# Ultra-Fast Neural Video Compression

Jiahao Li[1], Wenxuan Xie[1], Zhaoyang Jia[2]*, Bin Li[1], Zongyu Guo[1], Xiaoyi Zhang[1], Yan Lu[1]
[1]Microsoft Research Asia, [2] University of Science and Technology of China
{li.jiahao,wenxie,libin,zongyuguo,xiaoyizhang,yanlu}@microsoft.com, jzy_ustc@mail.ustc.edu.cn

## Abstract

*While neural video codecs (NVCs) have demonstrated superior compression ratio, their prohibitive computational complexity remains a critical barrier to real-world deployment. This paper introduces a chunk-based coding framework designed to significantly improve the rate-distortion-complexity trade-off. Instead of processing frames sequentially, our approach encodes a chunk of multiple frames into a single compact latent representation and decodes them simultaneously. This is enabled by cross-frame interaction modules for joint spatial-temporal modeling and frame-specific decoders for parallel reconstruction. This paradigm not only dramatically enhances coding throughput but also facilitates more effective modeling of long-term temporal correlations. To further boost speed, we propose a streamlined entropy coding mechanism that consolidates bit-stream interactions into a single step, substantially reducing decoding overhead. Building on these innovations, we present DCVC-UF (Ultra-Fast), a new NVC that sets a new SOTA in performance. Our experiments show that DCVC-UF can achieve ultra-fast encoding and decoding speeds, significantly outperforming previous leading codecs. DCVC-UF serves as a notable landmark in the journey of NVC evolution. The code is at* https://github.com/microsoft/DCVC.

## 1. Introduction

Neural video codecs (NVCs) have emerged as transformative technologies, offering unprecedented capabilities in removing video redundancy. Recent works [11, 16, 22, 28, 35, 43, 50, 51, 64] have driven rapid progress in improving compression ratios, enabling NVCs to surpass conventional codecs such as H.266/VTM [2]. Despite these advances, the practical deployment of NVCs still faces significant challenges due to their substantial complexity in encoding or decoding. Consequently, achieving a better trade-off among rate, distortion, and complexity is a critical research direction.

In response, recent approaches explore implicit neural representation (INR) [56] or Gaussian Splatting [24], where each video is overfitted into implicit parameters [6, 25] or explicit 2D Gaussians [12, 18]. Both paradigms offer low decoding complexity. However, as they need extensive online optimization for each video, their encoding complexity is quite high. For instance, [18] reports that the encoding times for INRs are usually in the order of $10^{-3}$ FPS.

Another direction to improve the compression ratio within limited computational budgets is to explore more efficient spatial-temporal correlation modeling. For traditional codec H.266/VTM, the hierarchical-B coding can achieve an average of 33.8% bitrate saving over the low-delay coding by introducing the bidirectional temporal prediction in a GOP (group of pictures). So, recent NVCs [10, 23, 55, 66] also follow a similar design to traditional hierarchical-B coding. However, they still operate on a frame-by-frame basis, where each frame relies on explicit motion vector for temporal prediction. The motion vector only captures the pixel

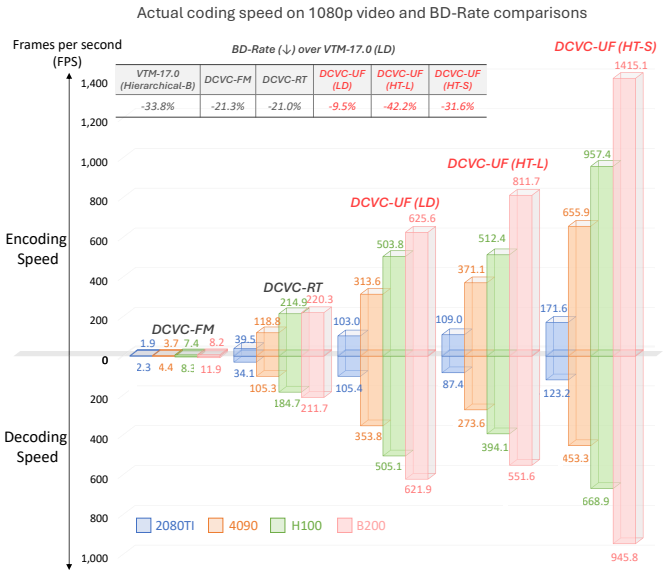


Figure 1. Encoding and decoding speed with actual bit-stream writing and reading on 1920 × 1080 videos across different GPUs. Our DCVC-UF models achieve unprecedented encoding and decoding speeds, demonstrating strong scalability and advanced rate-distortion-complexity trade-off on general-purpose GPUs.

*Done during Zhaoyang Jia's internship at Microsoft Research Asia.

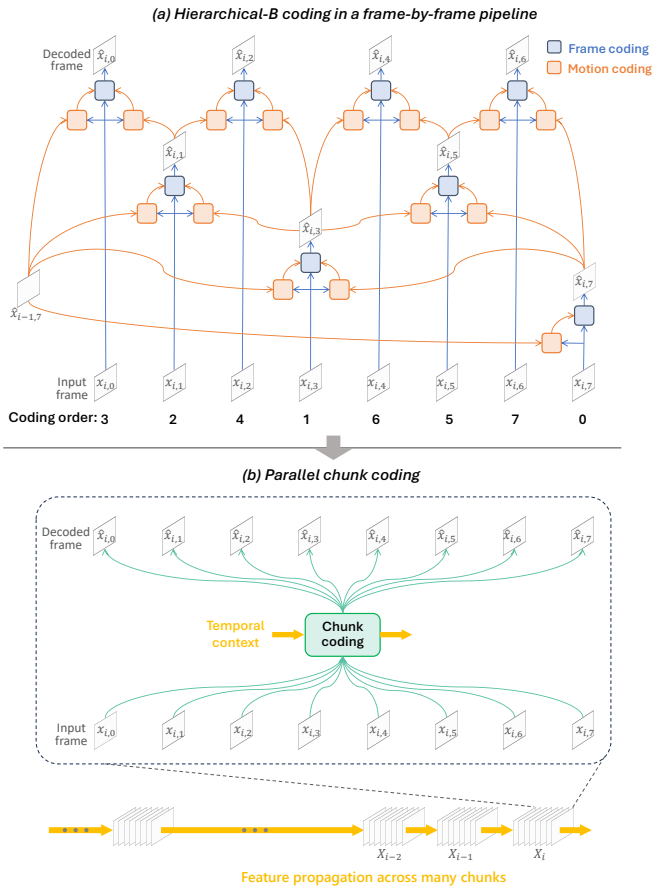


Figure 2. Comparison of coding paradigms, illustrated with an 8-frame example. (a) The commonly-used hierarchical-B coding operates frame-by-frame, particularly relying on the complex motion coding. Here we show two reference frame example for B frames. If using more reference frames, more corresponding motion codings are needed, increasing non-trivial cost. Moreover, the coding order adheres to a rigid, pre-defined hierarchy, and is not learnable. (b) Our chunk coding processes all frames of a chunk in parallel to automatically learn the spatial-temporal correlation. It eliminates the explicit motion, significantly enhances the throughput, and yet enables more efficient long-term temporal modeling.

displacement between only two frames, and cannot represent temporal correlation across multiple frames. Using different reference frames necessitates different motion vectors. In addition, motion vector struggles to handle the complex video dynamics or new contents [17], incurs non-trivial additional bitrate cost, and especially introduces the substantial system complexity. Hence, the capability of these NVCs' to balance rate, distortion, and complexity efficiently remains limited.

In this paper, motivated by the spatial-temporal autoencoder [19] and the motion vector-free DCVC-RT [22], we propose a chunk-based coding framework to address the aforementioned challenges. As shown in Fig. 2, instead of sequentially processing the video frame-by-frame, our approach divides the video into non-overlapping chunks of multiple frames. All frames within each chunk are then encoded into compact latent representations and decoded back simultaneously, which is designed to maximize coding throughput. Within this chunk-based framework, our architecture employs cross-frame interaction modules to jointly and implicitly model spatial-temporal correlations across all frames. Complementing this, a set of frame-specific decoders works in parallel to reconstruct each frame, adaptively tailoring the synthesis process to individual frame characteristics. This paradigm facilitates a more holistic compression strategy and also can maximize coding throughput. It amplifies the advancements of the motion vector-free design from DCVC-RT, removing the costly, iterative, and sequential motion estimation, motion entropy coding, and compensation processes between many frame pairs. Our approach significantly reduces the operational complexity, such as memory I/O and function call overhead, which are critical bottlenecks for practical coding speed.

Our chunk-based coding also enables more efficient modeling of long-term temporal context. One important reason why the previous SOTA DCVC series [22, 34, 35] surpasses leading traditional codecs is the enabling of the feature propagation mechanism in the latent space, which implicitly captures temporal correlations across multiple frames through joint training. Notably, DCVC-FM [35] shows that the compression ratio can be significantly boosted by increasing the training video length from 7 to 32 frames. However, extending the training to longer video sequences is challenging because each frame's separate latent representation incurs substantial training costs. In contrast, our framework encodes all frames in a chunk into a single compact latent, significantly reducing the latent size for a video. It allows for training on much longer video sequences within limited computational budgets, thereby facilitating the exploration of long-term temporal correlations to improve the chunk latent generation and the corresponding distribution estimation.

To further accelerate practical coding speed during the conversion between chunk latents and bit-streams, we introduce a streamlined entropy coding mechanism. Early NVCs [32, 42] typically employ auto-regressive decoding [47], which is inherently slow due to its sequential nature. The recent quadtree partition-based method in [34] reduces decoding steps to four by leveraging decoded partition latents to estimate the means and scales for the next partition in parallel. However, even this four-step interaction with the bit-stream incurs notable operational overhead, especially in real-time scenarios. Our proposed method further simplifies this process by estimating the scales for all partitions in a single step, while retaining the four-step estimation for the means to keep the spatial-channel correlation modeling capability. Since bit-stream decoding depends only on the scales, this allows us to consolidate bit-stream interactions into a single step, substantially reducing operational cost and

improving bit-stream decoding efficiency.

Together, these advancements culminate in our proposed NVC, named DCVC-UF (Ultra-Fast), which builds upon the DCVC series to deliver exceptional encoding and decoding speeds. To accommodate diverse application scenarios, we introduce several configurations of DCVC-UF. Fig. 1 shows the performance comparison, where the VTM (Low-Delay, LD) is as the anchor for BD-Rate calculation. When the chunk has multiple frames, we can achieve High-Throughput (HT) coding yet with high compression ratio. Our large version DCVC-UF (HT-L) saves an average of 42.2% bitrate, with achieving 371.1 encoding and 273.6 decoding FPS for 1080p video with 4090 GPU. Our small version DCVC-UF (HT-S) achieves 31.6% bitrate saving. The encoding and decoding speeds are boosted to 655.9 and 453.3, respectively. These two models will introduce the delay related to the chunk size, analogous to the hierarchical-B coding manner. So, to meet the low-delay requirement, we can also set the chunk size to 1 frame, i.e., DCVC-UF (LD), which can achieve 9.5% bitrate saving, yet achieve 313.6 encoding and 353.8 decoding FPS with 4090 GPU. Unlike traditional codecs requiring bespoke hardware, our NVC framework is built on general-purpose GPUs, allowing it to automatically benefit from rapid advancements in AI accelerators without re-engineering. This inherent scalability is demonstrated as the speeds of our models consistently improve across GPU generations, from consumer cards to datacenter accelerators like the B200, where DCVC-UF (HT-S) sets a new throughput record of 1415.1 encoding and 945.8 decoding FPS for 1080p. As consumer GPUs advance in the future, their speeds will also rise automatically.

Our main contributions are summarized as follows:

- We propose a chunk-based coding framework. It encodes a chunk of frames into a single compact latent and decodes back simultaneously, leveraging cross-frame interaction and frame-specific decoders. This design significantly enhances coding throughput and enables more efficient modeling of long-term temporal context.
- We design a streamlined entropy coding mechanism that decouples the estimation of scales and means for latent partitions. This allows bit-stream interactions to be consolidated into a single step, substantially reducing operational overhead and accelerating practical decoding speed.
- Extensive experiments demonstrate that our model, DCVC-UF, establishes a new record of the rate-distortion-complexity performance, significantly outperforming previous SOTA codecs across various settings.

## 2. Related Work

### 2.1. Low-Delay Neural Video Compression

Low-delay coding constrains the coding of the current frame to reference only previously decoded frames in temporal order, suitable for real-time communication applications. Many methods [4, 21, 36–38, 41, 42, 44, 52] adopt a residual coding inspired by traditional codecs, which requires a complex motion estimation, entropy coding, and compensation pipeline. The emerging conditional coding [20, 29, 32–35, 39, 45, 49, 50, 54] shows larger potential as the temporal context is not limited to pixel-domain prediction but can be any flexible feature. However, most of them still suffer from limited coding speed, as they often incorporate complex modules—particularly those related to motion processing. While some works [30, 57, 58] prioritize acceleration, their compression performance lags behind leading approaches.

### 2.2. Delay-Relaxed Neural Video Compression

Relaxing the frame reference constraint enables a larger design space, but increases delay. It is suited for delay-insensitive scenarios like offline storage and video streaming.

**Hierarchical-B Coding.** Drawing inspiration from the significant compression gains of hierarchical-B coding over low-delay configurations in traditional codecs, several recent NVCs [10, 13, 23, 55, 66] have adopted analogous coding structures. This allows frames to reference both past and future frames for improved prediction. However, these methods still operate on a frame-by-frame basis, relying on explicit motion vectors to align two frames. To mitigate the bitrate overhead of motion vectors, some approaches [10, 55] introduce complex motion vector prediction modules, which in turn increase both computational and operational costs.

**Online Optimization-based Coding.** This paradigm trains a specialized model for each video instance. INR-based methods [6, 7, 15, 25, 27, 28] overfit a small neural network to represent a video, and decode frames by querying the network with coordinates. However, INRs are inefficient for high-resolution video [18]. Consequently, recent works [12, 18, 31, 40, 62] explore explicit representations using Gaussian Splatting [24]. This method associates Gaussian parameters with video regions, enabling scalable representation and faster rendering [18]. While both approaches achieve high decoding speed, their per-video online optimization results in extremely high encoding complexity.

**Spatial-Temporal Autoencoders.** In video generation, spatial-temporal autoencoders serve as powerful tokenizers, compressing raw pixels into a compact latent space to mitigate the prohibitive computational costs of generation in the pixel domain [19]. Recent works [9, 26, 59, 61, 63, 67–69] commonly employ configurations with spatial (e.g., 8x) and temporal (e.g., 4x) compression [8]. While primarily designed for generation, the underlying principle of converting raw video chunk into compact representations makes these autoencoders a promising foundation for developing efficient video codecs. Actually, early works [19, 48] explored NVCs based on spatial-temporal autoencoder. However, [19, 48] use a vanilla autoencoder to mainly learn the the inner corre-

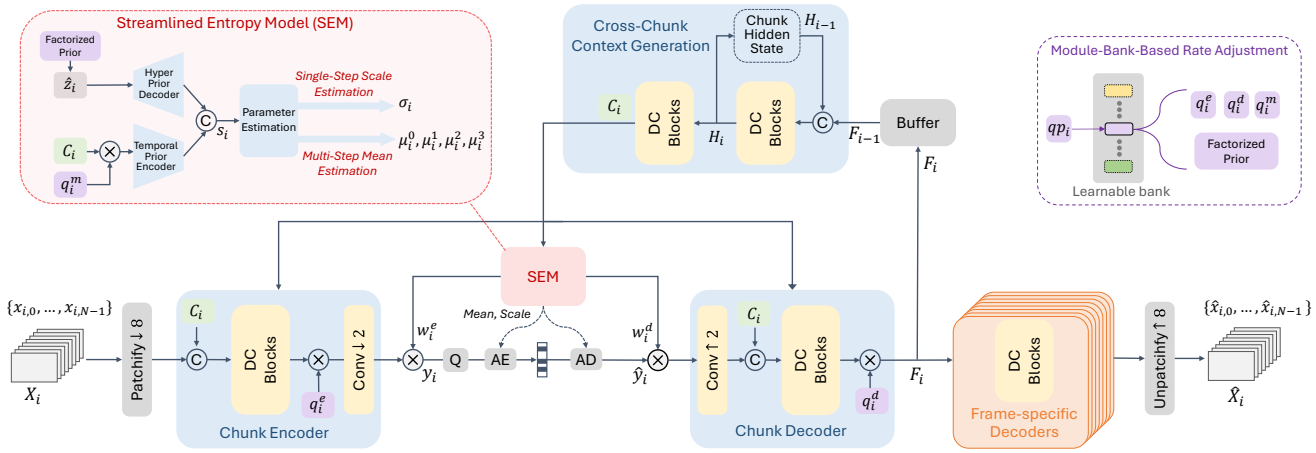


Figure 3. Framework overview of our DCVC-UF. DC Block, Q, AE and AD represent depth-wise convolution block, quantization, arithmetic encoder and decoder, respectively. After the patchify, the input chunk $X_i$ (comprising $N$ frames) directly encoded and decoded into feature $F_i$, conditioned on the temporal chunk context $C_i$. $F_i$ is then reconstructed into pixel domain using the frame-specific decoders. $qp_i$ is the input quantization parameter. The number of DC Blocks for each module is detailed in the supplementary material. In DCVC-UF, all frames in the chunk are processed in parallel to enable the high-throughput coding.

lation within a single chunk. The correlation across chunks is ignored, leading to their limited compression ratio.

Our work advances the spatial-temporal autoencoder paradigm for NVC. Within a chunk, unlike [19, 48], our autoencoder not only has cross-frame interaction modules for joint spatial-temporal modeling but also has frame-specific decoders tailoring the synthesis process to individual frame characteristics. Across different chunks, we build the efficient conditional coding, where temporal propagation is enabled to capture the implicit long-term correlation therein. In addition, we propose a streamlined entropy coding mechanism that consolidates bit-stream interactions into a single step, substantially accelerating the decoding. These make our NVC achieve significant rate-distortion-complexity trade-off advantage over [19, 48] and other previous SOTA codecs.

# 3. Proposed Method

## 3.1. Overview

As depicted in Fig. 3, our DCVC-UF is architected around a chunk-coding paradigm, building upon the DCVC series [22, 34, 35]. The input video is first segmented into non-overlapping chunks. For a given chunk $X_i = \{x_{i,0}, \ldots, x_{i,N-1}\}$ containing $N$ frames, the process begins by transforming it to 1/8 resolution via patch embedding. It is then conditioned on the temporal chunk context $C_i$, and fed into a chunk encoder. The role of the encoder is to distill the spatial-temporal information of the entire chunk into a compact latent representation $y_i$ efficiently. This latent is then quantized ($\hat{y}_i$) and efficiently converted into a bit-stream. During decoding, the process is reversed. The latent representation $\hat{y}_i$ is parsed from the bit-stream and fed to the chunk decoder, which generates a rich feature $F_i$. This feature serves a dual purpose: it is used by a set of parallel, frame-specific decoders to reconstruct the individual frames $\{\hat{x}_{i,0}, \ldots, \hat{x}_{i,N-1}\}$, and it is also propagated to the next chunk to form the next temporal context. DCVC-UF originates from the low-delay NVC DCVC-RT [22] which eliminates the explicit motion-related operations. DCVC-UF amplifies its advantage and enables high-throughput coding via our chunk-coding. DCVC-UF can boost the compression ratio of NVC to a new level with our frame-specific decoders (Sec. 3.2) and efficient long-term correlation learning (Sec. 3.3). In particular, our DCVC-UF also achieves unprecedented coding speed with our streamlined entropy model (Sec. 3.4).

## 3.2. Frame-Specific Decoders

Existing spatial-temporal autoencoders typically employ a unified decoder that applies identical reconstruction processes to all frames within a chunk. While this unified approach is straightforward to implement, it faces limitations when dealing with diverse contents. A single decoder must learn to handle all possible variations across different temporal positions, leading to a challenging optimization problem where the decoder needs to be a "jack of all trades". This often results in suboptimal reconstruction quality, as the decoder cannot fully specialize for the distinct characteristics that may appear at different temporal positions within a chunk. To address these limitations, we additionally design frame-specific decoders in our chunk-based framework, where each frame index within the chunk is assigned its own dedicated decoder. As illustrated in Fig. 3, after the chunk decoder generates the rich feature representation $F_i$ containing spatial-temporal information for all $N$ frames,

we deploy $N$ distinct decoders operating in parallel. Each decoder specializes in reconstructing the frame at its corresponding position.

This design shares some similarities with Mixture of Experts (MoE) [53] architecture, where specialized components handle different aspects of the content. In our case, each frame-specific decoder acts as an "expert" for its temporal position, allowing the model to better adapt to varying video content characteristics. By distributing the reconstruction task across multiple specialized decoders, we can achieve several advantages: (1) Each decoder can focus on learning patterns most relevant to its position, reducing the complexity of individual decoder optimization; (2) The parallel architecture naturally aligns with our chunk-based processing, enabling simultaneous reconstruction without sequential dependencies; (3) The specialization allows for more efficient parameter utilization, as each decoder can allocate its capacity to the specific challenges of its assigned position rather than attempting to generalize across all positions.

### 3.3. Efficient Long-Term Correlation Learning

One of the key advantages of our chunk-based coding framework is its ability to efficiently model long-term temporal correlations. Previous DCVC series have demonstrated that feature propagation mechanisms in the latent space can implicitly capture temporal correlations across multiple frames through joint training, which is a primary factor in their superiority over traditional codecs. Notably, DCVC-FM [35] showed that compression ratio can be significantly improved by extending training video length from 7 to 32 frames, enabling the model to learn more comprehensive temporal dependencies. However, scaling to even longer sequence in frame-based approaches faces fundamental limitations: each frame requires its own latent representation, leading to large training cost. This constraint severely restricts the temporal context that can be practically leveraged during training.

Our chunk-based architecture fundamentally addresses this limitation by encoding all frames within a chunk into a single compact latent representation. This design dramatically reduces the total latent size for a video sequence. This compact representation enables training on much longer sequence (if the batch size is 1, it can be up to 1,024 frames at $512 \times 512$ spatial size, within 24GB GPU memory cost), letting the model capture long-term temporal correlation. The extended temporal context benefits both the chunk latent generation process and the entropy model's distribution estimation. During training, the model learns to identify and exploit recurring patterns, scene structures, and dynamics that span across multiple chunks. The propagated chunk context $C_i$ carries forward essential information, helping subsequent chunks achieve higher compression efficiency.

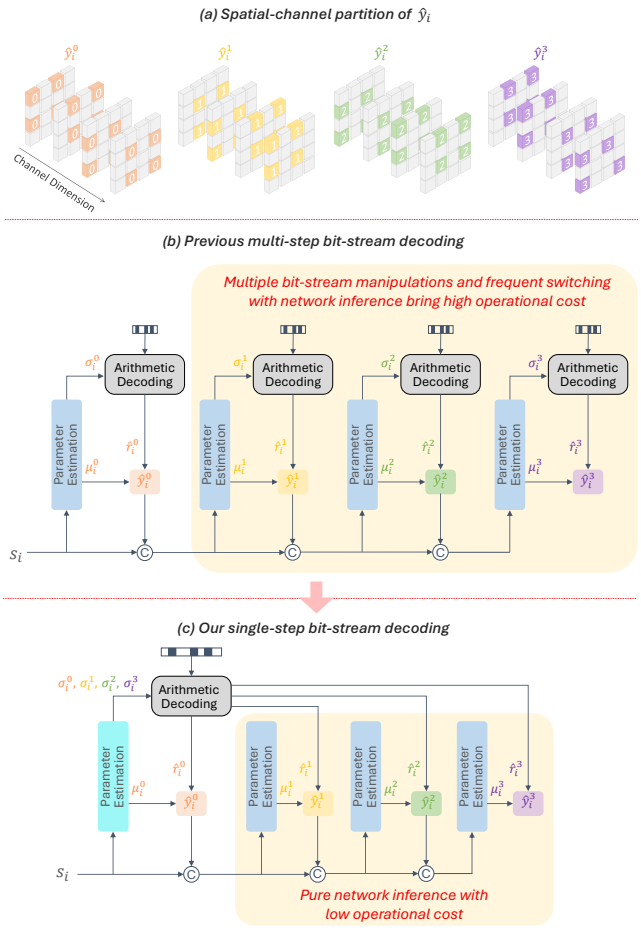


Figure 4. (a) A quadtree-like partition [34] for $\hat{y}_i$ is adopted. (b) Previous methods require interleaved entropy decoding and parameter estimation, which hinders practical decoding speed. (c) Our streamlined entropy model consolidates bit-stream manipulations into a single step, substantially accelerating decoding.

### 3.4. Streamlined Entropy Model

Efficient entropy coding is crucial for achieving high practical coding speed, as it directly determines how quickly latent representations can be converted to and from bit-streams. Recently, the quadtree partition-based entropy coding [34, 35] was proposed to explore spatial-channel correlations efficiently, demonstrating significantly higher decoding speed than the famous auto-regressive model [47]. As illustrated in Fig. 4 (a) and (b), the quantized latent $\hat{y}_i$ is divided into four partitions, where each partition's decoding depends on previously decoded partitions to estimate its distribution parameters (mean $\mu$ and scale $\sigma$). However, even this four-step process still incurs substantial operational overhead. As highlighted in Fig. 4 (b), the repeated bit-stream manipulations involve multiple arithmetic decoding calls, memory I/O operations, and costly synchronization between arithmetic decoding operations and neural network inference. If the

Table 1. BD-Rate (%) comparison in YUV420 colorspace. All frames are tested.

| Method | UVG | MCL-JCV | HEVC B | HEVC C | HEVC D | HEVC E | Average | Coding Speed | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Enc. | Dec. |
| *Low-Delay (LD) Codecs* | | | | | | | | | |
| VTM-17.0 (LD) | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.01 FPS | 23.6 FPS |
| HM-16.25 (LD) | 40.1 | 48.6 | 47.6 | 41.0 | 34.5 | 42.8 | 42.4 | 0.05 FPS | 39.6 FPS |
| DCVC-DC | 6.5 | −4.4 | 13.1 | −3.4 | −14.8 | 90.2 | 14.5 | 2.3 FPS | 2.9 FPS |
| DCVC-FM | −16.8 | −8.0 | −15.4 | −30.2 | −37.5 | −20.2 | −21.3 | 3.7 FPS | 4.4 FPS |
| DCVC-RT | −24.0 | −14.8 | −16.6 | −21.0 | −27.3 | −22.4 | −21.0 | 118.8 FPS | 105.3 FPS |
| **DCVC-UF (LD)** | −15.3 | −0.3 | −3.3 | −6.5 | −16.6 | −15.0 | −9.5 | **313.6** FPS | **353.8** FPS |
| *Delay-Relaxed Codecs* | | | | | | | | | |
| HM-16.25 (Hierarchical-B) | 4.9 | 17.3 | 12.6 | 11.3 | 3.2 | 1.6 | 8.5 | 0.06 FPS | 40.0 FPS |
| VTM-17.0 (Hierarchical-B) | −34.0 | −30.4 | −35.4 | −32.4 | −32.5 | −38.1 | −33.8 | 0.01 FPS | 23.1 FPS |
| **DCVC-UF (HT-S)** | −28.8 | −12.9 | −17.6 | −29.4 | −42.2 | −58.8 | −31.6 | **655.9** FPS | **453.3** FPS |
| **DCVC-UF (HT-L)** | −39.6 | −24.4 | −33.3 | −41.2 | −51.7 | −63.2 | −42.2 | **371.1** FPS | **273.6** FPS |

Intra-period=–1 for all codecs and settings. The coding speeds of NVCs are tested on $1920 \times 1080$ videos with 4090 GPU.

arithmetic decoding is performed on CPU, the cross-device switching and synchronization between CPU and GPU further exacerbate the operational burden.

To address these bottlenecks, we first revisit the entropy coding process. During encoding, the entropy model estimates the mean $\mu_i$ and scale $\sigma_i$ for the latent $y_i$, typically assuming a Gaussian distribution. After quantization via $\hat{r}_i = \text{round}(y_i - \mu_i)$, the result $\hat{r}_i$ is arithmetically encoded using scale $\sigma_i$. During decoding, $\hat{r}_i$ is recovered using only $\sigma_i$, and the final latent is reconstructed as $\hat{y}_i = \hat{r}_i + \mu_i$. The key insight is that bit-stream operations depend exclusively on scale parameters, which define the distribution width, while mean parameters merely shift the distribution center and can be applied post-decoding. This observation motivates us to decouple the estimation of means and scales, departing from the coupled approach in [34, 35]. As shown in Fig. 4 (c), our parameter estimation network takes the prior input $s_i$ (derived from hyper-prior $\hat{z}_i$ and temporal context $C_i$) and simultaneously predicts the mean $\mu_i^0$ for the first partition and the scales $\sigma_i$ for all four partitions in a single forward pass.

This architectural innovation enables a dramatic acceleration of the decoding pipeline. By eliminating sequential dependencies in scale estimation, we can perform arithmetic decoding for all partitions in one consolidated step, drastically reducing bit-stream manipulation overhead. We retain the four-step progressive estimation for means to preserve spatial-channel correlation modeling capacity, but since mean estimation requires no bit-stream interaction, it executes entirely on GPU without costly synchronization. This design minimizes memory transfer between processing units, eliminates multi-step decoding latency, and removes switching overhead between arithmetic operations and neural network inference. Our streamlined entropy model allows for better GPU utilization, combined with our chunk-based coding framework, enables DCVC-UF to achieve unprecedented decoding speed.

# 4. Experimental Results

## 4.1. Experimental Settings

**Implementation Details**. For delay-relaxed scenario, our high-throughput (HT) codec DCVC-UF uses a chunk size of $N = 8$. We provide two network scales—DCVC-UF (HT-S) and DCVC-UF (HT-L)—denoting small and large models, respectively. To enable low-delay (LD) operation, the framework also supports $N = 1$ (single-frame chunk), i.e., DCVC-UF (LD).

**Training Details**. Following [35], we first train DCVC-UF codecs on the ready-made 7-frame Vimeo-90k [65] dataset, then fine-tune using longer sequences generated from original Vimeo videos [3]. As discussed in Section 3.3, our chunk coding enables training on $512 \times 512$ videos with up to 1024

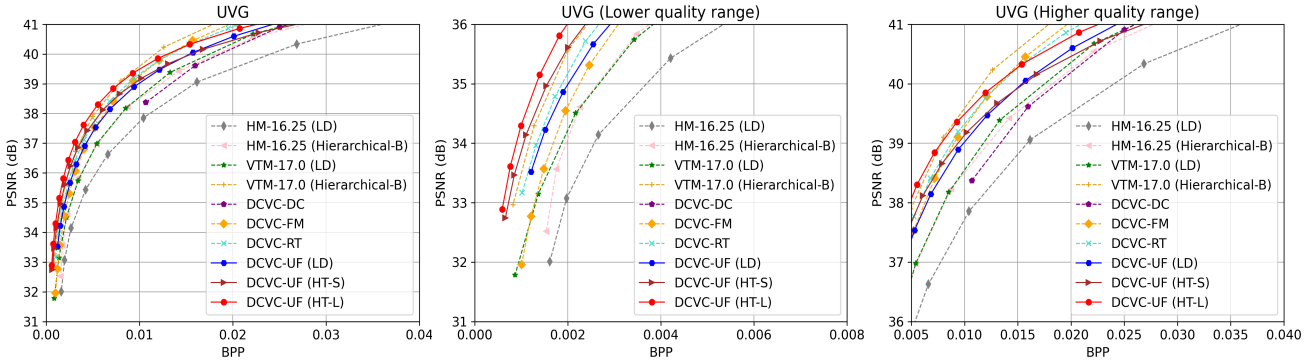


Figure 5. Rate-distortion curves on UVG dataset. BPP means bits per pixel. More curves are in the supplementary material.

frames. However, assembling diverse, high-quality videos of such length is still challenging. Therefore, we currently fine-tune with 128-frame sequences and leave the exploration of longer training datasets for future work.

**Testing Details**. We evaluate on HEVC Class B∼E [14], UVG [46], and MCL-JCV [60]. For traditional codecs, we compare with HM [1] and VTM [2], representing the best H.265 and H.266 encoders, respectively. Configuration details are in the supplementary material. For NVCs, we compare with previous SOTA DCVC series [22, 34, 35], with all models tested using actual bit-stream writing and decoding. Compression ratio is measured by BD-Rate [5], where positive values indicate a bitrate increase and negative values indicate savings. Video quality is reported using PSNR, with all frames evaluated in YUV420 colorspace. For both low-delay and delay-relaxed settings, the intra-period is set to –1 for all codecs to present their best compression ratios. We measure coding speed on a single GPU, using a sequential chunk-by-chunk coding process (chunk size $N = 8$ for HT and $N = 1$ for LD). We currently do not employ cross-chunk pipeline parallelism (e.g., overlapping the network inference and entropy coding of different chunks), indicating potential for further acceleration.

## 4.2. Comparisons with Previous SOTA Methods

Table 1 shows the BD-Rate comparison. In terms of the averaged bitrate saving, our DCVC-UF (HT-L) achieves the best performance, i.e., an average of 42.2% bitrate saving over VTM (LD). By contrast, the corresponding result of the VTM (Hierarchical-B) is 33.8%, where we use the default GOP size 32 and its maximum frame delay is 31 frames. Our chunk size is 8, leading to a maximum 7 frame delay, which is much smaller. If VTM (Hierarchical-B) also uses GOP size 8, the corresponding bitrate saving is reduced to 23.7%. It shows the compression efficiency of our chunk coding.

In terms of the actual encoding and decoding speeds on 4090 GPU, DCVC-UF (HT-L) can achieve 371.1 FPS and 273.6 FPS for 1080p videos, respectively. When using a smaller network structure, our DCVC-UF (HT-L) can boost the encoding and decoding speeds to 655.9 FPS and 453.3 FPS, respectively. But the average bitrate saving can still keep 31.6%, comparable with VTM (Hierarchical-B) using GOP 32. This shows the advanced rate-distortion-complexity trade-off of our chunk coding paradigm. For the low-delay setting, our DCVC-UF (LD) can reach 313.6 encoding and 353.8 decoding speeds. Although the bitrate saving over VTM (LD) is lower than that of DCVC-RT, the decoding speed acceleration is more than 3× times.

Figure 5 illustrates the rate-distortion curves on the UVG dataset. In the lower quality range, both DCVC-UF (HT-L) and DCVC-UF (HT-S) consistently outperform all previous codecs. At a higher quality range, VTM (Hierarchical-B) shows better performance, which is expected given our models' lightweight nature. However, this performance gap primarily manifests above 40 dB, where quality differences become imperceptible to human vision.

Table 2. Ablation study. VTM-17.0 (LD) is as the anchor for BD-Rate results. Decoding FPS is tested on 4090 GPU.

| ID | Settings | BD-Rate | Decoding FPS |
|---|---|---|---|
| A | DCVC-RT → *Anchor* | -21.0% | 105.3 |
| B | A + Chunk coding (w/o frame-specific decoder) | -10.1% | 349.1 |
| C | A + Chunk coding (w/ frame-specific decoder) | -25.3% | 343.2 |
| D | C + Streamlined entropy model | -23.4% | 453.3 |
| E | D + Training with 128-frame video → *Proposed DCVC-UF (HT-S)* | -31.6% | 453.3 |

## 4.3. Ablation Study

Table 2 presents an ablation study. Starting from DCVC-RT as our baseline (A), we first introduce chunk coding without frame-specific decoders (B), which significantly boosts decoding speed from 105.3 to 349.1 FPS but compromises compression efficiency (bitrate saving changes from 21.0% to 10.1%). This drop highlights the challenge of using a single

Table 3. Complexity analysis. The encoding/decoding speeds (frames per second, FPS) are evaluated across various resolutions and devices. Average BD-Rate results are presented using VTM-17.0 (LD) as the anchor. MACs are tested on 1080p videos. OOM means out-of-memory.

| Model | Average BD-Rate | Average MACs/frame | Params |
|---|---|---|---|
| DCVC-FM | −21.3% | 2642G | 18.3M |
| DCVC-RT | −21.0% | 385G | 20.7M |
| DCVC-UF (LD) | −9.5% | 170G | 9.7M |
| DCVC-UF (HT-S) | −31.6% | 211G | 81.2M |
| DCVC-UF (HT-L) | −42.2% | 343G | 120.5M |

(a) Computational complexity and BD-Rate.

| Model | 2080Ti | 4090 | A100 | H100 | B200 |
|---|---|---|---|---|---|
| DCVC-FM | 4.0 / 4.7 | 9.3 / 10.4 | 8.5 / 9.4 | 11.2/16.9 | 12.3/21.7 |
| DCVC-RT | 73.3 / 67.0 | 225.1 / 185.2 | 173.9 / 149.2 | 284.4/252.4 | 289.7/263.9 |
| DCVC-UF (LD) | 191.0/203.7 | 432.5/634.6 | 525/501.2 | 777.2/706.5 | 848.4/902.5 |
| DCVC-UF (HT-S) | 348.1/251.8 | 1194.5/985.2 | 1098.4/786.1 | 1901.5/1347.2 | 2318.2/1633.1 |
| DCVC-UF (HT-L) | 206.4/163.1 | 778.9/558.4 | 648.2/459.4 | 1083.1/752.0 | 1424.9/908.3 |

(b) Coding speed on $1280 \times 720$ videos.

| Model | 2080Ti | 4090 | A100 | H100 | B200 |
|---|---|---|---|---|---|
| DCVC-FM | 1.9/2.3 | 3.7/4.4 | 5.0/5.9 | 7.4/8.3 | 8.2/11.9 |
| DCVC-RT | 39.5/34.1 | 118.8/105.3 | 125.2/112.8 | 214.9/184.7 | 220.3/211.7 |
| DCVC-UF (LD) | 103.0/105.4 | 313.6/353.8 | 317.0/314.5 | 503.8/505.1 | 625.6/621.9 |
| DCVC-UF (HT-S) | 171.6/123.2 | 655.9/453.3 | 576.2/411.1 | 957.4/668.9 | 1415.1/945.8 |
| DCVC-UF (HT-L) | 109.0/87.4 | 371.1/273.6 | 331.0/247.4 | 512.4/394.1 | 811.7/551.6 |

(c) Coding speed on $1920 \times 1080$ videos.

| Model | 2080Ti | 4090 | A100 | H100 | B200 |
|---|---|---|---|---|---|
| DCVC-FM | OOM | OOM | 1.0/1.2 | 1.8/2.2 | 2.5/3.3 |
| DCVC-RT | 11.6/9.9 | 29.9/26.5 | 35.5/29.5 | 56.9/52.0 | 91.6/87.1 |
| DCVC-UF (LD) | 29.2/29.9 | 80.2/81.8 | 91.6/93.6 | 156.8/158.2 | 230.2/226.4 |
| DCVC-UF (HT-S) | 45.9/31.6 | 139.5/94.7 | 155.3/107.3 | 255.9/179.6 | 424.0/289.5 |
| DCVC-UF (HT-L) | 26.1/21.9 | 83/61.9 | 84.3/67.4 | 129.0/99.8 | 237.9/177.2 |

(d) Coding speed on $3840 \times 2160$ videos.

unified decoder for all temporal positions. Adding our frame-specific decoders (C) improves compression performance to 25.3% bitrate saving while maintaining high decoding speed at 343.2 FPS, validating our design of specialized decoders for each temporal position. With the streamlined entropy model (D), decoding reaches 453.3 FPS at 23.4% bitrate saving, evidencing the acceleration benefit of the single-step bit-stream interaction. Finally, extending training to 128-frame sequences (E) substantially improves bitrate saving to 31.6% without affecting decoding speed, confirming that our chunk-based framework enables efficient learning of long-term temporal correlations.

### 4.4. Throughput Scaling on General-Purpose GPUs

Traditional video codecs are tightly coupled to hardware, often requiring multi-year standardization cycles and bespoke silicon or platform-specific hand-tuning. In contrast, neural codecs rely on commodity GPU primitives (e.g., convolutions and matrix multiplications) that naturally benefit from progress in AI accelerators. This compute alignment eliminates most hardware-specific engineering across heterogeneous devices, enabling automatic speedups as either model architectures or GPU hardware improves. Table 3 demonstrates NVC's scalability across GPU generations. On the B200, DCVC-UF (HT-S) reaches 1415.1 encoding and 945.8 decoding FPS for 1080p videos, setting a new NVC speed record in history. These gains show the potential of NVCs for large-scale enterprise workloads (e.g., cloud video analytics, mass transcoding). The consistent improvements from consumer GPUs (e.g., 2080 Ti, 4090) to datacenter accelerators (e.g., H100, B200) validate its ability to harness general-purpose compute advances without GPU-specific re-engineering.

## 5. Conclusion and Limitation

This paper presents DCVC-UF NVC that achieves unprecedented encoding and decoding speeds while maintaining high compression efficiency. DCVC-UF adopts a chunk-based coding framework that processes multiple frames simultaneously. It not only dramatically enhances coding throughput but also facilitates more effective modeling of long-term temporal correlations. By introducing frame-specific decoders that act as specialized experts for each temporal position, our approach better adapts to diverse video content characteristics. Furthermore, our streamlined entropy model consolidates bit-stream interactions into a single step by decoupling scale and mean estimation, markedly reducing operational overheads. DCVC-UF represents a significant step in the evolution of NVC, fundamentally transforming the commonly-used yet complex hierarchical-B coding to a much more efficient chunk coding.

Despite these advances, the current DCVC-UF uses a fixed chunk size, which may not be optimal for videos with varying temporal characteristics. Future work could explore adaptive chunk sizing based on content complexity.

## References


[1] HM. https://vcgit.hhi.fraunhofer.de/jvet/HM. 7

[2] VTM. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM. 1, 7

[3] Original Vimeo links. https://github.com/anchen1011/toflow/blob/master/data/original_vimeo_links.txt. 6

[4] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020. 3

[5] Gisle Bjontegaard. Calculation of average PSNR differences between RD-curves. *VCEG-M33*, 2001. 7

[6] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021. 1, 3

[7] Hao Chen, Matthew Gwilliam, Ser-Nam Lim, and Abhinav Shrivastava. Hnerv: A hybrid neural representation for videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10270–10279, 2023. 3

[8] Junyu Chen, Wenkun He, Yuchao Gu, Yuyang Zhao, Jincheng Yu, Junsong Chen, Dongyun Zou, Yujun Lin, Zhekai Zhang, Muyang Li, et al. Dc-videogen: Efficient video generation with deep compression video autoencoder. *arXiv preprint arXiv:2509.25182*, 2025. 3

[9] Liuhan Chen, Zongjian Li, Bin Lin, Bin Zhu, Qian Wang, Shenghai Yuan, Xing Zhou, Xinhua Cheng, and Li Yuan. Odvae: An omni-dimensional video compressor for improving latent video diffusion model. *arXiv preprint arXiv:2409.01199*, 2024. 3

[10] Mu-Jung Chen, Yi-Hsin Chen, and Wen-Hsiao Peng. B-canf: Adaptive b-frame coding with conditional augmented normalizing flows. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(4):2908–2921, 2023. 1, 3

[11] Zhenghao Chen, Lucas Relic, Roberto Azevedo, Yang Zhang, Markus Gross, Dong Xu, Luping Zhou, and Christopher Schroers. Neural Video Compression with Spatio-Temporal Cross-Covariance Transformers. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 8543–8551, 2023. 1

[12] Zhenghao Chen, Zicong Chen, Lei Liu, Yiming Wu, and Dong Xu. Versatile video tokenization with generative 2d gaussian splatting. *arXiv preprint arXiv:2508.11183*, 2025. 1, 3

[13] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3

[14] D Flynn, K Sharman, and C Rosewarne. Common Test Conditions and Software Reference Configurations for HEVC Range Extensions, document JCTVC-N1006. *Joint Collaborative Team Video Coding ITU-T SG*, 16. 7

[15] Ge Gao, Ho Man Kwan, Fan Zhang, and David Bull. Pnvc: Towards practical inr-based video compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3068–3076, 2025. 3

[16] Ge Gao, Siyue Teng, Tianhao Peng, Fan Zhang, and David Bull. Givic: Generative implicit video compression. *arXiv preprint arXiv:2503.19604*, 2025. 1

[17] Yue Gao, Jiahao Li, Lei Chu, and Yan Lu. Implicit motion function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19278–19289, 2024. 2

[18] Lakshya Gupta and Imran N Junejo. Neural video compression using 2d gaussian splatting. *arXiv preprint arXiv:2505.09324*, 2025. 1, 3

[19] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7033–7042, 2019. 2, 3, 4

[20] Yung-Han Ho, Chih-Peng Chang, Peng-Yu Chen, Alessandro Gnutti, and Wen-Hsiao Peng. Canf-vc: Conditional augmented normalizing flows for video compression. *European Conference on Computer Vision*, 2022. 3

[21] Zhihao Hu, Zhenghao Chen, Dong Xu, Guo Lu, Wanli Ouyang, and Shuhang Gu. Improving deep video compression by resolution-adaptive flow coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 193–209. Springer, 2020. 3

[22] Zhaoyang Jia, Bin Li, Jiahao Li, Wenxuan Xie, Linfeng Qi, Houqiang Li, and Yan Lu. Towards practical real-time neural video compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-25, 2024*, 2025. 1, 2, 4, 7

[23] Wei Jiang, Junru Li, Kai Zhang, and Li Zhang. Biecvc: Gated diversification of bidirectional contexts for learned video compression. *arXiv preprint arXiv:2505.09193*, 2025. 1, 3

[24] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 3

[25] Hyunjik Kim, Matthias Bauer, Lucas Theis, Jonathan Richard Schwarz, and Emilien Dupont. 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*, pages 9347–9358, 2024. 1, 3

[26] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. 3

[27] Ho Man Kwan, Ge Gao, Fan Zhang, Andrew Gower, and David Bull. Hinerv: Video compression with hierarchical encoding-based neural representation. *Advances in Neural Information Processing Systems*, 36:72692–72704, 2023. 3

[28] Ho Man Kwan, Ge Gao, Fan Zhang, Andrew Gower, and David Bull. Nvrc: Neural video representation compres-

sion. *Advances in Neural Information Processing Systems*, 37:132440–132462, 2024. 1, 3

[29] Théo Ladune, Pierrick Philippe, Wassim Hamidouche, Lu Zhang, and Olivier Déforges. Conditional Coding for Flexible Learned Video Compression. In *Neural Compression: From Information Theory to Applications – Workshop @ ICLR 2021*, 2021. 3

[30] Hoang Le, Liang Zhang, Amir Said, Guillaume Sautiere, Yang Yang, Pranav Shrestha, Fei Yin, Reza Pourreza, and Auke Wiggers. Mobilecodec: neural inter-frame video compression on mobile devices. In *Proceedings of the 13th ACM Multimedia Systems Conference*, pages 324–330, 2022. 3

[31] Inseo Lee, Youngyoon Choi, and Joonseok Lee. Gaussian-video: Efficient video representation and compression by gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 4471–4480, 2025. 3

[32] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. *Advances in Neural Information Processing Systems*, 34:18114–18125, 2021. 2, 3

[33] Jiahao Li, Bin Li, and Yan Lu. Hybrid spatial-temporal entropy modelling for neural video compression. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1503–1511, 2022.

[34] Jiahao Li, Bin Li, and Yan Lu. Neural video compression with diverse contexts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22616–22626, 2023. 2, 4, 5, 6, 7

[35] Jiahao Li, Bin Li, and Yan Lu. Neural Video Compression with Feature Modulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 17-21, 2024*, 2024. 1, 2, 3, 4, 5, 6, 7

[36] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-LVC: Multiple frames prediction for learned video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3546–3554, 2020. 3

[37] Bowen Liu, Yu Chen, Rakesh Chowdary Machineni, Shiyu Liu, and Hun-Seok Kim. MMVC: Learned Multi-Mode Video Compression with Block-based Prediction Mode Selection and Density-Adaptive Entropy Coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18487–18496, 2023.

[38] Haojie Liu, Ming Lu, Zhan Ma, Fan Wang, Zhihuang Xie, Xun Cao, and Yao Wang. Neural video coding using multi-scale motion compensation and spatiotemporal context model. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(8):3182–3196, 2020. 3

[39] Jerry Liu, Shenlong Wang, Wei-Chiu Ma, Meet Shah, Rui Hu, Pranaab Dhawan, and Raquel Urtasun. Conditional entropy coding for efficient video compression. In *European Conference on Computer Vision*, pages 453–468. Springer, 2020. 3

[40] Xiang Liu, Bin Chen, Zimo Liu, Yaowei Wang, and Shu-Tao Xia. An exploration with entropy constrained 3d gaussians for 2d video compression. In *The Thirteenth International Conference on Learning Representations*, 2023. 3

[41] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: an end-to-end deep video compression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 3

[42] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. An end-to-end learning framework for video compression. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3292–3308, 2020. 2, 3

[43] Wenzhuo Ma and Zhenzhong Chen. Diffusion-based perceptual neural video compression with temporal diffusion information reuse. *arXiv preprint arXiv:2501.13528*, 2025. 1

[44] Wufei Ma, Jiahao Li, Bin Li, and Yan Lu. Uncertainty-Aware Deep Video Compression with Ensembles. *IEEE Transactions on Multimedia*, 2024. 3

[45] Fabian Mentzer, George Toderici, David Minnen, Sung-Jin Hwang, Sergi Caelles, Mario Lucic, and Eirikur Agustsson. Vct: A video compression transformer. *arXiv preprint arXiv:2206.07307*, 2022. 3

[46] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. UVG dataset: 50/120fps 4K sequences for video codec analysis and development. In *Proceedings of the 11th ACM Multimedia Systems Conference*, pages 297–302, 2020. 7

[47] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, 31, 2018. 2, 5

[48] Jorge Pessoa, Helena Aidos, Pedro Tomás, and Mário AT Figueiredo. End-to-end learning of video compression using spatio-temporal autoencoders. In *2020 IEEE Workshop on Signal Processing Systems (SiPS)*, pages 1–6. IEEE, 2020. 3, 4

[49] Linfeng Qi, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Motion information propagation for neural video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6111–6120, 2023. 3

[50] Linfeng Qi, Zhaoyang Jia, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Long-term temporal context gathering for neural video compression. In *European Conference on Computer Vision*, pages 305–322. Springer, 2024. 1, 3

[51] Linfeng Qi, Zhaoyang Jia, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Generative latent coding for ultra-low bitrate image and video compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025. 1

[52] Oren Rippel, Alexander G Anderson, Kedar Tatwawadi, Sanjay Nair, Craig Lytle, and Lubomir Bourdev. ELF-VC: Efficient learned flexible-rate video coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14479–14488, 2021. 3

[53] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021. 5

[54] Xihua Sheng, Jiahao Li, Bin Li, Li Li, Dong Liu, and Yan Lu. Temporal Context Mining for Learned Video Compression. *IEEE Transactions on Multimedia*, 2022. 3

[55] Xihua Sheng, Li Li, Dong Liu, and Shiqi Wang. Bi-directional deep contextual video compression. *IEEE Transactions on Multimedia*, 2025. 1, 3

[56] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020. 1

[57] Kuan Tian, Yonghang Guan, Jinxi Xiang, Jun Zhang, Xiao Han, and Wei Yang. Towards real-time neural video codec for cross-platform application using calibration information. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 7961–7970, 2023. 3

[58] Ties Van Rozendaal, Tushar Singhal, Hoang Le, Guillaume Sautiere, Amir Said, Krishna Buska, Anjuman Raha, Dimitris Kalatzis, Hitarth Mehta, Frank Mayer, et al. Mobilenvc: Real-time 1080p neural video compression on a mobile device. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4323–4333, 2024. 3

[59] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3

[60] Haiqiang Wang, Weihao Gan, Sudeng Hu, Joe Yuchieh Lin, Lina Jin, Longguang Song, Ping Wang, Ioannis Katsavounidis, Anne Aaron, and C-C Jay Kuo. MCL-JCV: a JND-based H. 264/AVC video quality assessment dataset. In *2016 IEEE international conference on image processing (ICIP)*, pages 1509–1513. IEEE, 2016. 7

[61] Junke Wang, Yi Jiang, Zehuan Yuan, Bingyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. *Advances in Neural Information Processing Systems*, 37:28281–28295, 2024. 3

[62] Longan Wang, Yuang Shi, and Wei Tsang Ooi. Gsvc: Efficient video representation and compression through 2d gaussian splatting. In *Proceedings of the 35th Workshop on Network and Operating System Support for Digital Audio and Video*, pages 15–21, 2025. 3

[63] Pingyu Wu, Kai Zhu, Yu Liu, Liming Zhao, Wei Zhai, Yang Cao, and Zheng-Jun Zha. Improved video vae for latent video diffusion model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18124–18133, 2025. 3

[64] Naifu Xue, Zhaoyang Jia, Jiahao Li, Bin Li, Zihan Zheng, Yuan Zhang, and Yan Lu. Single-step diffusion-based video coding with semantic-temporal guidance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2026. 1

[65] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 6

[66] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 1, 3

[67] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 3

[68] Sijie Zhao, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Muyao Niu, Xiaoyu Li, Wenbo Hu, and Ying Shan. Cv-vae: A compatible video vae for latent generative video models. *Advances in Neural Information Processing Systems*, 37: 12847–12871, 2024.

[69] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all. *arXiv preprint arXiv:2412.20404*, 2024. 3