Enhanced Neural Video Representation Compression across Extreme Complexity and Quality Scales

Ho Man Kwan[1⋆], Tianhao Peng[1⋆], Fan Zhang[1], Mike Nilsson[2], Andrew Gower[2], and David Bull[1]

[1] Visual Information Lab, University of Bristol, UK
{hm.kwan, tianhao.peng, fan.zhang, dave.bull@bristol.ac.uk}
[2] Network Connectivity Services Research, BT, UK
{mike.nilsson, andrew.p.gower}@bt.com

Abstract. Implicit neural representations (INRs) have recently emerged as a promising approach to video compression, delivering competitive rate-distortion performance alongside rapid decoding. However, existing neural video codecs struggle to balance complexity and scalability. Lightweight models often suffer from degraded compression performance when scaled to different bitrate/quality levels, whereas high-performance models exhibit limited scalability, as their model complexity typically increases with quality. This lack of a unified architecture capable of maintaining consistent complexity across a wide range of bitrates severely limits their diverse real-world deployment. To address these challenges, we introduce NVRC++[1], a novel INR-based video codec that utilizes a lightweight INR with multiple high-resolution feature grids, providing high scalability at any given complexity level. This is paired with an optimization framework that enables efficient overfitting on high-resolution grids for long video sequences, thereby exploiting spatio-temporal redundancies without prohibitive computational or memory overhead. Additionally, an advanced entropy model is designed for efficiently compressing the high-dimensional grid parameters. As a result, NVRC++ provides four complexity levels (from 7kMACs/pixel to 360kMACs/pixel), each spanning wide bitrate and quality ranges while supporting real-time decoding. The experimental results show that NVRC++ offers a much faster decoding speed (up to 7.6x) compared to the SOTA INR-based video codec, NVRC, while delivering comparable performance.

Keywords: Video compression · Implicit neural representation

1 Introduction

Recent work in neural video compression has delivered coding performances that are comparable to the latest standard-based video codecs [6]. Many of

⋆ Equal contribution.
[1] Project page: https://hmkx.github.io/nvrc-pp/

these learning-based methods [33, 36] are based on autoencoder networks, which transform input videos into a latent domain before performing quantization and entropy coding. Such approaches typically learn a generalized model from large training datasets and are deployed during the inference phase to compress an unseen video. Due to the large capacity required for a generalized model, these methods are often associated with high computational complexity, which hinders their application in real-world scenarios.

More recently, implicit neural representations (INR) [8,12,22,23] have provided an attractive alternative solution for video compression. Typically, an INR-based video codec utilizes an overfitted neural network to represent a video and further employs model compression and/or entropy coding techniques to encode model parameters, including layer weights [8,34] and feature grids [7,21,22,28]. Although it is relatively slow when involving an overfitting process in encoding, these INR-based methods can effectively exploit spatio-temporal redundancies in an implicit manner via the shared neural network layers that are subsequently used for decoding video frames. The latest neural representation based codecs [23] have shown competitive rate-quality performance compared to conventional codecs such as HEVC HM [44] and VVC VTM [5], as well as the best performing autoencoder-based neural video codecs [31, 32].

While demonstrating promising compression performance, a critical gap remains in the research community with lightweight and high-performance INR-based/overfitted codecs. Specifically, most performance focused methods [7, 8, 22, 23, 34] suffer from poor scalability, as they require different model configurations for various target quality levels or rate points. Consequently, their computational complexity increases with video quality, resulting in significant decoding computational overhead at high bitrates. This low scalability characteristic prevents their application in real-world scenarios, while a practical neural video codec should ideally achieve a wide bitrate/quality range with a single-size model. Conversely, lightweight overfitted codecs can achieve rate scalability with a single or narrow complexity range, but they yield inferior compression performance due to their lightweight models [21, 24, 29] and the independent coding of patches [21]. There are also overfitted INR-based codecs which do not benefit much from the overfitting process [11], which can be effective for exploiting global spatial-temporal redundancy [22,23]. This lack of a unified architecture capable of reaching diverse complexity levels while maintaining high rate-quality scalability prevents their application in diverse real-world scenarios.

An efficient way to bridge this complexity and scalability gap is the adoption of high-dimensional feature grids [21,24,29] for the high performance codec, but with diverse complexity settings. By storing dynamic spatio-temporal features directly, these grids allow an INR to represent higher-quality details without increasing model complexity. However, directly deploying high-dimensional feature grids can be suboptimal. While recent work shows that high performance INR-based codecs offer optimal performance when encoding an entire long video sequence with a single model [22, 54, 56], optimizing INRs with high-resolution grids for long sequences requires substantial computational and memory resources, particularly

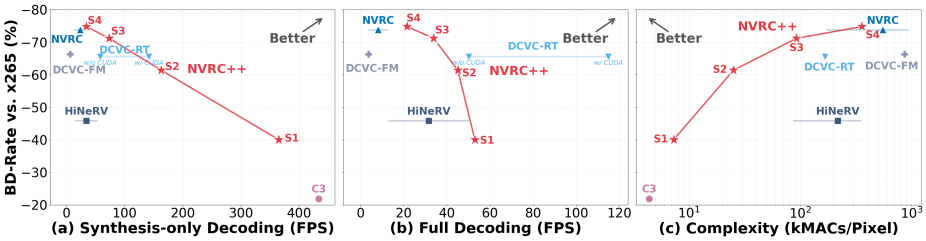

Fig. 1: The trade off between decoding complexity (in terms of FPS and MACs/pixel) and coding performance (in BD-rate against x265 (*veryslow*)) [52] of the proposed method and existing neural video codecs. **(a)** BD-rate vs. ***synthesis-only*** decoding speed, where for over-fitted approaches the entropy decoding step is excluded as it is a process separate from frame decoding [21, 23]; **(b)** BD-rate vs. ***full*** decoding speed, where the entropy decoding step is included (C3 [21] is omitted here as no entropy encoder/decoder is available for it, and for DCVC-RT [18] we report the speed both with and without its custom CUDA kernels); **(c)** BD-rate vs. computational complexity in kMACs/pixel. NVRC++ achieves a superior complexity-compression trade-off, delivering state-of-the-art performance across an unprecedented quality range at every complexity level from 7 to 360 kMAC/pixel. Note that we have not utilized custom CUDA kernel, while still achieving very fast decoding speed.

when the interdependency between grid features must be modeled, for instance, by an entropy model. To manage this, existing works typically use high-resolution grids but encode the video in individual patches [21] or short video clips [11, 24]. Furthermore, while high-resolution grids improve performance at high bitrates, they may impact low-bitrate performance because these grids increase the data volume, resulting in sub-optimal compression performance [24].

To address the scalability issue and bridge the performance gap across the entire complexity spectrum, this paper proposes a novel neural video representation compression framework, NVRC++. Targeting high scalability, it utilizes a neural representation consisting of multiple high-resolution feature grids, allowing a single model to achieve a wide bitrate/quality range and fast decoding with every complexity configuration. To avoid suboptimal performance when directly deploying high-resolution grids (as mentioned above), we also developed an enhanced optimization framework inspired by the hierarchical coding structure in conventional video codecs to allow the optimization of these high-dimensional grids under computational resource constraints. This improves performance across the entire bitrate/quality range and the decoding speed. Furthermore, to reduce the additional storage required by high-resolution grids, NVRC++ is integrated with an improved multi-dimensional grid entropy model to better exploit spatio-temporal redundancy within videos and between grids of different scales. Finally, our framework is built upon HiNeRV++, an improved INR designed for better performance and efficiency by leveraging multi-reference context from higher-resolution feature grids. The main contributions are summarized:

– **A novel INR-based codec with high scalability** that supports multiple complexity levels. By utilizing high-resolution features, it achieves high scala-

bility and fast decoding at each complexity level while maintaining competitive coding performance.
- **An efficient optimization framework** inspired by the hierarchical coding structures in conventional video codecs; it effectively exploits spatio-temporal redundancies within parameters, enables the optimization of high dimensional grids under computational constraints, and achieves lower decoding latency.
- **An advanced multi-dimensional grid entropy model**, which offers an efficient compression method for high dimensional grid parameters by leveraging multiple prior information, yielding high entropy encoding and decoding speeds.
- **An enhanced INR, HiNeRV++**, which supports real-time decoding across various complexity configurations, achieving a wide bitrate range within each configuration and contributing to superior compression performance.

The NVRC++ codecs (four variants at different complexity levels) have been benchmarked against state-of-the-art conventional, autoencoder-based, and INR-based video codecs. The results (as summarized in Fig. 1) show that all NVRC++ codecs offer an excellent trade off between decoding complexity and coding performance. Specifically, NVRC++ (S3) achieves a much faster decoding speed (7.6 times) compared to the SOTA INR-based codec, NVRC, while maintaining comparable performance.

2 Related Work

Neural video compression has emerged as a prominent research field in recent years, leveraging deep learning techniques to achieve improved coding efficiency. Based on the variational autoencoder framework and end-to-end optimization that originate from learned image compression [2], DVC [36] was the first end-to-end optimized video codec to adapt a residual coding-based architecture for traditional codecs, where each component is replaced by its learnable alternative. Later works achieved enhanced coding efficiency by integrating a variety of techniques such as 3D modeling [13], conditional coding [17, 26, 33, 35], conditional residual coding [9] and Transformer-based architectures [10,38,53]. The SOTA neural video codecs have demonstrated superior capability [18–20, 45, 50], offering performance comparable to the latest standard video codecs [4, 48].

Neural representation for video compression. When applied to video coding, an INR [40, 43, 47] learns a mapping between coordinate inputs and the corresponding target pixel values for a single instance of video [34, 47] or a set of videos [34]. This approach effectively converts video compression into a model compression task, where the network represents videos with model weights. The pioneering work NeRV [8] established a three-step compression pipeline: model pruning with fine-tuning to reduce model size, weight quantization to lower precision, and entropy coding to minimize statistical correlations. Subsequent work focused on enhancing different stages/components in this pipeline, including entropy minimization [12, 25, 37, 54], architectural innovations [34, 56], and flow-based motion compensation [14, 28, 55]. To further improve coding efficiency,

single/multiple resolution feature grids were introduced to effectively represent adaptive visual content information in the latent space [21, 28, 29]. A significant milestone was the NVRC framework [23], which proposed an end-to-end optimized compression framework for neural video representations. This has been reported to be the best INR-based video codec to date, offering comparable performance to VVC VTM (Random Access mode).

3 Method

Preliminaries. In the context of INR-based video compression, most methods utilize an overfitting approach initialized by NeRV [8]; i.e., neural representations are used for coding videos in an instance-adaptive manner, where a neural network is trained from scratch to learn the coordinate-to-output mapping. Specifically, given a patch coordinate (i, j, k), where $0 \leq i < \frac{W}{W_{patch}}$, $0 \leq j < \frac{H}{H_{patch}}$, and $0 \leq k < \frac{T}{T_{patch}}$, of a 3D patch $\hat{V}_{patch}$ with size $T_{patch} \times H_{patch} \times W_{patch}$ within the input video $V^{gt} \in \mathbb{R}^{T \times H \times W \times C}$, the corresponding normalized pixel coordinate x within that patch is first embedded into high dimensions, which is commonly achieved through positional encodings [8, 41]:

$$\gamma(x) = \text{Concat}\big(\sin(2^0\pi x), \cos(2^0\pi x), \ldots, \sin(2^{L_{freq}-1}\pi x), \cos(2^{L_{freq}-1}\pi x)\big), \tag{1}$$

where L_{freq} is the number of frequency bands. Alternatively, recent works utilize feature grids [21, 22, 27–29], where a set of learnable feature embeddings $\mathcal{F}$ is stored in multi-resolution grids. The corresponding feature f is extracted via trilinear interpolation at each level and concatenated across levels:

$$f = \text{Concat}\big(\text{Interp}(x, \mathcal{F}_0), \ldots, \text{Interp}(x, \mathcal{F}_{L_{grid}-1})\big), \tag{2}$$

where $\mathcal{F}_l$ represents the grid at resolution level l, and L_{grid} is the number of grid levels. Typically, these grids are implemented with multiple resolutions, where the grid dimensions are significantly lower than the original video dimensions to maintain a compact representation. By performing a forward pass at every coordinate using the embedded features, a reconstruction, V, of the video V^{gt} can be decoded. The neural network parameters θ can thus be treated as a compact representation of the signal, and techniques such as quantization and entropy coding [8, 12] are then used to further compress θ into a bitstream. This process is typically lossy, and the quality is subject to both the representational capability and the compression ratio. The objective of INR-based video coding can thus be considered as a classical rate-distortion optimization problem based on neural compression [1]:

$$\text{argmin}_\theta \mathcal{L} = \text{argmin}_\theta \left(R(\hat{\theta}) + \lambda D(V, V^{gt}) \right). \tag{3}$$

Here, θ and $\hat{\theta}$ are the original and quantized model parameters. R and D are the measurements of rate and distortion, respectively.

While the latest INR-based video codecs demonstrate promising compression performance [7, 23, 54], there is a major limitation that hinders practical applications: their model sizes typically vary with bitrate and quality. This implies that the model complexity scales with the target bitrate/quality, which is a critical problem: a large INR-based codec [22, 23] could be as costly as the end-to-end autoencoder-based models [30, 36], which are trained as a generic model. In this case, overfitted INR-based codecs could lose their natural advantage in low decoding complexity due to instance adaptive coding. Moreover, having a decoder with fixed or nearly fixed complexity could be beneficial for both software and hardware implementations.

Compared to NeRV-based methods, there are also approaches that utilize lightweight neural networks together with high-resolution feature grids, e.g., COOL-CHIC [29] and C3 [21], for encoding videos. These approaches can achieve a wide range of quality with a single representation due to the use of high-resolution features, which alleviates the need for a high-capacity decoder network to represent the fine-detailed visual features. However, these approaches are significantly outperformed by the best performing NeRV-based methods [22, 23] in terms of compression performance. This may be due to (i) the tiny decoders or entropy models used [21, 29], which limit expressivity for exploiting the spatio-temporal redundancies within signals and modeling the data distributions, and (ii) the strategy that partitions videos into independent chunks for coding [21], which allows training with high dimensional feature grids within the limited hardware memory but overlooks the redundancy across chunks, as the chunks are independently coded.

3.1 NVRC++

In this context, we propose NVRC++, a novel neural video representation compression framework (shown in Fig. 2) that advances INR-based video compression. Our framework significantly extends NVRC [23], a state-of-the-art INR-based codec, with four key innovations. (i) NVRC++ utilizes INRs with higher-resolution feature grids to decouple model complexity from reconstruction quality. This allows a single model configuration to span a wide bitrate range at fixed complexity, enabling adaptation to diverse hardware constraints; (ii) an enhanced optimization framework is employed to facilitate the training of the INR with high dimensional grids while performing compression at the same time; (iii) improved entropy models are designed to exploit the spatial and temporal redundancies between grid parameters more effectively; and (iv) an enhanced lightweight INR, HiNeRV++, which supports high reconstruction quality at different complexities while achieving fast decoding.

Following NVRC [23], in NVRC++, an INR with parameters $\theta = \{\theta_{grid}, \theta_{layer}\}$ is used for video coding, where θ_{grid} and θ_{layer} are the feature grids and network layer parameters, respectively. These parameters are optimized for reconstructing the video with the rate-distortion constraint, where the corresponding quantized parameters $\hat{\theta} = \{\hat{\theta}_{grid}, \hat{\theta}_{layer}\}$ are coded into the bitstream. Optimizing these parameters also involves quantization and entropy model parameters ϕ and their

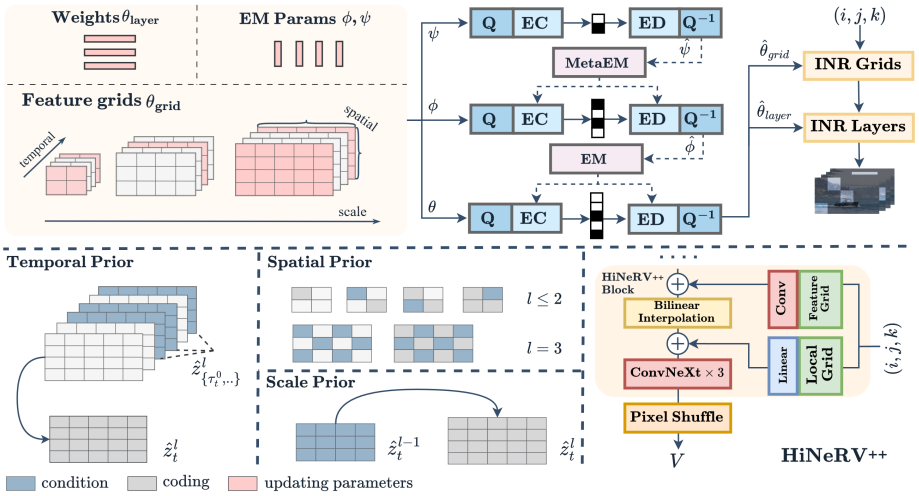

Fig. 2: The overall NVRC++ framework. It contains a hierarchical parameter coding structure, while also contains an advance grid entropy model based on temporal, spatial and scale priors, and the improved INR network, HiNeRV++. It also features random sampling for updating only partial parameters for more efficient training process.

quantized parameters $\hat{\phi}$, as well as the meta compression parameters ψ (and $\hat{\psi}$), where all these parameters form a hierarchical parameter structure and are jointly optimized in an end-to-end manner. We follow [23] for the model compression framework, but with an improved optimization process (Sec. 3.2), a grid entropy model with multiple priors (Sec. 3.3), and an enhanced lightweight INR (Sec. 3.4).

3.2 Overfitting with high-resolution grid features

Although deploying INRs with high resolution grids offers better scalability, naively applying them will introduce higher complexity and a larger memory footprint, particularly when integrated with the contextual entropy models required for competitive compression. Specifically, when an autoregressive model is employed as the entropy model using techniques such as 3D convolutions [21], all dependent parameters are processed simultaneously within a large grid tensor. This results in substantial computational and memory overhead. We address this limitation by introducing an *in-parameter coding structure*. This structure models the dependency in a manner following conventional coding structures while making optimization based on overfitting feasible. In addition, we found that the model performance drops, especially at low bitrates, when the high resolution grids are introduced to the INR model directly. We propose to use a *feature grid masking* technique, which regularizes and improves model performance.

In-parameter coding structure. Inspired by the coding structures used in both conventional and neural video coding, we introduce an *in-parameter coding structure* that partitions the feature grid parameters into temporal slices and performs coding by leveraging the dependency. Specifically, for a single resolution

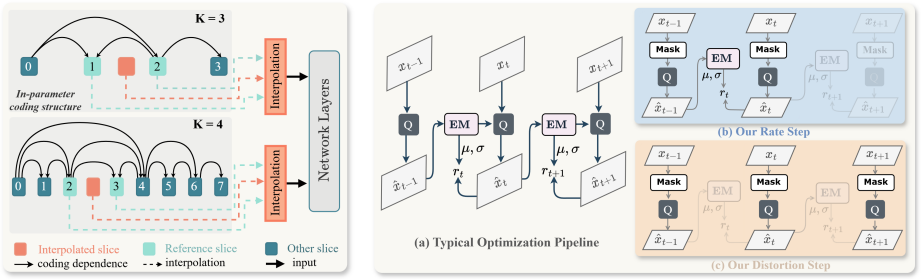

Fig. 3: (**Left**) Proposed in-parameter coding and multi-reference inputs. Grid parameters are organized into temporal slices and compressed using a Hierarchical-B structure to leverage inter-slice dependencies. Unlike standard practices that rely solely on interpolated features, HiNeRV++ utilizes neighboring raw slices as multi-reference context to enhance reconstruction. While $K = 3/4$ is shown for illustration, our implementation scales up to $K = 10$. (**Right**) Optimization pipeline. To avoid the propagation issues of typical coding structure in overfitted codecs, we: (1) use non-conditional quantization to decouple slice optimization; (2) apply decoupled rate-distortion steps with random sampling (sampled components are highlighted); and (3) employ feature masking to improve the optimization with high-resolution grids. Low-delay coding is shown for demonstration, though Hierarchical-B is used in practice.

grid parameters $z \in \mathbb{R}^{T_{\text{grid}} \times H_{\text{grid}} \times W_{\text{grid}} \times C_{\text{grid}}}$ (and the corresponding quantized parameters $\hat{z}$), we divide it into temporal slices $\{z_t \mid 0 \leq t < T_{\text{grid}}\}$ and utilize a contextual entropy model for modeling the conditional distribution $p(\hat{z}_t | \hat{z}_{\tau_t^0}, ...)$, which is subject to the rate–distortion objective. Here, we utilize the hierarchical B-frame coding structure with K levels (an example is shown in Fig. 3 (Left)), where the number of temporal slices at each level is $1, 1, \ldots, 2^{K-2}$, respectively, and $\tau_t^0, \tau_t^1, ...$ are the indices of the reference slices for the t-th slice, i.e., the slices on which $\hat{z}_t$ is conditioned, such that $\hat{z}_{\tau_t^0}, \hat{z}_{\tau_t^1}, ...$ are at the higher hierarchical levels and are always coded prior to $\hat{z}_t$. By introducing the in-parameter coding structure, each slice depends on the slices at the previous levels but not on all other slices within the same grid tensor. Different from the common approach in neural codecs, which applies the coding structure across temporal frames, where each frame can be individually decoded into the corresponding video frame, our *in-parameter coding structure* approach utilizes slices in the feature grids, each of which can be shared across multiple video frames due to the use of temporal interpolation [21, 22, 28]. We apply the same process to different resolution grid parameters.

Practical implementation for optimization. Although structures such as *low-delay* or *hierarchical B-frame* styles have been widely adopted for various neural video codecs [30, 36], their common implementations are not compatible with overfitted codecs, especially for long video sequences. In particular, most neural video codecs utilize quantization step sizes and offsets[2], which are conditioned

[2] In related works, the mean of the distributions (e.g. Gaussian) is commonly implemented as the quantization offset in their actual implementations.

on the previously coded frames. This creates a *feature propagation chain* that accumulates over video frames. Directly deploying this approach for overfitting long sequences is inefficient or even impossible; the network activations for all frames must be cached to compute gradients during backpropagation, requiring memory far beyond what current GPUs can provide.

To address this issue, we propose two modifications in NVRC++ when deploying the coding structure: (i) NVRC++ utilizes quantization with step sizes that are not conditioned on previous frames, and eliminates the use of offsets; and (ii) due to the removal of the feature propagation chain, NVRC++ can perform random sampling during the optimization process, which reduces the memory footprint while maintaining training-testing consistency. By utilizing this unconditional and symmetric quantization, NVRC++ not only improves the efficiency of rate estimation but also decouples it from distortion computation. For instance, rate and distortion can be calculated separately to reduce both memory footprint and computational cost, following [23]. Fig. 3 (Right) shows a comparison between the conventional and our proposed optimization pipelines, where we utilize random sampling during rate estimation and skip the entropy models in distortion computation.

During rate estimation, we scale the computed rate using its expectation to ensure consistency when balancing the rate-distortion trade-off. As a result, our largest model, which contains feature grids with over 100M parameters, can still encode a 600-frame 1920×1080 video sequence [39], when running on a consumer GPU with only 24GB of memory.

Feature grid masking. Finally, we observe that when incorporating high-resolution features, overfitted codecs utilizing multi-resolution feature grids can suffer from reduced performance at low bitrates. This is likely due to the multi-resolution grids, where output pixel features are obtained via interpolation. In lower-resolution grids, multiple neighboring pixels share the same underlying features because the output resolution is significantly higher than the grid resolution; this allows the model to effectively exploit spatio-temporal redundancies. However, high-resolution grids can easily fit video details, allowing them to assign features to very small regions or even individual pixels. Consequently, these high-resolution features are shared less effectively across the sequence, preventing the effective exploitation of spatio-temporal redundancy at lower bitrates.

To address this issue, we propose a random masking mechanism, following the concept of Dropout [16], for feature grids with higher resolutions at the beginning of training. This forces the model to utilize lower-level grids during the early stages of optimization. The dropout is applied for both rate and distortion calculations, ensuring that the balance between the two terms remains intact. The dropout ratio is annealed throughout the training process, eventually reaching zero in the later stages. We apply different schedules to different sets of grid features depending on their resolution (details are provided in *Supplementary*). We found that this simple approach effectively balances performance across different bitrates, as shown in the experimental results.

3.3 Multi-dimensional grid entropy model

Although higher dimensional grids allow INR models to achieve improved reconstruction quality without increasing complexity, they are costly in storage, limiting their practical applications in video coding. Recent work utilizing an autoregressive model [42] for entropy modeling [18, 21, 23], while demonstrating promising performance, is impractical due to its slow, sequential coding process. Here, we introduce an improved grid entropy model, inspired by the advances in autoencoder-based image and video compression [2, 15, 30–32], where different priors are used to enhance entropy modeling while providing high coding speed.

As shown in Fig. 2, our grid entropy model leverages (i) scale priors, where larger scale feature grids are conditioned by grids with a lower dimension in a hierarchical manner [2]; (ii) temporal priors, which we code grid slices sequentially and utilize the coded slices as conditions [30], as mentioned in Sec. 3.2; and (iii) spatial priors, following common approaches in neural coding to utilize spatial context to improve compression [15, 31, 32].

For a set of multi-scale feature grid parameters $\{z^l \mid 0 \leq l < L_{grid}\}$, where L_{grid} is the number of grid levels, we first encode the grids at a lower resolution, as they will be utilized as the condition for coding higher resolution grids. For coding a particular quantized feature slice $\hat{z}_t^l$, we gather the reference slices in the same scale, e.g., $\hat{z}_{\tau_t^0}^l$, as the temporal prior, and the coarser level slice, $\hat{z}_t^{l-1}$, as the condition. Here, we obtain $\hat{z}_t^{l-1}$ by linear interpolation when $\hat{z}^{l-1}$ has a lower resolution than $\hat{z}^l$. Finally, we perform the coding of $\hat{z}_t^l$ in a multi-step manner, following the spatial prior used for image and video coding, where we mask the grid slice $\hat{z}_t^l$ by $m_s(\hat{z}_t^l)$ at the s-th step and utilize the previously coded features in the same slice, $m_0(\hat{z}_t^l)$, ..., $m_{s-1}(\hat{z}_t^l)$, as the condition, where $m_s(\cdot)$ is the masking pattern at the s-th step following [15, 31, 32].

While using a larger number of coding steps can improve performance, we utilize 1-4 coding steps, depending on the resolution, to reduce the amount of computation and memory footprint. The entropy model first employs the quantization level with a specific step size δ^l, which is shared between all temporal and spatial positions to perform quantization: $\hat{z}_t^l = \lfloor \frac{z_t^l}{\delta^l} \rceil \cdot \delta^l$. The distribution of entropy parameters, including the mean $\mu^{l,t,s}$ and scale $\sigma^{l,t,s}$, will then be estimated by:

$$p(m_s(\hat{z}_t^l)|\hat{z}_t^{l-1}, \hat{z}_{\tau_t^0}^l, \dots, \dots, m_0(\hat{z}_t^l), \dots). \tag{4}$$

This implementation details can be found in *Supplementary*.

3.4 Implicit Neural Representation

In NVRC++, we developed an efficient neural representation for video coding, named HiNeRV++ (shown in Fig. 2, with a full-scale figure provided in *Supplementary*). HiNeRV++ is an enhanced variant of HiNeRV [22], but with the following modifications.

Higher resolution grid features. Since one of our focuses is to improve coding scalability with respect to the rate and quality levels, we utilize multiple feature

grids for feature learning, where the grid resolution is much higher than that of HiNeRV. In HiNeRV, there are four stages of network blocks, where the blocks in each stage perform the computation in the same resolution. Here we utilize encodings extracted from different sets of feature grids [22] as the input to the first three stages of the network blocks, while HiNeRV only does this in the first stage. We also continue using multi-resolution grids in the first input stage.

Multi-reference inputs. In the HiNeRV variant used in NVRC [23], 3D convolution is employed during the encoding process to extract the input encoding from the feature grids. To provide the INR with stronger input conditions, we propose utilizing multiple feature slices as the input for HiNeRV++, as shown in Fig. 3 (Left). Specifically, we gather both the interpolated features and multiple neighboring feature grid slices in the original grid to serve as the input. This provides the model with higher-quality input encodings, as the raw, nearby slices are not subject to interpolation. Consequently, the model can learn to directly leverage these high-fidelity encodings rather than solely relying on interpolated features, as in the original HiNeRV [22]. We apply this modification to all encoding layers.

Removal of the highest level blocks. The blocks in the last stage, together with the head layer in HiNeRV++, are replaced by a single pixel shuffle layer [46]. Although pixel shuffle layers have high parameter complexity with the number of channels and are not efficient for the INR-based compression task [22], they are efficient when used as the output layer, since there are only a few channels in the pixel domain. Furthermore, the computation and memory costs in the highest resolution are substantial, which could slow down inference if stronger blocks, such as ConvNeXt blocks [49], are placed at that resolution.

4 Experimental Settings

Test datasets. The evaluation of the NVRC++ framework is based on commonly used datasets: UVG [39] and MCL-JCV [51]. Both of them contain full HD (1920×1080) sequences, where the frame counts for each sequence range from 300 to 600 for UVG and from 120 to 150 for MCL-JCV. These datasets comprise 7 and 30 sequences, respectively.

Implementation details. NVRC++ is implemented based on HiNeRV [22] and NVRC [23], with the same training strategy. Unlike HiNeRV and NVRC, we configured NVRC++ with four different complexity scales (S1 (smallest), S2, S3 and S4 (largest)), by varying the decoder and local grid channels in HiNeRV++. Each scale employs a single model configuration that achieves a wide quality range through the use of high-resolution feature grids and the advanced entropy model. This allows us to evaluate the trade-off between decoding complexity and compression performance across different computational budgets. More details on implementation are provided in *Supplementary*.

Benchmark methods. For benchmarking NVRC++, three conventional codecs, including x265 (*veryslow*) [52], HM-18.0 (*randomaccess*) [44], and VTM-20.0 (*ran-*

Table 1: BD-rate results of the proposed NVRC++ model (configured at four scales S1-S4) against conventional and neural video codecs on UVG and MCL-JCV datasets. Coding complexity is reported in terms of kMACs per pixel and Frames Per Second (FPS). HiNeRV and NVRC complexities are presented as ranges across various quality scales. FPS are obtained using the same platform with the NVIDIA RTX 4090 GPU with FP16 mixed precision. For NVRC/NVRC++/C3, we report the synthesis transform and the entropy coding time separately following [23]. *: with custom CUDA kernel. **: Entropy coding is not implemented in the official code.

BD-rate (%)	UVG		MCL-JCV		Enc. Complexity	Dec. Complexity	
Method	PSNR	MS-SSIM	PSNR	MS-SSIM	FPS	kMACs/Pixel	FPS
x265 (*veryslow*) [52]	0.00	0.00	0.00	0.00	-	-	-
HM (*RA*) [44]	-44.54	-43.85	-39.91	-41.61	-	-	-
VTM (*RA*) [5]	-62.81	-61.74	-58.86	-61.66	-	-	-
DCVC-FM [33]	-66.30	-67.16	-58.33	-59.45	5.9	369.6 (I)/865.5 (P)	3.7
DCVC-RT [18]	-65.56	-68.05	-57.40	-61.69	113.0	364.9 (I)/166.8 (P)	57.8/141.9*
HiNeRV [22]	-45.84	-63.04	-28.75	-44.79	0.014-0.026	87.3-346.3	13.3-49.9
NVRC [23]	-73.74	-80.65	-51.61	-66.83	0.006-0.016 (13.8-26.2)	173.4-930.6	14.0-33.2 (5.6-19.5)
C3 [21]	-21.91	-14.06	-19.42	-38.35	0.0004	4.4	434.0 (N/A**)
NVRC++ (S1)	-40.06	-67.59	-22.07	-58.02	0.020 (74.9)	7.3	365.0 (62.0)
NVRC++ (S2)	-61.41	-76.78	-47.61	-67.80	0.015 (75.0)	25.1	162.9 (62.4)
NVRC++ (S3)	-71.20	-81.25	-55.90	-70.80	0.010 (75.9)	92.5	73.4 (62.8)
NVRC++ (S4)	-74.82	-81.88	-51.84	-67.30	0.005 (70.5)	357.7	34.0 (59.23)

domaccess), are employed. Two autoencoder-based neural video codecs, DCVC-FM [33] and DCVC-RT [18], are included, both using the single intra-frame setting for the best performance. Additionally, three state-of-the-art INR-based codecs, HiNeRV [22], NVRC [23] and C3 [21] have also been included in this experiment. All results are obtained from the original paper or produced by their open-source implementations and the checkpoints, if provided.

Evaluation metrics. The evaluation was performed in the RGB color space with the BT.601 color conversion. PSNR and MS-SSIM are used here to assess video quality, based on which Bjøntegaard Delta rate (BD-rate) [3] figures are calculated for NVRC++ and each benchmark codec against x265 (*veryslow*). Full sequences are used in the evaluation.

Additional results. We provide additional experimental results, including evaluations on the JVET-CTC Class B dataset and in the YUV420 color space, in the *Supplementary*.

5 Results and Discussion

5.1 Overall Performance

The overall performance of the proposed NVRC++ framework is summarized in Tab. 1. The corresponding rate-quality curves are shown in Fig. 4 (Left).

Compression performance across scales. NVRC++ demonstrates highly scalable compression across four variants (S1-S4), progressively matching or exceeding the baselines of x265, HM, VTM, and the SOTA NVRC. It excels particularly on long sequences (e.g., UVG dataset). At the highest capacity,

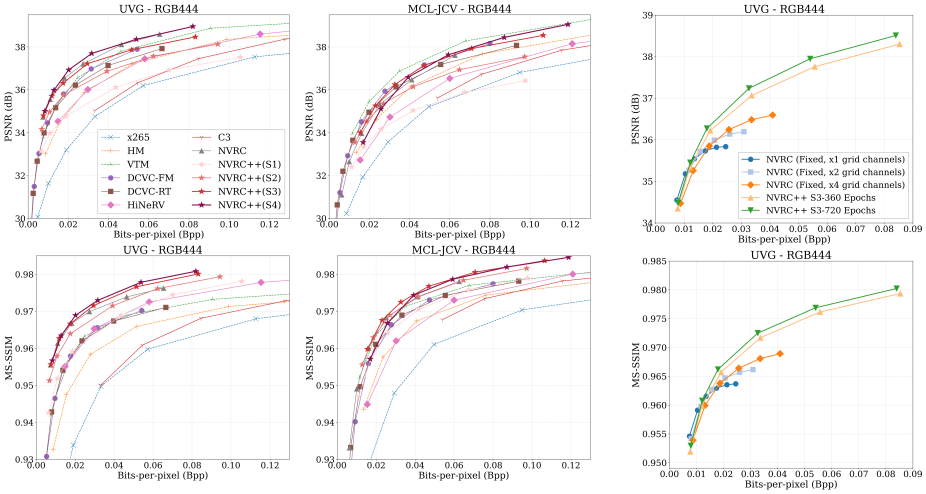

Fig. 4: (Left) RD curves of benchmarked codecs. (Right) Comparison of NVRC++ (92.5kMACs/px) with NVRC (with fixed complexity (102.7kMACs/px)) for scalability.

S4/S3 outperform NVRC's performance in UVG/MCL-JCV while decoding up to 3.5x/7.6x faster. In the mid-complexity range, S3 outperforms VTM and DCVC-RT, and the lightweight S2 beats HM and rivals DCVC-RT at low bitrates, achieving DCVC-level quality (in MS-SSIM) at a fraction of their computational cost. Finally, the ultra-lightweight S1 comfortably outperforms x265 and recent INRs like C3 at similar complexities, while rivaling HiNeRV and HM in MS-SSIM.

Decoding speed and complexity. A key advantage of NVRC++ is its constant decoding complexity across bitrates for a given scale. Unlike NVRC [23] and HiNeRV [22], which exhibit bitrate-dependent costs (173–931 and 87–346 kMACs/Pixel), Each NVRC++ variant maintains fixed complexity for different rate/quality points. It is up to 7.6 × faster than NVRC in decoding speed. The overall complexity-performance trade off of NVRC++ and other benchmark codecs is illustrated by Fig. 1.

Qualitative results. We provided a visual comparison between the proposed method and two SOTA neural video codecs, NVRC and DCVC-RT in Fig. 5. It can be observed that, with similar or even lower bitrates, NVRC++ produces reconstructed content with improved visual quality compared to these two benchmarks, which further confirms its coding performance in perceptual quality.

5.2 Ablation study

The effectiveness of primary innovations in NVRC++ has been validated in an ablation study based on the UVG dataset. The results are summarized in Tab. 2. Specifically, we replaced the multiple high resolution feature grids with only the low-resolution grids [22] (v1), removed the random feature grid masking strategy (v2), disabled the spatial prior (v3), temporal prior (v4), and scale prior (v5) in the entropy model, substituted our multi-reference input design with a 2D

Table 2: The ablation study results on the UVG dataset. Here NVRC++ (S2) is used as the anchor.

BD-rate (%)	PSNR	MS-SSIM
(v1) w/ Low resolution grid	44.01	59.00
(v2) w/o Masking	30.33	39.77
(v3) w/o Spatial prior	1.42	0.52
(v4) w/o Temporal prior	5.13	4.00
(v5) w/o Scale prior	6.12	6.60
(v6) w/ 2D stem	35.55	31.64
(v7) w/ 3D stem	7.76	8.44

Table 3: Comparison between HiNeRV (S) and HiNeRV++ (S2). Results are measured on the UVG dataset.

		HiNeRV	**HiNeRV++**
Parameters	Grids/Layers	**0.35M**/2.84M	100.06M/**1.23M**
Complexity	kMAC/px	87.3	**24.8**
	Dec. FPS	35.5	**162.9**
PSNR (dB)	@37 Epochs	33.70	**37.36**
	@75 Epochs	34.53	**38.27**
	@150 Epochs	34.99	**39.03**
	@300 Epochs	35.27	**39.46**

convolutional stem (v6) and a 3D convolutional stem (v7). It can be observed that all these test variants are associated with inferior performance compared to the original NVRC++ - this confirms the contribution of each component tested. A more detailed analysis is provided in *Supplementary*.

5.3 Additional analysis

HiNeRV++ vs HiNeRV. We have also validated the proposed HiNeRV++ architecture by comparing it with the original HiNeRV, with results shown in Tab. 3. It is noted that by offloading the representational burden from neural layers to high-resolution feature grids, HiNeRV++ drastically increases grid parameters while reducing layer complexity, leading to a lower computational cost and a substantial increase in decoding speed. Furthermore, HiNeRV++ exhibits faster convergence and superior final quality, outperforming HiNeRV by 2.09dB in PSNR at 300 epochs with just 37 epochs of training, proving that lightweight networks with high-resolution grids are a superior solution for INR-based video compression.

Scalability of NVRC++. We also performed a comparison between NVRC and NVRC++ regarding their scalability with respect to quality/rate. For NVRC, we re-implemented the model with a fixed complexity using varying numbers of feature grid channels. For NVRC++, we utilized the S3 variant. Our results in Fig. 4 (Right) show that NVRC achieves a range of quality levels by varying the number of grid channels, at the cost of reduced compression performance at lower bitrates. In contrast, NVRC++ achieves a significantly wider quality range while maintaining high compression performance across all bitrates. This improvement is attributed to both the use of higher-resolution grids and the proposed optimization mechanisms.

6 Conclusion

In this paper, we have proposed an enhanced neural video representation compression framework, NVRC++, aiming to address the low scalability issue associated with existing INR-based video codecs. Through integrating new innovations,

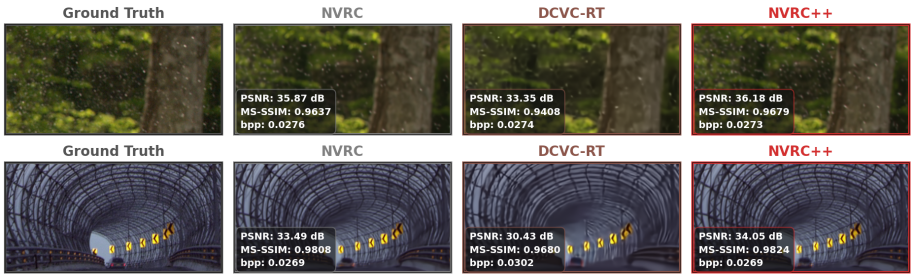

Fig. 5: Qualitative comparison between GT, NVRC, DCVC-RT and NVRC++ (ours).

including multiple high-resolution feature grids, a multi-dimensional grid entropy model, and an in-parameter coding structure, NVRC++ has significantly improved its scalability, enabling the use of a single model configuration for a wide range of bitrates and quality levels. More importantly, it is associated with a higher encoding efficiency (up to 7.6 times faster than NVRC) and achieves real-time decoding (73.4 fps), while offering similar coding performance to NVRC. This work represents an important step forward towards practical applications of INR-based video coding methods.

Acknowledgements

This work was supported by UK EPSRC (iCASE Awards), BT, the UKRI MyWorld Strength in Places Programme and the University of Bristol. Part of the computational work was also supported by the facilities provided by the Advanced Computing Research Centre at the University of Bristol.

References

1. Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end optimized image compression. In: ICLR. OpenReview.net (2017)
2. Ballé, J., Minnen, D., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior. In: ICLR. OpenReview.net (2018)
3. Bjontegaard, G.: Calculation of average psnr differences between rd-curves. ITU SG16 Doc. VCEG-M33 (2001)
4. Bross, B., Wang, Y.K., Ye, Y., Liu, S., Chen, J., Sullivan, G.J., Ohm, J.R.: Overview of the versatile video coding (VVC) standard and its applications. IEEE Transactions on Circuits and Systems for Video Technology **31**(10), 3736–3764 (2021)
5. Browne, A., Ye, Y., Kim, S.H.: Algorithm description for Versatile Video Coding and Test Model 19 (VTM 19). In: the JVET meeting. ITU-T and ISO/IEC (2023)
6. Bull, D., Zhang, F.: Intelligent image and video compression: communicating pictures. Academic Press (2021)
7. Chen, H., Gwilliam, M., Lim, S.N., Shrivastava, A.: HNeRV: A hybrid neural representation for videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10270–10279 (2023)

8. Chen, H., He, B., Wang, H., Ren, Y., Lim, S.N., Shrivastava, A.: NeRV: Neural representations for videos. Advances in Neural Information Processing Systems **34**, 21557–21568 (2021)
9. Chen, Y., Xie, H., Chen, C., Gao, Z., Benjak, M., Peng, W., Ostermann, J.: Maskcrt: Masked conditional residual transformer for learned video compression. IEEE Trans. Circuits Syst. Video Technol. **34**(11), 11980–11992 (2024)
10. Chen, Z., Relic, L., Azevedo, R., Zhang, Y., Gross, M., Xu, D., Zhou, L., Schroers, C.: Neural video compression with spatio-temporal cross-covariance transformers. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 8543–8551 (2023)
11. Gao, G., Kwan, H.M., Zhang, F., Bull, D.: Pnvc: Towards practical inr-based video compression. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 3068–3076 (2025)
12. Gomes, C., Azevedo, R., Schroers, C.: Video compression with entropy-constrained neural representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18497–18506 (2023)
13. Habibian, A., van Rozendaal, T., Tomczak, J.M., Cohen, T.: Video compression with rate-distortion autoencoders. In: ICCV. pp. 7032–7041. IEEE (2019)
14. He, B., Yang, X., Wang, H., Wu, Z., Chen, H., Huang, S., Ren, Y., Lim, S.N., Shrivastava, A.: Towards scalable neural representation for diverse videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6132–6142 (2023)
15. He, D., Zheng, Y., Sun, B., Wang, Y., Qin, H.: Checkerboard Context Model for Efficient Learned Image Compression. In: CVPR. pp. 14771–14780 (2021)
16. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
17. Ho, Y.H., Chang, C.P., Chen, P.Y., Gnutti, A., Peng, W.H.: CANF-VC: Conditional augmented normalizing flows for video compression. In: European Conference on Computer Vision. pp. 207–223. Springer (2022)
18. Jia, Z., Li, B., Li, J., Xie, W., Qi, L., Li, H., Lu, Y.: Towards practical real-time neural video compression. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 12543–12552 (2025)
19. Jiang, W., Li, J., Zhang, K., Zhang, L.: Biecvc: Gated diversification of bidirectional contexts for learned video compression. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 7248–7257 (2025)
20. Jiang, W., Li, J., Zhang, K., Zhang, L.: Ecvc: Exploiting non-local correlations in multiple frames for contextual video compression. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 7331–7341 (2025)
21. Kim, H., Bauer, M., Theis, L., Schwarz, J.R., Dupont, E.: C3: High-performance and low-complexity neural compression from a single image or video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
22. Kwan, H.M., Gao, G., Zhang, F., Gower, A., Bull, D.: HiNeRV: Video compression with hierarchical encoding-based neural representation. Advances in Neural Information Processing Systems **36** (2024)
23. Kwan, H.M., Gao, G., Zhang, F., Gower, A., Bull, D.: Nvrc: Neural video representation compression. arXiv preprint arXiv:2409.07414 (2024)
24. Kwan, H.M., Peng, T., Gao, G., Zhang, F., Nilsson, M., Gower, A., Bull, D.: Ultra-lightweight neural video representation compression. CoRR **abs/2512.04019** (2025)

25. Kwan, H.M., Zhang, F., Gower, A., Bull, D.: Immersive video compression using implicit neural representations. In: PCS. pp. 1–5. IEEE (2024)
26. Ladune, T., Philippe, P., Hamidouche, W., Zhang, L., Déforges, O.: Conditional coding for flexible learned video compression. arXiv preprint arXiv:2104.07930 (2021)
27. Ladune, T., Philippe, P., Henry, F., Clare, G., Leguay, T.: COOL-CHIC: Coordinate-based low complexity hierarchical image codec. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13515–13522 (2023)
28. Lee, J.C., Rho, D., Ko, J.H., Park, E.: Ffnerv: Flow-guided frame-wise neural representations for videos. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 7859–7870 (2023)
29. Leguay, T., Ladune, T., Philippe, P., Déforges, O.: COOL-CHIC video: Learned video coding with 800 parameters. In: 2024 Data Compression Conference (DCC). pp. 23–32. IEEE (2024)
30. Li, J., Li, B., Lu, Y.: Deep contextual video compression. In: NeurIPS. pp. 18114–18125 (2021)
31. Li, J., Li, B., Lu, Y.: Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression. In: ACM Multimedia. pp. 1503–1511. ACM (2022)
32. Li, J., Li, B., Lu, Y.: Neural video compression with diverse contexts. In: CVPR. pp. 22616–22626. IEEE (2023)
33. Li, J., Li, B., Lu, Y.: Neural video compression with feature modulation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 26099–26108 (2024)
34. Li, Z., Wang, M., Pi, H., Xu, K., Mei, J., Liu, Y.: E-nerv: Expedite neural video representation with disentangled spatial-temporal context. In: European Conference on Computer Vision. pp. 267–284. Springer (2022)
35. Liu, J., Wang, S., Ma, W.C., Shah, M., Hu, R., Dhawan, P., Urtasun, R.: Conditional entropy coding for efficient video compression. In: European Conference on Computer Vision. pp. 453–468. Springer (2020)
36. Lu, G., Ouyang, W., Xu, D., Zhang, X., Cai, C., Gao, Z.: DVC: An end-to-end deep video compression framework. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11006–11015 (2019)
37. Maiya, S.R., Girish, S., Ehrlich, M., Wang, H., Lee, K.S., Poirson, P., Wu, P., Wang, C., Shrivastava, A.: Nirvana: Neural implicit representations of videos with adaptive networks and autoregressive patch-wise modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14378–14387 (2023)
38. Mentzer, F., Toderici, G., Minnen, D., Hwang, S.J., Caelles, S., Lucic, M., Agustsson, E.: VCT: A video compression transformer. arXiv preprint arXiv:2206.07307 (2022)
39. Mercat, A., Viitanen, M., Vanne, J.: UVG Dataset: 50/120fps 4K Sequences for Video Codec Analysis and Development. In: MMSys. pp. 297–302. ACM (2020)
40. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4460–4470 (2019)
41. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
42. Minnen, D., Ballé, J., Toderici, G.: Joint autoregressive and hierarchical priors for learned image compression. In: NeurIPS. pp. 10794–10803 (2018)

43. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 165–174 (2019)
44. Rosewarne, C., Sharman, K., Sjoberg, R., Sullivan, G.: High efficiency video coding (HEVC) test model 16 (HM 16) improved encoder description update 16. In: the JVET meeting. ITU-T and ISO/IEC (2022)
45. Sheng, X., Li, L., Liu, D., Wang, S.: Bi-directional deep contextual video compression. IEEE Transactions on Multimedia (2025)
46. Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: CVPR. pp. 1874–1883. IEEE Computer Society (2016)
47. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Advances in neural information processing systems **33**, 7462–7473 (2020)
48. Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the high efficiency video coding (HEVC) standard. IEEE Transactions on circuits and systems for video technology **22**(12), 1649–1668 (2012)
49. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
50. Teng, S., Jiang, Y., Gao, G., Zhang, F., Davis, T., Liu, Z., Bull, D.: Benchmarking conventional and learned video codecs with a low-delay configuration. In: VCIP. pp. 1–5. IEEE (2024)
51. Wang, H., Gan, W., Hu, S., Lin, J.Y., Jin, L., Song, L., Wang, P., Katsavounidis, I., Aaron, A., Kuo, C.J.: MCL-JCV: A JND-based H.264/AVC video quality assessment dataset. In: ICIP. pp. 1509–1513. IEEE (2016)
52. x265: `https://www.videolan.org/developers/x265.html`
53. Xiang, J., Tian, K., Zhang, J.: MIMT: Masked image modeling transformer for video compression. In: The Eleventh International Conference on Learning Representations (2022)
54. Zhang, X., Yang, R., He, D., Ge, X., Xu, T., Wang, Y., Qin, H., Zhang, J.: Boosting neural representations for videos with a conditional decoder. arXiv preprint arXiv:2402.18152 (2024)
55. Zhang, Y., Van Rozendaal, T., Brehmer, J., Nagel, M., Cohen, T.: Implicit neural video compression. arXiv preprint arXiv:2112.11312 (2021)
56. Zhu, J., Zhang, X., Tang, L., Jiang, J.: Msnerv: neural video representation with multi-scale feature fusion. arXiv preprint arXiv:2506.15276 (2025)