{\mathbf{y}}_{\text{ref}_1;t}^l, \tilde{\mathbf{y}}_{\text{ref}_2;t}^l$ and the reconstructed latent tokens from higher-level layers, $\tilde{\mathbf{y}}_t^{>l}$, as input to further exploit spatio-temporal and hierarchical redundancy.

3.2 Encoder

The ELIC (He et al. 2022) model is adopted as the content encoder E to map I/P/B frames to the latent $\{\mathbf{y}_t^l\}_{l=1}^L$. For P/B frames, E is reconditioned by concatenating the hierarchical optical flow features, extracted by the motion encoder E_m, adapted from SpyNet (Ranjan and Black 2017), to incorporate the estimated motion information for further decorrelation. At each level, the 'patchification' is performed using a 2D convolution layer with a scale-aware stride $s_l = k \times 2^{(l-1)}$. After pre-training, we keep E and E_m fixed and only overfit the lower dimensional $\{\mathbf{y}_t^l\}_{l=1}^L$.

3.3 Quantization

A hierarchical quality structure (Li, Li, and Lu 2024; Jiang et al. 2024) is adopted, where the allowable bitrates are reweighted per frame based on its impact to the overall RD trade-offs within the GOP. We use a ConvLSTM network to capture the spatiotemporal characteristics of already decoded frames and estimate the quality parameter. This module is fixed after pretraining to prevent the frame-wise overfitting process from destroying the acquired hierarchical quality structure. For quantization, another set of finer-

[2]We follow the GoP definition in H.266/Versatile Video Coding Test Model (VTM), where the length of a GoP equals the number of consecutive bi-directional predicted (B) frames plus 1.

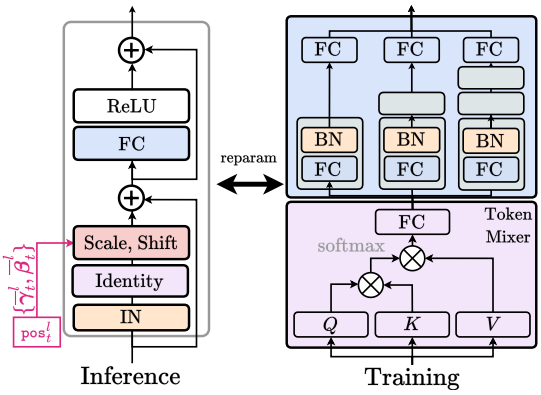

Figure 3: The architectures of the reparameterized ModMixer block during training and inference.

grained channel-wise quality parameters $\{\mathbf{q}^l_{\mathbf{y};t}\}_{l=1}^L$ are used. For inverse quantization, the corresponding $\{\tilde{\mathbf{q}}^l_{\mathbf{y};t}\}_{l=1}^L$ are retrieved and encoded into the bitstream as side information. The quantization and inverse quantization steps are exemplified using $\mathbf{y}^l_t$ as:

$$\hat{\mathbf{y}}^l_t[c][i][j] = \lfloor \mathbf{q}^l_{\mathbf{y};t}[c] \cdot \mathbf{y}^l_t \rceil, \tag{1a}$$

$$\tilde{\mathbf{y}}^l_t[c][i][j] = \tilde{\mathbf{q}}^l_{\mathbf{y};t}[c] \cdot \hat{\mathbf{y}}^l_t, \tag{1b}$$

where c, i, and j denote the channel and spatial indices, respectively. Here, we avoid division in both cases to avoid numerical instabilities. The quantization parameters for weight updates $\{\mathbf{q}_{\boldsymbol{\psi}}\}_{l=1}^L$ are updated following a similar procedure.

3.4 Entropy Coding

We use arithmetic coding to entropy code $\{\hat{\mathbf{y}}^l_t\}_{l=1}^L$. The PMFs are estimated by a Gaussian distribution $\mathcal{N}(\cdot)$, where the location and scale parameters $(\boldsymbol{\mu}_t[c][i][j], \boldsymbol{\sigma}_t[c][i][j])$ of each element $\hat{\mathbf{y}}^l_t[c][i][j]$ are semi-autoregressively predicted by the entropy network $g_{\boldsymbol{\psi}}$ based on the spatial, temporal, and multi-scale contexts of the element. We use the quadtree-based factorization (Li, Li, and Lu 2023) to encode $\hat{\mathbf{y}}^l_t$, but with two modifications. First, we respectively split the channels of $\hat{\boldsymbol{\gamma}}^l_t$ and $\hat{\boldsymbol{\beta}}^l_t$ (they are entropy coded in parallel) into four uneven groups (He et al. 2022), with sizes proportional to 1, 1, 2 and 4, i.e., the number of symbols decoded per decoding step is doubled, with more contexts made available. Moreover, we replace all concatenation operations to aggregate information with element-wise modulation, which we empirically found to be more efficient in information aggregation, and deploy the new ModMixer module (detailed in the next subsection) to construct the entropy model. The architecture illustration of the entropy model is provided in the *Supplementary*. For entropy model's updates $\{\Delta\hat{\boldsymbol{\psi}}^l_t\}_{l=1}^L$, a fully factorized nonparametric density model $\boldsymbol{\pi}$ (Ballé et al. 2018) is deployed, following (Gomes, Azevedo, and Schroers 2023; Zhang et al. 2024).

3.5 ModMixer

To enhance model performance without introducing extra inference cost, we devise a novel **ModMixer** module as a basic building block used in our entropy model $g_{\boldsymbol{\psi}}$ and decoder D. This approach is inspired by reparameterization methods (Ding et al. 2022; Shi, Zhou, and Gu 2024) that leverage the interconversion between linear architectures to trade training-time complexity for inference-time efficiency. Without loss of generality, we define an arbitrary linear layer, $\mathbf{W}\mathbf{h} + \mathbf{b}$, to be contracted algebraically from this general formulation:

$$\sum_{m=1}^{M} \left(\overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\gamma}_\mathbf{W}} \mathbf{W}_m + \overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\beta}_\mathbf{W}}\right) \mathbf{h} + \left(\overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\gamma}_\mathbf{b}} \mathbf{b}_m + \overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\beta}_\mathbf{b}}\right), \tag{2}$$

where $\mathbf{W}_m \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ and $\mathbf{b}_m \in \mathbb{R}^{C_{\text{out}}}$ denote the trainable, parallel basis of $\mathbf{W}$ and $\mathbf{b}$, respectively, and the coefficients $\{\overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\gamma}_\mathbf{W}}, \overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\beta}_\mathbf{W}}, \overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\gamma}_\mathbf{b}}, \overline{\boldsymbol{\Lambda}}_{m;\boldsymbol{\beta}_\mathbf{b}}\} \in \mathbb{R}^{C_{\text{out}}}$ denote the channel-wise affine mixand values. Here, each basis may be further expanded serially into (taking $\mathbf{W}_m$ as an example) $\mathbf{W}_m = \mathbf{W}_{m;N}\mathbf{W}_{m;N-1}\cdots\mathbf{W}_{m;1}$. Compared to standard reparameterization methods, Equation (2) establishes a more generalized visual tuning scheme that better suits the context of instance-adaptive video compression. With the affine transformation applied to both weight and bias terms, it is a *superset* encompassing both weight and feature space tuning, where the mixands could be flexibly re-casted to feature modulation (as in decoder), weight update (as in entropy model), or any other forms of visual tuning accordingly.

In our approach, we leverage the above reparameterization idea and propose a new basic building block, ModMixer, as illustrated in Figure 3. Each FC layer is reparameterized both serially and in parallel through stacks of linear layers with varying depths per branch (one per channel group) to capture the hierarchical representations efficiently. Further, a self-attention-style token-mixer is attached before the FC layer to induce inference-time spatial mixing. During pretraining, the token mixer is progressively *degenerated* and absorbed into the Instance Normalization (IN) and proceeding FC layer (Lin et al. 2024), by initializing a mask $\mathbf{M}$ as $\mathbf{1}$ that is gradually decayed to $\mathbf{0}$:

$$\mathbf{M} \odot \text{TokenMixer}\left(\mathbf{h}\right) + \left(1 - \mathbf{M}\right) \odot \mathbf{h} + \mathbf{h}. \tag{3}$$

The mask decaying strategy is inspired by (Wang et al. 2023; Peng et al. 2024), which leverage the more expressive teacher model to distill the smaller student. During overfitting, we optimize the residual updates, w.r.t the pretrained mixands and basis, which are then consolidated into either weight updates $\Delta\boldsymbol{\psi}^l_t$ for the entropy model or absorbed into the hierarchical latent grids $\mathbf{y}^l_t$ for the decoder (see *Supplementary* for full derivations and implementation details).

3.6 Decoder

In Figure 2, the decoder $D(\cdot)$, modulated by the positional embedded latents, incrementally restores spatial information while sequentially composing high-frequency details over low-frequency elements. Inspired by the hierarchical encoding method proposed by HiNeRV (Kwan et al. 2024), our method expresses each coordinate as an ordered set of digits, each of which corresponds to a hierarchy that recursively encodes residuals of the coarser-grained representational level.

This corresponds to a hierarchical coordinate system that decomposes a global coordinate $\texttt{pos}$ into multiple levels of finer detail with base B_l. At each level l, the local coordinate for interpolation is computed as:

$$\texttt{pos}^l = \left\lfloor \frac{\texttt{pos}}{\prod_{k=L}^{l-1} B_k} \right\rfloor \bmod B_l. \tag{4}$$

We make a simple improvement to the original strategy to cope with the increase in patch size at inference time compared to that seen by the model during pretraining. The likely substantial increase in resolution, denoted by k, could result in more fine-tuning endeavours. Maintaining the original grid spacing would result in an increase in overall grid numbers, potentially complicating hierarchical pattern capturing and increasing memory/computational demands To address this, we propose to instead *mix and rescale* the bases $\{B_l\}_{l=1}^{L}$ non-linearly according to the ratio (k). The local coordinate at level l is re-calculated as:

$$\texttt{pos}^l = \left\lfloor \frac{\texttt{pos}}{\prod_{k=L}^{l-1} w_k B_k} \right\rfloor \bmod (w_l B_l), \tag{5}$$

subject to $w_1 w_2 \cdots w_L = k$ and $w_1 \geq w_2 \geq \cdots \geq w_L \geq 1$. Solving this constraint yields a set of $\{w_l\}_{l=1}^{L}$ evenly distributed on a log-scale, as will be detailed in *Supplementary*. The formulation by Equation (5) essentially distributes the interpolation pressure across different hierarchies. Low frequencies, which are typically less sensitive to resolution changes, can handle more scaling without significant loss of information (sparser grid partition at lower resolution). High frequencies, on the other hand, are preserved more carefully from being scaled less. Based on Equation (5), the l-th layer of decoder is given by:

$$\ddot{\boldsymbol{\gamma}}_t^l, \ddot{\boldsymbol{\beta}}_t^l = \text{Interp}_{\text{bilinear}}\left(\tilde{\boldsymbol{\gamma}}_t^l, \texttt{pos}^l\right), \text{Interp}_{\text{blinear}}\left(\tilde{\boldsymbol{\beta}}_t^l, \texttt{pos}^l\right),$$
$$\ddot{\mathbf{h}}_t^{l-1} = \text{ReLU}\left(\mathbf{W}^l\left(\left(\ddot{\boldsymbol{\gamma}}_t^l \text{IN}\left(\mathbf{h}_t^l\right) + \ddot{\boldsymbol{\beta}}_t^l\right) + \mathbf{h}_t^l\right) + \mathbf{b}^l\right),$$
$$\mathbf{h}_t^{l-1} = \text{Upsample}_{\text{bilinear}}(\ddot{\mathbf{h}}_t^{l-1}; S_{\text{default}}^{l-1} = 2).$$

3.7 Optimization Strategy

The full model is first pretrained offline then overfitted during inference to adapt to the input video sequence to be compressed. Considering the pre-trained parameters, the "meta-initialization" shared between the encoder and the decoder, the iteratively refined parameter updates, relative to this meta-initialization, are quantized alongside associated side information and then entropy coded into the bitstream. We follow (Lu et al. 2020; Li, Li, and Lu 2023) to aggregate the loss over multiple frames to reduce error propagation and establish the hierarchical quality structure during pretraining, which involves minimizing the following rate-distortion loss within each GOP:

$$\frac{1}{N}\sum_{t=1}^{N}\left(\sum_{l=1}^{L}\left(R(\hat{\mathbf{y}}_t^l) + R(\hat{\mathbf{q}}_{\text{ch};t}^l)\right) + q_{\text{glob};t}\cdot\lambda_t\cdot d(\mathbf{x}_t, \hat{\mathbf{x}}_t)\right), \tag{6}$$

where $d(\cdot,\cdot)$ denotes the distortion, $R(\cdot)$ stands for the rate, and $t \in 1, \cdots, N$ denotes the displaying order of each frame in a GOP. Following (Kim et al. 2024; Leguay et al. 2024), the quantization of hierarchical latents and weight update is approximated by progressively annealed soft-rounding with additive noises when calculating $R(\cdot)$, and by the straight-through estimator (STE) (Minnen and Singh 2020) when rounding them to optimize the distortion metric. Here MSE is used as the distortion metric targeting the best PSNR performance, while additional models are trained by fine-tuning MSE models using $200 \cdot (1 - \text{MS-SSIM}(\mathbf{x}_t, \hat{\mathbf{x}}_t))$ as the distortion metric to yield MS-SSIM-based baselines. We also adopt the mixed distortion loss: $d = 0.7 \times \text{L1} + 0.3 \times (1 - \text{MS-SSIM})$, commonly used in NeRV-based models (Chen et al. 2021; Kwan et al. 2024), with the results provided in the *Supplementary*. During frame-wise overfitting, encoding entails searching for the optimal values of the hierarchical latents, network parameters, and quantization parameters:

$$\sum_{l=1}^{L}\left(R(\hat{\mathbf{y}}_t^l) + R(\Delta\hat{\boldsymbol{\psi}}_t^l) + R(\hat{\mathbf{q}}_{\text{ch};t}^l)\right) + q_{\text{glob};t}\cdot\lambda\cdot d(\mathbf{x}_t, \hat{\mathbf{x}}_t). \tag{7}$$

4 Experiments

Datasets. The proposed PNVC is pre-trained on Vimeo-90k (Xue et al. 2019) and evaluated on UVG (Mercat, Viitanen, and Vanne 2020), MCL-JCV (Wang et al. 2016), and HEVC B-E (Bossen et al. 2013), where the results for HEVC sequences are provided in the *Supplementary*.

Baselines. The PNVC model is compared with **nine** open-sourced state-of-the-art conventional and neural baselines including (i) two conventional codecs - H.265/HEVC Test Model HM 18.0 (Sharman, Sjoberg, and Sullivan 2022) and H.266/VVC Test Model VTM 20.0 (Browne, Ye, and Kim 2023); (ii) four neural codecs - VCT (Mentzer et al. 2022), DCVC-HEM (Li, Li, and Lu 2022), and DCVC-DC (Li, Li, and Lu 2023); (iii) four INR-based codecs - FFNeRV (Lee et al. 2023), HNeRV-Boost (Zhang et al. 2024), C3 (Kim et al. 2024) and HiNeRV (Kwan et al. 2024).Additional comparisons are provided in the *Supplementary*.

Test conditions. Two coding configurations are used: Low Delay (LD) and Random Access (RA), based on the VTM common test conditions (Bossen et al. 2023). The LD setting uses one intra frame per sequence (the first), with subsequent P frames predicted from previous frames only (i.e., GOP=1). In RA mode, each GOP consists of one intra frame and 31 B/P frames, with a maximum latency of 31 frames (i.e., GOP=32) and `IntraPeriod`=32. The RA configuration follows the hierarchical B-frame structure specified in the JVET CTC. We note that these settings do not apply to INR-based codecs, which encode the entire sequences.

Metrics. For each test rate point of a video codec, we calculate the bitrate (in terms of bit/pixel, bpp) and their video quality in terms of PSNR and MS-SSIM in the RGB colorspace. Based on these, we further calculate the Bjøntegaard Delta Rate (BD-rate) (Bjontegaard 2001) using the proposed PNVC (RA) as anchor. We also measured the complexity figures of each tested codec, including the num-

Codec (mode)	UVG		MCL-JCV		Model Complexity (UVG)				Latency
	BD-rate (%)		BD-rate (%)		FPS		Params. (M)	kMACs/pixel	
	(PSNR)	(MS-SSIM)	(PSNR)	(MS-SSIM)	Encoding	Decoding	Full model	Decoding	
HM 18.0 (LD)*	-0.00	-0.00	-0.00	-0.00	6.17	54.3	N/A	N/A	0
HM 18.0 (RA)	-21.21	-17.02	-20.36	-18.10	3.48	33.2	N/A	N/A	31
VTM 20.0 (LD)	-28.94	-28.90	-28.47	-31.35	0.05	30.7	N/A	N/A	0
VTM 20.0 (RA)	-50.08	-45.94	-47.78	-48.02	0.01	23.8	N/A	N/A	31
VCT	+20.12	-42.06	+20.70	-48.57	1.10	0.39	154.2	5336	0
DCVC-HEM	-34.96	-63.05	-33.81	-50.05	0.87	1.51	48.7	1581	0
DCVC-DC	-43.94	-60.83	-43.46	-70.52	0.96	1.29	50.8	1274	0
FFNeRV	+81.50	+50.21	+100.93	+80.31	0.0336 ± 0.013	36.4 ± 2.7	12.1 ± 8.7	36.8 ± 1.6	Full seq.
C3 (Adaptive)	+6.47	+20.88	+8.13	-18.41	0.0015 ± 0.003	17.6 ± 0.001	0.01 ± 0.002	3.3 ± 0.8	30-75
HNeRV-Boost	+2.42	+18.96	+75.8	+225.6	0.0155 ± 0.009	47.6 ± 24.3	11.6 ± 8.9	520.0 ± 359.0	Full seq.
HiNeRV	-25.42	-50.90	-1.64	-22.34	0.0157 ± 0.009	19.7 ± 13.5	30.3 ± 27.1	682.8 ± 595.1	Full seq.
PNVC (LD)	**-33.58**	**-56.93**	**-32.97**	**-57.12**	0.014	25.3	21.8	101.1	0
PNVC (RA)	**-41.93**	**-63.41**	**-40.29**	**-61.81**	0.011	22.6	21.8	101.1	31

Table 1: BD-rate results w.r.t H.265/HEVC Test Model HM 18.0 (LD), denoted by *, together with the complexity figures including kMACs/pixel, FPS, and model size, on the UVG dataset. Complexity figures for INR benchmarks are reported with $\pm$ to indicate the range as their sizes and configurations change for different bitrates.

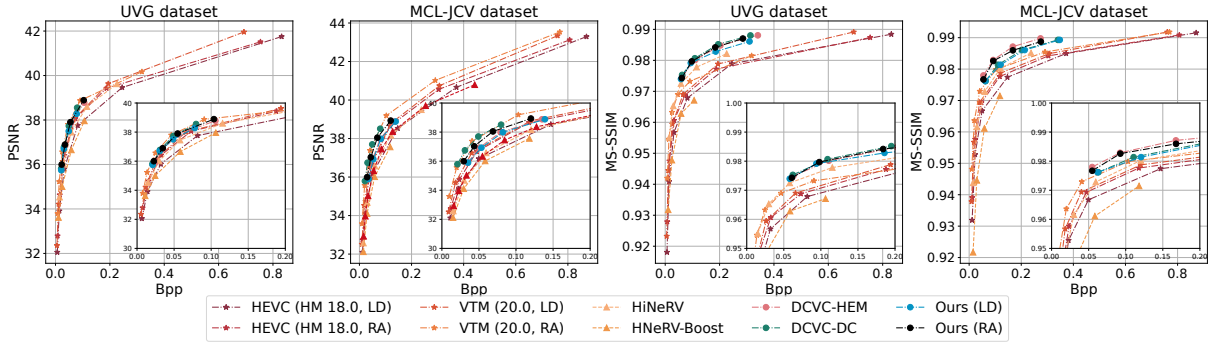

Figure 4: RD performance comparison on the UVG and MCL-JCV datasets. Here, the two best performers per type is plotted.

ber of model parameters, the decoding complexity (MACs/pixel) and average encoding and decoding speeds (FPS).

5 Results

5.1 Quantitative Results

The BD-rate results in Table 1 show that PNVC (both LD and RA) outperforms state-of-the-art INR-based codecs and proves competitive with leading conventional and neural video codecs. For example, PNVC (RA) outperforms HiNeRV by 25.42% and 50.9% in BD-rate in terms of PSNR on the UVG and MCL-JCV dataset, respectively. It should be noted that PNVC embodies latency constraints (31 frames for the RA mode and 0 for the LD mode, the same as HM and VTM), whereas other INR models do not. These demonstrate the strong rate-distortion performance of the proposed model. Among all codecs tested, the next-generation standard VTM (RA) remains to be the best performing baseline.

5.2 Qualitative Results

We present qualitative visual comparisons between frames reconstructed by HiNeRV (the best INR-based video codec) and the proposed PNVC in Figure 5. The visual results show that our method is able to yield visually pleasing reconstructions, accurately capturing the fine texture details and fast/-complex motions present in the frame. Please see *Supplementary* for more thorough comparisons.

5.3 Complexity Performance

The complexity figures of PNVC and other benchmark codecs, including FPS to fully encode/decode a frame, full model size (number of parameters) and decoding complexity (MACs/pixel), are summarized in Table 1. These are measured by averaging over a GOP of 1080P videos, including a full roll-out of the entropy coding process, on a PC with an NVIDIA 3090 GPU and an Intel Core i7-12700 CPU. It can be observed that our model achieves a higher decoding speed, comparable to the conventional

Figure 5: Visual quality comparison between HiNeRV and the proposed PNVC reconstructed content at similar bitrates.

and INR baselines and considerably faster than all the neural video codecs. The PNVC decoder is similar in size to INR models and much smaller than neural codecs. The well-roundness of our proposed model is illustrated by the Radar plot, as shown in Figure 1, in which decoding speed, encoding latency, BD-rate in PSNR and MS-SSIM of the selected, top-performing baselines are visualized. The proposed PNVC demonstrates the best overall rate-distortion-complexity trade-off, although its encoding speed is relatively low, similar to other INR-based codecs, which limits real-time deployment compared to autoencoder-based models and VTM. Nonetheless, its compatibility with Low Delay and Random Access modes, low decoding complexity, and strong rate-quality performance offer significant potential for streaming scenarios where encoding in real-time is unnecessary and could be performed in large data centers. A more detailed breakdown on the computational and memory complexities is provided in the *Supplementary*.

5.4 Ablation Studies

To validate the contribution of each design component in PNVC, we performed ablations with various model variants. As summarized in Table 2, all tested variants underperform compared to PNVC on both datasets, demonstrating the effectiveness of each design component.

Reparameterization. We tested the effectiveness of reparameterization by removing it from pre-training (V1.1), performing online overfitting with the pre-trained model but disregarding the overparameterized basis (V1.2), removing the reparameterization for MLP (V1.3), and removing the degenerated token mixer component (V1.4).

Optimization. We also validated the adaptive quality control parameter $q_{\text{glob};t}$ in both the LD and RA coding configurations by removing them from the optimization process (V2.1 and V2.2, respectively). The performance of pre-training only without overfitting is ablated in (V2.3) and (V2.4).

Entropy model. We adopted the original quadtree-chunking strategy instead of using uneven channel grouping (V3.1) and replaced temporal feature modulation with concatenation (V3.2) to ablate our proposed modifications.

Decoder. We respectively replaced the adaptive scaled hierarchical encoding (V4.1) with the original hierarchical grading scheme (overfitted for the same number of steps), the rescaling feature $\{\boldsymbol{\gamma}_t^l\}_{l=1}^L$ (V4.2) or the shifting feature $\{\boldsymbol{\beta}_t^l\}_{l=1}^L$ (V4.3) to quantify their contributions to the overall performance gain.

Version	Ablation Option	UVG	MCL
(V1.1)	✗ re-param. @ pre-train.	5.23	5.35
(V1.2)	✗ re-param. @ overfit.	3.01	2.99
(V1.3)	✗ re-param. @ fully-connected	1.97	1.89
(V1.4)	✗ re-param. @ token-mix	3.39	3.38
(V2.1)	✗ adaptive QP (LD)	3.77	3.24
(V2.2)	✗ adaptive QP (RA)	5.35	5.29
(V2.3)	✗ overfitting (LD)	7.95	7.95
(V2.4)	✗ overfitting (RA)	9.47	9.34
(V3.1)	✗ uneven channel grouping	1.76	1.91
(V3.2)	✗ modulation ✓ concat.	3.39	3.57
(V4.1)	✗ scaled hierarchical griding	1.21	1.16
(V4.2)	✗ rescaling $\boldsymbol{\gamma}$	0.75	0.76
(V4.3)	✗ shifting $\boldsymbol{\beta}$	0.89	0.91

Table 2: Ablation study results in terms of BD-rate (%) w.r.t the full PNVC codec. Positive values indicate coding loss.

6 Conclusion

We propose PNVC, an INR-based video codec that combines pretrained, autoencoder-based compression methods with coordinate-based overfitting. PNVC supports Low Delay and Random Access coding modes and achieves competitive rate-distortion performance with fast decoding speed. Through innovations including reparameterized visual tuning, hierarchical quality control, modulation-based entropy modeling, and scale-adaptive positional embedding, PNVC outperforms VTM 20.0 (LD) and HiNeRV, matches DCVC-DC in compression efficiency, and maintains a low decoding complexity (<200k MACs/pixel) and a decoding speed of >20 FPS @1080P. Flexible latency configurations (LD and RA) improve the practicality of INR-based video codecs in streaming scenarios. However, PNVC requires relatively high computational and memory resources during overfitted encoding. Future work should explore parameter-efficient tuning methods (Chen et al. 2024) to address this limitation.

Acknowledgements

This work is funded by the UKRI MyWorld Strength in Places Programme (SIPF00006/1). We also thank the support by the Advanced Computing Research Centre, University of Bristol, for providing the computational facilities.

References

Agustsson, E.; Minnen, D.; Johnston, N.; Balle, J.; Hwang, S. J.; and Toderici, G. 2020. Scale-space flow for end-to-end optimized video compression. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 8503–8512.

Bai, Y.; Dong, C.; Wang, C.; and Yuan, C. 2023. PS-NeRV: Patch-wise stylized neural representations for videos. In *IEEE Int. Conf. Image Process.*, 41–45. IEEE.

Ballé, J.; Minnen, D.; Singh, S.; Hwang, S. J.; and Johnston, N. 2018. Variational Image Compression with a Scale Hyperprior. In *Int. Conf. Learn. Represent.*

Bjontegaard, G. 2001. Calculation of average PSNR differences between RD-curves. *ITU SG16 Doc. VCEG-M33*.

Bossen, F.; Boyce, J.; Suehring, K.; Li, X.; and Seregin, V. 2023. VTM Common Test Conditions and Software Reference Configurations for SDR Video. In *the JVET meeting*, JVET-T2010.

Bossen, F.; et al. 2013. Common test conditions and software reference configurations. *JCTVC-L1100*, 12(7): 1.

Bross, B.; Wang, Y.-K.; Ye, Y.; Liu, S.; Chen, J.; Sullivan, G. J.; and Ohm, J.-R. 2021. Overview of the Versatile Video Coding (VVC) Standard and Its Applications. *IEEE Trans. Circuit Syst. Video Technol.*, 31(10): 3736–3764.

Browne, A.; Ye, Y.; and Kim, S. H. 2023. Algorithm description for Versatile Video Coding and Test Model 19 (VTM 19). In *the JVET meeting*, JVET-AC2002. ITU-T and ISO/IEC.

Chen, H.; Gwilliam, M.; Lim, S.-N.; and Shrivastava, A. 2023. HNeRV: A Hybrid Neural Representation for Videos. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 10270–10279.

Chen, H.; He, B.; Wang, H.; Ren, Y.; Lim, S. N.; and Shrivastava, A. 2021. NeRV: Neural Representations for Videos. *Adv. Neural Inform. Process. Syst.*, 34: 21557–21568.

Chen, Z.; Zhou, L.; Hu, Z.; and Xu, D. 2024. Group-aware Parameter-efficient Updating for Content-adaptive Neural Video Compression. In *ACM Int. Conf. Multimedia*, 11022–11031.

Child, R. 2021. Very Deep {VAE}s Generalize Autoregressive Models and Can Outperform Them on Images. In *Int. Conf. Learn. Represent.*

Ding, X.; Zhang, X.; Han, J.; and Ding, G. 2022. Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 11963–11975.

Dupont, E.; Golinski, A.; Alizadeh, M.; Teh, Y. W.; and Doucet, A. 2021. COIN: Compression with Implicit Neural representations. In *ICLR Workshop on Neural Compression: From Information Theory to Applications*.

Gomes, C.; Azevedo, R.; and Schroers, C. 2023. Video Compression with Entropy-Constrained Neural Representations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 18497–18506.

He, B.; Yang, X.; Wang, H.; Wu, Z.; Chen, H.; Huang, S.; Ren, Y.; Lim, S.-N.; and Shrivastava, A. 2023. Towards Scalable Neural Representation for Diverse Videos. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 6132–6142.

He, D.; Yang, Z.; Peng, W.; Ma, R.; Qin, H.; and Wang, Y. 2022. ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 5718–5727.

Ho, Y.-H.; Chang, C.-P.; Chen, P.-Y.; Gnutti, A.; and Peng, W.-H. 2022. CANF-VC: Conditional Augmented Normalizing Flows for Video Compression. In *Eur. Conf. Comput. Vis.*, 207–223. Springer.

Hu, Z.; Lu, G.; and Xu, D. 2021. FVC: A New Framework towards Deep Video Compression in Feature Space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 1502–1511.

Hu, Z.; and Xu, D. 2023. Complexity-guided Slimmable Decoder for Efficient Deep Video Compression. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 14358–14367.

Jiang, W.; Li, J.; Zhang, K.; and Zhang, L. 2024. ECVC: Exploiting Non-Local Correlations in Multiple Frames for Contextual Video Compression. *arXiv preprint arXiv:2410.09706*.

Kim, H.; Bauer, M.; Theis, L.; Schwarz, J. R.; and Dupont, E. 2024. C3: High-performance and low-complexity neural compression from a single image or video. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 9347–9358.

Kwan, H. M.; Gao, G.; Zhang, F.; Gower, A.; and Bull, D. 2024. HiNeRV: Video Compression with Hierarchical Encoding-based Neural Representation. *Adv. Neural Inform. Process. Syst.*, 36: 72692–72704.

Lee, J. C.; Rho, D.; Ko, J. H.; and Park, E. 2023. FFNeRV: Flow-Guided Frame-Wise Neural Representations for Videos. In *ACM Int. Conf. Multimedia*, 7859–7870.

Lee, J. C.; Rho, D.; Nam, S.; Ko, J. H.; and Park, E. 2024. Coordinate-aware modulation for neural fields. In *The Twelfth International Conference on Learning Representations*.

Leguay, T.; Ladune, T.; Philippe, P.; and Déforges, O. 2024. Cool-chic video: Learned video coding with 800 parameters. *arXiv preprint arXiv:2402.03179*.

Li, J.; Li, B.; and Lu, Y. 2021. Deep Contextual Video Compression. *Adv. Neural Inform. Process. Syst.*, 34: 18114–18125.

Li, J.; Li, B.; and Lu, Y. 2022. Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression. In *ACM Int. Conf. Multimedia*, 1503–1511.

Li, J.; Li, B.; and Lu, Y. 2023. Neural Video Compression with Diverse Contexts. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 22616–22626.

Li, J.; Li, B.; and Lu, Y. 2024. Neural Video Compression with Feature Modulation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 26099–26108.

Li, Z.; Wang, M.; Pi, H.; Xu, K.; Mei, J.; and Liu, Y. 2022. E-NeRV: Expedite neural video representation with disentangled spatial-temporal context. In *Eur. Conf. Comput. Vis.*, 267–284. Springer.

Lin, S.; Lyu, P.; Liu, D.; Tang, T.; Liang, X.; Song, A.; and Chang, X. 2024. MLP Can Be A Good Transformer Learner. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19489–19498.

Lu, G.; Cai, C.; Zhang, X.; Chen, L.; Ouyang, W.; Xu, D.; and Gao, Z. 2020. Content Adaptive and Error Propagation Aware Deep Video Compression. In *Eur. Conf. Comput. Vis.*, 456–472.

Lu, G.; Ouyang, W.; Xu, D.; Zhang, X.; Cai, C.; and Gao, Z. 2019. DVC: An End-To-End Deep Video Compression Framework. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 11006–11015.

Lu, M.; Duan, Z.; Zhu, F.; and Ma, Z. 2024. Deep Hierarchical Video Compression. In *AAAI*, 8859–8867.

Ma, D.; Zhang, F.; and Bull, D. R. 2020. MFRNet: A New CNN Architecture for Post-Processing and In-loop Filtering. *IEEE Journal of Selected Topics in Signal Processing*, 15(2): 378–387.

Mehta, I.; Gharbi, M.; Barnes, C.; Shechtman, E.; Ramamoorthi, R.; and Chandraker, M. 2021. Modulated Periodic Activations for Generalizable Local Functional Representations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 14214–14223.

Mentzer, F.; Toderici, G. D.; Minnen, D.; Caelles, S.; Hwang, S. J.; Lucic, M.; and Agustsson, E. 2022. VCT: A Video Compression Transformer. *Advances in Neural Information Processing Systems*, 35: 13091–13103.

Mercat, A.; Viitanen, M.; and Vanne, J. 2020. UVG dataset: 50/120fps 4K sequences for video codec analysis and development. In *ACM Int. Conf. Multimedia*, 297–302.

Minnen, D.; and Singh, S. 2020. Channel-wise autoregressive entropy models for learned image compression. In *2020 IEEE International Conference on Image Processing*, 3339–3343. IEEE.

Peng, T.; Gao, G.; Sun, H.; Zhang, F.; and Bull, D. 2024. Accelerating Learnt Video Codecs with Gradient Decay and Layer-Wise Distillation. In *Picture Coding Symposium*.

Ranjan, A.; and Black, M. J. 2017. Optical Flow Estimation Using A Spatial Pyramid Network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 4161–4170.

Ruan, H.; Shao, Y.; Yang, Q.; Zhao, L.; and Niyato, D. 2024. Point Cloud Compression with Implicit Neural Representations: A Unified Framework. *arXiv preprint arXiv:2405.11493*.

Sharman, C. R. K.; Sjoberg, R.; and Sullivan, G. 2022. High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Improved Encoder Description Update 16. In *the JVET meeting*, JVET-Y1002. ITU-T and ISO/IEC.

Shi, K.; Zhou, X.; and Gu, S. 2024. Improved Implicit Neural Representation with Fourier Reparameterized Training. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 25985–25994.

Strümpler, Y.; Postels, J.; Yang, R.; Gool, L. V.; and Tombari, F. 2022. Implicit Neural Representations for Image Compression. In *Eur. Conf. Comput. Vis.*, 74–91.

Sullivan, G. J.; Ohm, J.; Han, W.; and Wiegand, T. 2012. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Trans. Circuit Syst. Video Technol.*, 22(12): 1649–1668.

Tang, C.; Sheng, X.; Li, Z.; Zhang, H.; Li, L.; and Liu, D. 2024. Offline and Online Optical Flow Enhancement for Deep Video Compression. In *AAAI*, volume 38, 5118–5126.

van Rozendaal, T.; Huijben, I.; and Cohen, T. S. 2021. Overfitting for Fun and Profit: Instance-Adaptive Data Compression. In *Int. Conf. Learn. Represent.*

Wang, G.-H.; Li, J.; Li, B.; and Lu, Y. 2023. EVC: Towards Real-Time Neural Image Compression with Mask Decay. In *Int. Conf. Learn. Represent.*

Wang, H.; Gan, W.; Hu, S.; Lin, J. Y.; Jin, L.; Song, L.; Wang, P.; Katsavounidis, I.; Aaron, A.; and Kuo, C.-C. J. 2016. MCL-JCV: a JND-based H. 264/AVC video quality assessment dataset. In *IEEE Int. Conf. Image Process.*, 1509–1513.

Wiegand, T.; Sullivan, G. J.; Bjøntegaard, G.; and Luthra, A. 2003. Overview of the H.264/AVC Video Coding Standard. *IEEE Trans. Circuit Syst. Video Technol.*, 13(7): 560–576.

Xiang, J.; Tian, K.; and Zhang, J. 2022. MIMT: Masked Image Modeling Transformer for Video Compression. In *Int. Conf. Learn. Represent.*

Xue, T.; Chen, B.; Wu, J.; Wei, D.; and Freeman, W. T. 2019. Video Enhancement with Task-oriented Flow. *International Journal of Computer Vision*, 127: 1106–1125.

Yan, N.; Liu, D.; Li, H.; Li, B.; Li, L.; and Wu, F. 2018. Convolutional Neural Network-Based Fractional-Pixel Motion Compensation. *IEEE Trans. Circuit Syst. Video Technol.*, 29(3): 840–853.

Yang, H.; Oh, S.; and Park, E. 2024. Parameter-Efficient Instance-Adaptive Neural Video Compression. *arXiv preprint arXiv:2405.08530*.

Yang, Y.; and Mandt, S. 2023. Computationally-Efficient Neural Image Compression with Shallow Decoders. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 530–540.

Zhang, X.; Yang, R.; He, D.; Ge, X.; Xu, T.; Wang, Y.; Qin, H.; and Zhang, J. 2024. Boosting Neural Representations for Videos with a Conditional Decoder. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2556–2566.

Zhang, Y.; van Rozendaal, T.; Brehmer, J.; Nagel, M.; and Cohen, T. 2021. Implicit Neural Video Compression. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*.