# Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression

Jiahao Li
Microsoft Research Asia
Beijing, China
li.jiahao@microsoft.com

Bin Li
Microsoft Research Asia
Beijing, China
libin@microsoft.com

Yan Lu
Microsoft Research Asia
Beijing, China
yanlu@microsoft.com

## ABSTRACT

For neural video codec, it is critical, yet challenging, to design an efficient entropy model which can accurately predict the probability distribution of the quantized latent representation. However, most existing video codecs directly use the ready-made entropy model from image codec to encode the residual or motion, and do not fully leverage the spatial-temporal characteristics in video. To this end, this paper proposes a powerful entropy model which efficiently captures both spatial and temporal dependencies. In particular, we introduce the latent prior which exploits the correlation among the latent representation to squeeze the temporal redundancy. Meanwhile, the dual spatial prior is proposed to reduce the spatial redundancy in a parallel-friendly manner. In addition, our entropy model is also versatile. Besides estimating the probability distribution, our entropy model also generates the quantization step at spatial-channel-wise. This content-adaptive quantization mechanism not only helps our codec achieve the smooth rate adjustment in single model but also improves the final rate-distortion performance by dynamic bit allocation. Experimental results show that, powered by the proposed entropy model, our neural codec can achieve 18.2% bitrate saving on UVG dataset when compared with H.266 (VTM) using the highest compression ratio configuration. It makes a new milestone in the development of neural video codec. The codes are at https://github.com/microsoft/DCVC.

## CCS CONCEPTS

• **Computing methodologies** → **Reconstruction**; **Computer vision problems**.

## KEYWORDS

Video compression, entropy model, quantization

**ACM Reference Format:**
Jiahao Li, Bin Li, and Yan Lu. 2022. Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), October 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3503161.3547845

## 1 INTRODUCTION

Recent years have witnessed the flourish of neural image codec. During the development, many works focus on the design of entropy model to accurately predict the probability distribution of the quantized latent representation, like factorized model [7], hyper prior [8], auto-regressive prior [36], mixture Gaussian model [12], transformer-based model [23], and so on. Benefited from these continuously improved entropy models, the compression ratio of neural image codec has outperformed the best traditional codec H.266 intra coding [10]. Inspired by the success of neural image codec, recently the neural video codec attracts more and more attentions.

Most existing works on neural video codec can be roughly classified into three categories: residual coding-based, conditional coding-based, and 3D autoencoder-based solutions. Among them, many methods [6, 14, 19, 20, 28–30, 32, 34, 39, 40, 46, 49] belong to the residual coding-based solution. The residual coding comes from the traditional hybrid video codec. Specifically, the motion-compensated prediction is first generated, and then its residual with the current frame is coded. For conditional coding-based solutions [24–27], the temporal frame or feature is served as condition for the coding of the current frame. When compared with residual coding, conditional coding has lower or equal entropy bound [24]. As for 3D autoencoder-based solution [15, 37, 42], it is a natural extension of neural image codec by expanding the input dimension. But it brings larger encoding delay and significantly increases the memory cost. In a summary, most these existing works focus on how to generate the optimized latent representation by exploring different data flows or network structures. As for the entropy model, they usually directly use the ready-made solutions (e.g., hyper prior [8] and auto-regressive prior [36]) from neural image codec to code the latent representation. The spatial-temporal correlation has not been fully explored in the design of entropy model for video. Thus, the RD (rate-distortion) performance of previous SOTA (state-of-the-art) neural video codec [41] is limited and only slightly better than H.265, which was released in 2013.

Therefore, this paper proposes a comprehensive entropy model which can efficiently leveraging both spatial and temporal correlations, and then helps the neural video codec outperform the latest traditional standard H.266. In particular, we introduce the latent prior and dual spatial prior. The latent prior explores the temporal correlation of the latent representation across frames. The quantized latent representation of the previous frame is used to predict the distribution of that in the current frame. Via the cascaded training strategy, the propagation chain of latent representation is formed. It enables us to build the implicit connection between the latent representation of the current frame and that of the long-range

*MM '22, October 10–14, 2022, Lisboa, Portugal*

reference frame. Such connection helps the neural codec further squeeze the temporal redundancy among the latent representation.

In our entropy model, the dual spatial prior is proposed to reduce the spatial redundancy. Most existing neural codecs rely on the auto-regressive prior [36] to explore the spatial correlation. However, auto-regressive prior is a serialized solution and follows a strict scaning order. Such kind of solution is parallel-unfriendly and inferences in a very slow speed. By contrast, our dual spatial prior is a two-step coding solution following the checkerboard context model [16], which is much more time-efficient. In [16], all channels use the same coding order (even positions are always first coded and then are used as context for the odd positions). It cannot efficiently cope with various video contents because sometimes coding the even positions first has worse RD performance than coding the odd positions first. Thus, to solve this problem, our dual spatial prior introduces the mechanism that first codes the half latent representation of both odd and even positions, and then the coding of the left latent representation can benefit from the contexts from all positions. At the same time, the correlation across the channel is also exploited during the two-step coding. Without bringing extra coding dependency, out dual spatial prior makes the scope of spatial context doubled and exploits the channel context. These will result in more accurate prediction on distribution.

For neural video codec, another challenge is how to achieve smooth rate adjustment in single model. For traditional codec, it is achieved by adjusting the quantization parameter. However, most neural codecs lack such capability and use fixed quantization step (QS). To achieve different rates, the codec needs to be retrained. It brings huge training and model storage burden. To solve this problem, we introduce an adaptive quantization mechanism at multi-granularity levels, which is powered by our entropy model. In our design, the whole QS is determined at three different granularities. First, the global QS is set by the user for the specific target rate. Then it is multiplied by the channel-wise QS because different channels contain information with different importance, similar to the channel attention mechanism [18]. At last, the spatial-channel-wise QS generated by our entropy model is multiplied. This can help our codec cope with various video contents and achieve precise rate adjustment at each position. In addition, it is noted that using entropy model to learn the QS not only helps our codec obtain the capability of smooth rate adjustment in single model but also improves the final RD performance. This is because the entropy model will learn to allocate more bits to the more important contents which are vital for the reconstruction of the current and following frames. This kind of content-adaptive quantization mechanism enables the dynamic bit allocation to boost the final compression ratio.

Powered by our versatile entropy model, our neural codec with only single model can achieve significant bitrate saving over previous SOTA neural video codecs. For example, there is a significant 57.1% bitrate saving over DCVC [27] on UVG dataset [3]. Better yet, our neural codec has outperformed the best traditional codec H.266 (VTM) [4, 10], which uses the low delay configuration with the highest compression ratio setting. For UVG dataset, an average of 18.2% bitrate saving is achieved over H.266 (VTM), when oriented to PSNR. If oriented to MS-SSIM, the corresponding bitrate saving is 35.1%. These substantial improvements show that our model makes a new milestone in the development of neural video codec.

Our contributions are summarized as follows:

- For neural video codec, we design a powerful and parallel-friendly entropy model to improve the prediction on probability distribution. The proposed latent prior and dual spatial prior can efficiently capture the temporal and spatial dependency, respectively. This helps us further squeeze the redundancy in video.
- Our entropy model is also versatile. Besides the distribution parameters, it also generates the QS at spatial-channel-wise. Via the multi-granularity quantization mechanism, our neural codec is able to achieve smooth rate adjustment in single model. Meanwhile, the QS at spatial-channel-wise is content-adaptive and improves the final RD performance by dynamic bit allocation.
- Powered by the proposed entropy model, our neural video codec pushes the compression ratio to a new height. To the best of our knowledge, our codec is the first end-to-end neural video codec to exceed H.266 (VTM) using the highest compression ratio configuration. The bitrate saving over H.266 (VTM) is 18.2% on UVG dataset in terms of PSNR. If oriented to MS-SSIM, the bitrate saving is even higher.

# 2 RELATED WORK

## 2.1 Neural Image Compression

The neural image codec has developed rapidly in recent years. The early work [43] uses a compressive autoencoder-based framework to achieve similar RD performance with JPEG 2000. Recently, many works focus on the design of entropy model. Ballé *et al.* [7] proposed the factorized model and got better RD performance than JPEG 2000. The hyper prior [8] introduces the hierarchical design and uses additional bits to estimate the distribution, which gets comparable results with H.265. Subsequently, the auto-regressive prior [36] was proposed to explore the spatial correlation. It obtains higher compression ratio together with hyper prior. However, auto-regressive prior inferences in a serialized order for all spatial positions, and thus it is quite slow. The checkerboard context model [16] solves the complexity problem by introducing the two-step coding. In addition, to further boost the compression ratio, the mixture Gaussian model [12] was proposed and its RD results are on par with H.266. Recently, the vision transformer attracts lots of attention, and the corresponding entropy model [23] helps the neural image codec exceed H.266 intra coding.

## 2.2 Neural Video Compression

The success of neural image codec also pushes the development of neural video codec. The pioneering work DVC [33] follows the traditional codec, and uses the residual coding-based framework where motion-compensated prediction is first generated and then the residual is coded using hyper prior [8]. With the help of auto-regressive prior [36], its following work DVCPro achieves higher compression ratio.

Most recent works follow this motion estimation [22, 38] and residual coding-based framework. Then more advanced networks structures are proposed for generating the optimized residual or motion. For example, the residual is adaptively scaled by learned parameter in [50]. The optical flow estimation in scale space [6]

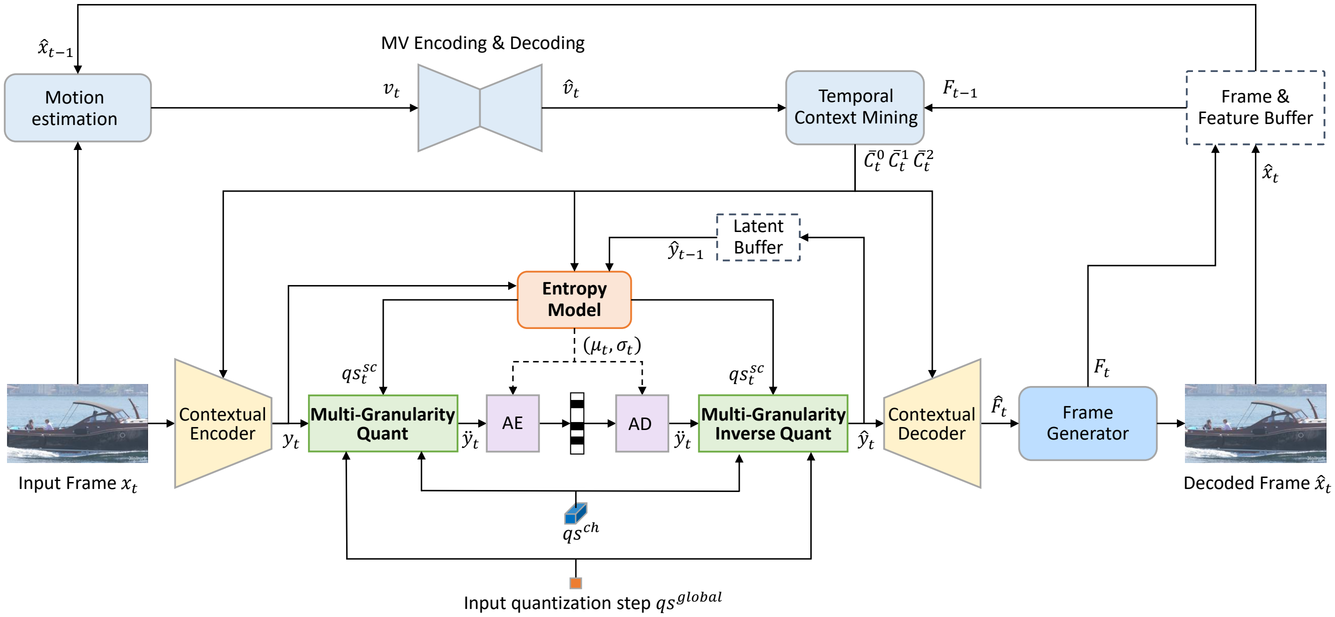


**Figure 1: The overall framework of our method. Quant shorts for quantization. AE and AD are arithmetic encoder and decoder. The entropy model and quantization mechanism for coding motion vector follow the similar design with those of $y_t$, and we omit them for simplification.**

was proposed to reduce the residual energy in fast motion area. In [19], the rate distortion optimization is applied to improve the coding of motion. The deformable compensation [20] is used to improve the prediction in feature space. Lin *et al.* [28] proposed using multiple reference frames to reduce the residual energy. In [28, 39], the motion prediction is introduced to improve the coding efficiency of motion.

Besides the residual coding, other coding frameworks are also investigated. For example, 3D autoencoder [15, 37, 42] was proposed to encode the multiple frames simultaneously. It is a natural extension of neural image codec by expanding the input dimension. However, such kind of framework will bring significant encoding delay and is not suitable for real-time scenarios. Another emerging coding framework is the conditional coding whose entropy bound is lower than or equal to residual coding [24]. For example, Ladune *et al.* [24–26] used the conditional coding to code the foreground contents. In DCVC [27], the condition is the extensible high-dimension feature rather than the 3-dimension predicted frame. The following work [41] further boosts the compression ratio by introducing the feature propagation and multi-scale temporal contexts.

However, most existing neural video codecs focus on how to generate the optimized latent representation and design the network structures therein. As for the entropy model used for coding the latent representation, the ready-made solutions from neural image codec are directly used. In this paper, we focus on the entropy model design by efficiently leveraging both spatial and temporal correlations. Actually, some works also have begun to investigate it. For example, the conditional entropy coding was proposed in [31]. The temporal context prior extracted from temporal feature is used in [27, 41]. Yang *et al.* [49] proposed the recurrent entropy model. However, these works [27, 31, 41, 49] focus more on utilizing temporal correlation. Although the work in [27] also investigates the spatial correlation, the auto-regressive prior is used and leads to a very slow coding speed. An entropy model which not only fully exploits the spatial-temporal correlation but also has low complexity is desired. To meet this requirement, we specially design the latent prior and time-efficient dual spatial prior to equip the entropy model, and push the compression ratio to a new height. In addition, all these methods [27, 31, 41, 49] need to train separate model for each rate point. By contrast, we design a multi-granularity quantization, which is powered by our entropy model. This content-adaptive quantization mechanism helps our codec achieve smooth rate adjustment in single model. The rate adjustment in single model was investigated for neural image codec via the gain unit [13]. And yet, to the best of our knowledge, our codec is the first neural video codec to obtain such capability via our multi-granularity quantization.

# 3 PROPOSED METHOD

## 3.1 Framework Overview

To achieve higher compression ratio for the input frame $x_t$ ($t$ is the frame index), we adopt the conditional coding-based framework rather than the residual coding-based framework. Specifically, we follow the DCVC [27] and its improved work [41], and then redesign the core modules therein. The framework of our codec is presented in Fig. 1. As shown in this figure, the entire coding process can be roughly divided into three steps: temporal context generation, contextual encoding/decoding, and reconstruction.

**Temporal context generation.** To fully explore the temporal correlation, we generate the multi-scale contexts $\bar{C}_t^0$, $\bar{C}_t^1$,$\bar{C}_t^2$ at different resolutions via the temporal context mining module (more

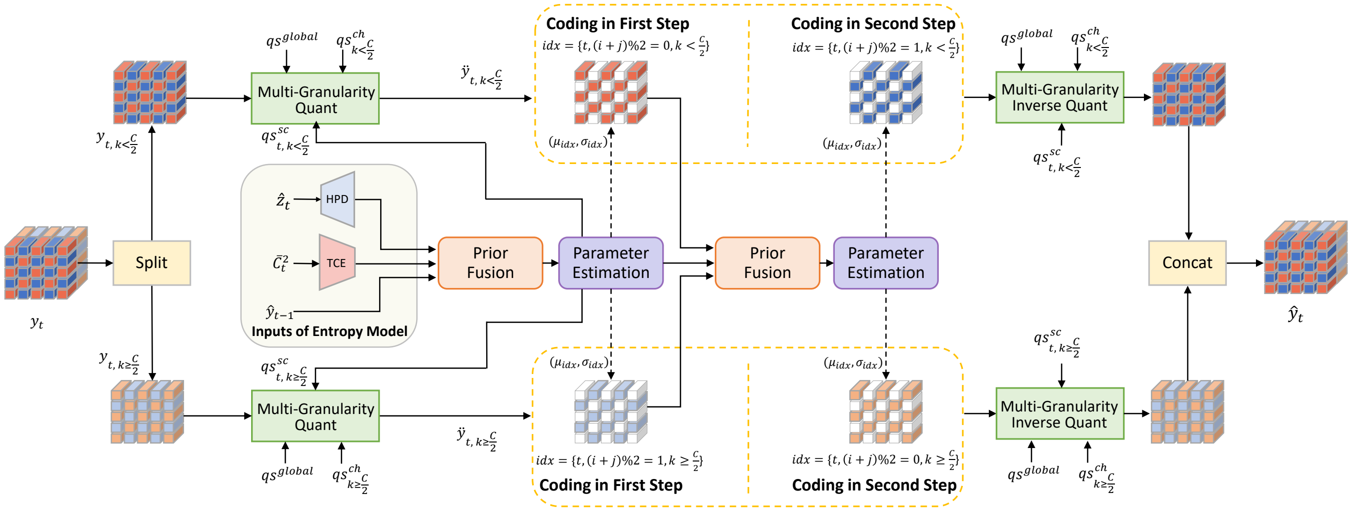

$y_t$
Split
$y_{t,k<\frac{C}{2}}$
$y_{t,k\geq\frac{C}{2}}$
$qs^{global}$
$qs^{ch}_{k<\frac{C}{2}}$
Multi-Granularity Quant
$qs^{sc}_{t,k<\frac{C}{2}}$
$\ddot{y}_{t,k<\frac{C}{2}}$
$qs^{sc}_{t,k\geq\frac{C}{2}}$
Multi-Granularity Quant
$qs^{global}$
$qs^{ch}_{k\geq\frac{C}{2}}$
$\ddot{y}_{t,k\geq\frac{C}{2}}$
$\hat{z}_t$
HPD
$\bar{C}_t^2$
TCE
$\hat{y}_{t-1}$
Inputs of Entropy Model
Prior Fusion
Parameter Estimation
Prior Fusion
Parameter Estimation
Coding in First Step
$idx = \{t, (i+j)\%2 = 0, k < \frac{C}{2}\}$
Coding in Second Step
$idx = \{t, (i+j)\%2 = 1, k < \frac{C}{2}\}$
$(\mu_{idx}, \sigma_{idx})$
$(\mu_{idx}, \sigma_{idx})$
$(\mu_{idx}, \sigma_{idx})$
$(\mu_{idx}, \sigma_{idx})$
$idx = \{t, (i+j)\%2 = 1, k \geq \frac{C}{2}\}$
Coding in First Step
$idx = \{t, (i+j)\%2 = 0, k \geq \frac{C}{2}\}$
Coding in Second Step
$qs^{global}$
$qs^{ch}_{k<\frac{C}{2}}$
Multi-Granularity Inverse Quant
$qs^{sc}_{t,k<\frac{C}{2}}$
Concat
$\hat{y}_t$
$qs^{sc}_{t,k\geq\frac{C}{2}}$
Multi-Granularity Inverse Quant
$qs^{global}$
$qs^{ch}_{k\geq\frac{C}{2}}$

**Figure 2: The illustration of our entropy model and the coding of quantized latent representation. $i$, $j$, and $k$ indicate the height, width, and channel indexes, respectively. $C$ is the channel number. $qs^{global}$, $qs^{ch}$, and $qs^{sc}$ are global, channel-wise, spatial-channel-wise quantization steps, respectively. HPD shorts for hyper prior decoder. TCE means temporal context encoder. For simplification, the arithmetic encoder and decoder following the parameter estimation are omitted in the two-step coding.**

details of this module can be found in [41]). To carry richer information, we also use the temporal feature $F_{t-1}$ rather than previous decoded frame $\hat{x}_{t-1}$ as the module input. As for motion estimation, we adopt the light-weight SPyNet [38] for acceleration.

**Contextual encoding/decoding.** Conditioned by the multi-scale contexts, the current frame $x_t$ is transformed into latent representation $y_t$ by the contextual encoder. To achieve bitrate saving, the $y_t$ is quantized to $\ddot{y}_t$ before being sent to the arithmetic encoder which generates the bit-stream. During the decoding, $\ddot{y}_t$ is decoded from bit-steam by arithmetic decoder and inversely quantized to $\hat{y}_t$. Also conditioned on the multi-scale contexts, the contextual decoder decodes the high-resolution feature $\hat{F}_t$ from $\hat{y}_t$. In this encoding/decoding process, how to accurately estimate the distribution of $\ddot{y}_t$ by the entropy model is vital for bitrate reduction. To this end, we propose the hybrid spatial-temporal entropy model (Section 3.2). To support smooth rate adjustment in single model, the multi-granularity quantization powered by our entropy model is proposed (Section 3.3).

**Reconstruction.** After obtaining the high-resolution feature $\hat{F}_t$, our target is generating the high-quality reconstructed frame $\hat{x}_t$ via the frame generator. Different from DCVC [27] and [41] only using plain residual blocks [17], we proposing using the W-Net [47] based structure which ties two U-Nets. Such kind of network design can effectively enlarge the receptive field of model with acceptable complexity. This results in stronger generation ability for model.

## 3.2 Hybrid Spatial-Temporal Entropy Model

For arithmetic coding, it needs to know the probability mass function (PMF) of $\ddot{y}_t$ to code it. However, we do not know its true PMF $p(\ddot{y}_t)$ and usually approximate it with an estimated PMF $q(\ddot{y}_t)$. The cross-entropy $\mathbb{E}_{\ddot{y}_t \sim p}[-log_2 q(\ddot{y}_t)]$ captures the average number of bits needed by the arithmetic coding without considering the negligible overhead. In this paper, we follow the existing work [2] and assume that $q(\ddot{y}_t)$ follows the Laplace distribution. Thus, our target is designing an entropy model which can accurately estimate the distribution parameter of $q(\ddot{y}_t)$ to reduce the cross-entropy.

To improve the estimation, for each element $\ddot{y}_{t,i,j,k}$ ($i$, $j$, and $k$ are the height, width, and channel indexes) therein, we need to fully mine the correlation between it and its known information, e.g., the previous decoded latent representation, temporal context feature, and so on. Theoretically, $\ddot{y}_{t,i,j,k}$ may correlate to all these information from all previous decoded positions. For traditional codec, it is unable to explicitly exploit such correlation due to the huge space. Thus, traditional codec usually uses simple handcrafted rules to use context from a few neighbour positions. By contrast, the deep learning enables the capability of automatically mining the correlation in huge space.

Thus, we propose feeding manifold inputs to the entropy model, and let the model extract the complementary information from the rich high-dimensional inputs. The illustration of our entropy model is shown in Fig. 2. As shown in this figure, the inputs not only contain the commonly-used hyper prior $\hat{z}_t$ and the temporal context $\bar{C}_t^2$, but also include the $\hat{y}_{t-1}$ as the latent prior. Although both $\bar{C}_t^2$ and $\hat{y}_{t-1}$ are from temporal direction, they have different characteristics. For example, $\bar{C}_t^2$ at 4x down-sampled resolution usually contains lots of motion information (more details can be found in [41]). By contrast, $\hat{y}_{t-1}$ is in the latent representation domain at 16x down-sampled resolution, and has more similar characteristics with $y_t$. Thus, $\bar{C}_t^2$ and $\hat{y}_{t-1}$ can provide the complementary auxiliary information to improve estimation. In addition, it is noted that we adopt the cascaded training strategy [11, 41], where the gradients will back propagate to multiple frames. Under such training strategy, the propagation chain of latent representation is formed. It means that the connection between the latent representation of

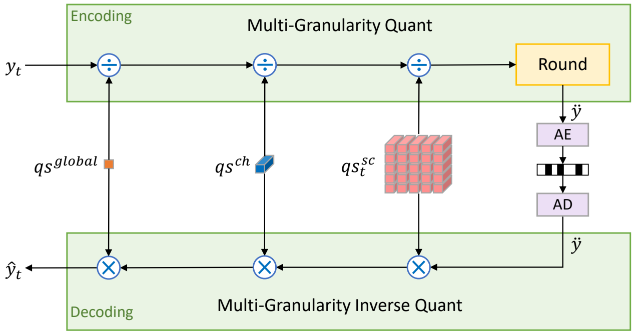


**Figure 3: Multi-granularity quantization and the corresponding inverse quantization.**

the current frame and that of long-range reference frame is also built. Such connection is very helpful for extracting the correlation across the latent representation of multiple frames, and then results in more accurate prediction on distribution.

Besides using the latent prior $\hat{y}_{t-1}$ to enrich the input, our entropy model also adopts the dual spatial prior to exploit the spatial correlation. To pursue a time-efficient mechanism, we follow the checkerboard model [16] rather than the commonly-used autoregressive model [36] which seriously slows down the coding speed. However, all channels in the original checkerboard model use the same coding order, namely the even positions are always first coded and then used as context for the coding of odd positions. Such coding order cannot handle various videos because sometimes coding the even positions first has worse RD performance than coding the odd positions first. Therefore, we design the dual spatial prior, where half channels in both even and odd positions are first coded at the same time.

As shown in Fig. 2, the latent representation is split into two chunks along the channel dimension. During the first-step coding, the above branch in Fig. 2 will code the even positions $\ddot{y}_{t,(i+j)\%2=0,k<\frac{C}{2}}$ in the first chunk. Simultaneously, the below branch codes the odd positions $\ddot{y}_{t,(i+j)\%2=1,k\geq\frac{C}{2}}$ in the second chunk. The uncoded positions (i.e., the white regions in Fig. 2) in these two chunks are set as zero. After the first-step coding, the coded $\ddot{y}_{t,(i+j)\%2=0,k<\frac{C}{2}}$ and $\ddot{y}_{t,(i+j)\%2=1,k\geq\frac{C}{2}}$ are fused together and then further generate the contexts for the second-step coding. During the second-step coding, the left positions in each chunk are coded. The above branch codes the odd positions $\ddot{y}_{t,(i+j)\%2=1,k<\frac{C}{2}}$ in the first chunk and the below branch codes the even positions $\ddot{y}_{t,(i+j)\%2=0,k\geq\frac{C}{2}}$ in the second chunk. As shown in Fig. 2, this coding manner enables that the second-step coding can benefit from the contexts from all positions. When compared with original checkerboard model [16], the scope of spatial context is doubled and results in more accurate prediction on distribution. It is noted that, for both steps, the first chunk and second chunk will be added and then sent to the arithmetic encoder. Thus, our dual spatial prior will not bring any additional coding delay when compared with [16].

In addition, from the perspective of channel dimension, our dual spatial prior also mines the correlation across channels. The coded $\ddot{y}_{t,(i+j)\%2=0,k<\frac{C}{2}}$ during the first-step coding process can be also used as the condition for the coding of $\ddot{y}_{t,(i+j)\%2=0,k\geq\frac{C}{2}}$ during the second-step coding process. It is similar for the coding of $\ddot{y}_{t,(i+j)\%2=1,k\geq\frac{C}{2}}$ and $\ddot{y}_{t,(i+j)\%2=1,k<\frac{C}{2}}$, where the coding direction of channel is inverse. In a summary, our proposed dual spatial prior further squeezes the redundancy in $\ddot{y}_t$ by more efficiently exploiting the correlation across the spatial and channel positions.

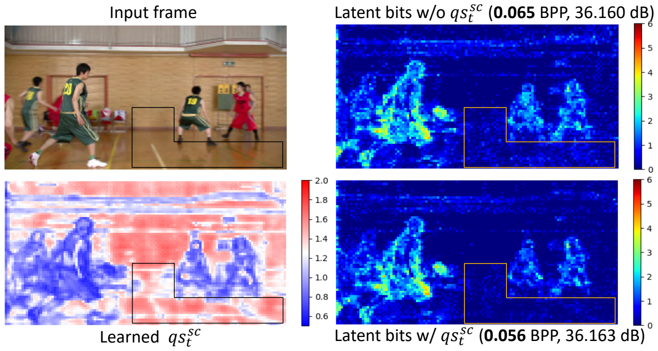


**Figure 4: Effect of $qs^{sc}$. BPP means bits per pixel. In this example, $qs^{sc}$ brings 13.8% BPP reduction under similar quality. Zoom in for better view.**

## 3.3 Rate Adjustment in Single Model

It is a big pain that most existing neural codecs cannot handle rate adjustment in single model. To achieve different rates, the model needs to be retrained by adjusting the weight in the RD loss. It will bring large training cost and model storage burden. Such shortcoming calls for a mechanism supporting neural codec to achieve wide rate range in single model. To this end, we propose an adaptive quantization mechanism to enable this capability.

As shown in Fig. 3, our multi-granularity quantization involves three different kinds of quantization step (QP): the global QS $qs^{global}$, the channel-wise QS $qs^{ch}$, and the spatial-channel-wise QS $qs^{sc}$. The $qs^{global}$ is only a single value and is set from the user input for controlling the target rate. As all positions take the same QS, the $qs^{global}$ brings a coarse quantization effect. Thus, motivated by the channel attention mechanism [18], we also design a modulator $qs^{ch}$ to scale the QS at different channels because different channels carry information with different importance. However, the different spatial positions also have different characteristics due to the various video contents. Thus, we follow [21] and learn the spatial-channel-wise $qs^{sc}$ to achieve the precise adjustment on each position.

The $qs^{sc}$ is generated by our entropy model, as shown in Fig. 2. It is noted that, for each frame, it is dynamically changed to adapt the video contents. Such design not only helps us achieve smooth rate adjustment but also improves the final RD performance by content-adaptive bit allocation. The important information which is vital for the reconstruction or is referenced by the coding of the subsequent frames will be allocated with smaller QS. We show a visualization example of $qs^{sc}$ in Fig. 4. In this example, the model learns that the moving players are more important and produces smaller QS for these regions. By contrast, the backgrounds have larger QS, which brings considerable bitrate saving (as shown in region indicated by the yellow line). The ablation study in section 4.3 shows $qs^{sc}$ can achieve substantial improvements on multiple datasets.

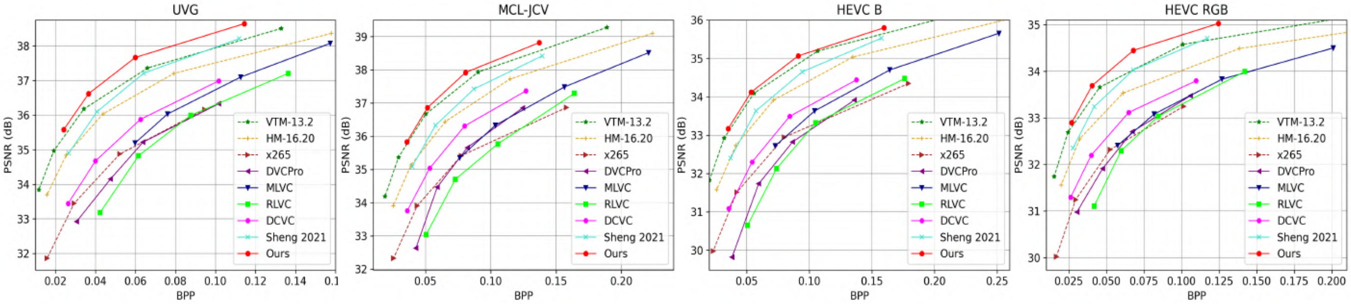


**Figure 5: PSNR and BPP curves.**

**Table 1: BD-Rate (%) comparison for PSNR. The anchor is VTM-13.2.**

| | UVG | MCL-JCV | HEVC B | HEVC C | HEVC D | HEVC E | HEVC RGB | Average |
|---|---|---|---|---|---|---|---|---|
| VTM-13.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| HM-16.20 | 40.5 | 45.4 | 40.4 | 40.9 | 36.0 | 46.2 | 42.1 | 41.6 |
| x265 | 191.5 | 160.3 | 143.4 | 105.2 | 96.1 | 128.4 | 151.2 | 139.4 |
| DVCPro [34] | 227.0 | 180.8 | 209.8 | 220.6 | 166.4 | 446.2 | 178.5 | 232.8 |
| RLVC [49] | 224.2 | 214.3 | 207.0 | 212.4 | 149.2 | 392.8 | 195.5 | 227.9 |
| MLVC [28] | 113.6 | 124.0 | 118.0 | 213.7 | 166.5 | 237.6 | 151.2 | 160.7 |
| DCVC [27] | 126.1 | 98.2 | 115.0 | 150.8 | 109.6 | 266.2 | 109.6 | 139.4 |
| Sheng 2021 [41] | 17.1 | 30.6 | 28.5 | 60.5 | 27.8 | 67.3 | 17.9 | 35.7 |
| Ours | −18.2 | −6.4 | −5.1 | 15.0 | −8.9 | 7.1 | −16.4 | −4.7 |

During the decoding, the corresponding inverse quantization is applied. It is noted that, the $qs^{global}$ needs to be transmitted to the decoder along with the bit-stream. However, this overhead is negligible as only single number is transmitted for each frame or video (flexible setting for user). The modulator $qs^{ch}$ belongs to a part of our neural codec and is learned during the training.

# 4 EXPERIMENTAL RESULTS

## 4.1 Experimental Setup

**Datasets.** We use Vimeo-90k [48] for training. The videos are randomly cropped into 256x256 patches. For testing, we use the same test videos as [41]. All these test videos are widely used in the evaluation of traditional and neural video codecs, including HEVC Class B, C, D, E, and RGB. In addition, the 1080p videos from UVG [35] and MCL-JCV [44] datasets are also tested.

**Test conditions.** We test 96 frames for each video. The intra period is set to 32 rather than 10 or 12. The reason is that intra period 32 gets closer to the practical usage in the real applications. For example, when compared with intra period 12, intra period 32 has an average of 23.8% [41] bitrate saving for HM. We follow the low delay encoding settings as most existing works [27, 33, 34, 41]. The compression ratio is measured by BD-Rate [9], where negative numbers indicate bitrate saving and positive numbers indicate bitrate increase.

Besides x265 [5] (*veryslow* preset is used), our benchmarks include HM-16.20 [1] and VTM-13.2 [4], which represent the best encoder of H.265 and H.266, respectively. For HM and VTM, we follow [41] and use the configuration with the highest compression ratio. We also compare previous SOTA neural video codecs including DVCPro [34], MLVC [28], RLVC [49], DCVC [27], as well as Sheng 2021 [41].

**Implementation and training details.** The entropy model and quantization for the latent representation of motion vector follow those of $y_t$. The only difference is the input of entropy model. In the coding of motion vector, the inputs are the corresponding hyper prior and the latent prior, i.e., the quantized latent representation of motion vector from the previous frame. There is no temporal context prior for the coding of motion vector because the generation of temporal context depends on the decoded motion vector. In addition, as we target at single model handling multiple rates, we also train a neural image codec supporting such capability for intra coding.

During the training, the loss function includes the distortion and rate: $Loss = \lambda \cdot D + R$. $D$ refers to the distortion between the input frame and the reconstructed frame. The distortion can be $L_2$ loss or MS-SSIM [45] for different visual targets. $R$ represents the bits used for encoding $\ddot{y}_t$ and the quantized latent representation of motion vector, both associated with the bits used for encoding their corresponding hyper prior.

We adopt the multi-stage training, same as [41]. In addition, to train single model supporting rate adjustment, we use different $\lambda$ values in different optimization steps. For simplifying the training process, 4 $\lambda$ values (85, 170, 380, 840) are used. During the training, 4 $qs^{global}$ values will be learned via the RD loss with each corresponding $\lambda$ value. We trained image and video models separately

**Table 2: BD-Rate (%) comparison for MS-SSIM. The anchor is VTM-13.2.**

| | UVG | MCL-JCV | HEVC B | HEVC C | HEVC D | HEVC E | HEVC RGB | Average |
|---|---|---|---|---|---|---|---|---|
| VTM-13.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| HM-16.20 | 36.9 | 43.7 | 36.7 | 38.7 | 34.9 | 40.5 | 37.2 | 38.4 |
| x265 | 150.5 | 137.6 | 129.3 | 109.5 | 101.8 | 109.0 | 121.9 | 122.8 |
| DVCPro [34] | 68.1 | 37.8 | 61.7 | 59.1 | 23.9 | 212.5 | 57.3 | 74.3 |
| RLVC [49] | 83.9 | 72.1 | 66.6 | 76.5 | 34.1 | 268.4 | 60.5 | 94.6 |
| DCVC [27] | 33.6 | 4.7 | 31.0 | 22.8 | 1.2 | 124.7 | 36.5 | 36.4 |
| Sheng 2021 [41] | −10.1 | −24.4 | −24.1 | −23.3 | −37.3 | −8.0 | −25.9 | −21.9 |
| Ours | −35.1 | −46.8 | −48.1 | −44.6 | −55.7 | −47.5 | −47.0 | −46.4 |

and they have different $qs^{global}$ values. It is noted that, although we only use 4 $\lambda$ values during the training, the model still can achieve smooth rate adjustment by manually adjusting the $qs^{global}$ during the testing, and the corresponding study is presented in Section 4.4.

## 4.2 Comparisons with Previous SOTA Methods

Table 1 and Table 2 show the BD-Rate (%) comparisons in terms of PSNR and MS-SSIM, respectively. The best traditional codec VTM is used as anchor. From Table 1, we can find that our neural codec achieves an average of 4.7% bitrate saving over VTM on all datasets. By contrast, the second best method Sheng 2021 [41] is far behind than VTM and has 35.7% bitrate increase. To the best of our knowledge, this is the first end-to-end neural video codec that outperforms VTM using the highest compression ratio configuration, which is an important milestone in the development of neural video codec. In particular, our neural codec performs better for 1080p videos (HEVC B, HEVC RGB, UVG, MCL-JCV). Fig. 5 shows RD curves on these datasets. We can find our codec consumes the least bits under the same quality. These results verify the effectiveness of our entropy model on exploiting the correlation among the volumed video data. In addition, the high-resolution video will be more popular in the future and the advantage of our neural codec will be more obvious. When oriented to MS-SSIM, our neural video codec has larger improvement. As shown in Table 2, we achieve an average of 46.4% bitrate saving over VTM on all datasets.

## 4.3 Ablation Study

To verify the effectiveness of each component, we conduct a comprehensive ablation study. We analyze the effect of entropy model input, our dual spatial prior, the multi-granularity quantization, as well as the design on frame generator. For simplification, we only use HEVC testsets in ablation study. The comparisons are measured by BD-Rate (%).

**The inputs of entropy model.** As shown in Fig. 2, our entropy model contains three different kinds of inputs: hyper prior, temporal context prior, and latent prior. Table 3 compares the effectiveness of these inputs. When removing our latent prior, there is 10.1% bitrate increase. If only enabling one prior input, the first and second most important inputs are hyper prior (15.6%) and our latent prior (19.3%). From these comparisons, we can find that the hyper prior is still the most important. But enriching the entropy model input via our latent prior also brings significant bitrate saving.

**Table 3: Effect of entropy model inputs.**

| Hyper prior | Temporal context prior | Latent prior | B | C | D | E | RGB | Average |
|---|---|---|---|---|---|---|---|---|
| ✓ | ✓ | ✓ | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| ✓ | ✓ | ✗ | 9.5 | 5.6 | 6.4 | 18.4 | 10.7 | 10.1 |
| ✓ | ✗ | ✓ | 9.2 | 8.8 | 9.1 | 15.0 | 9.1 | 10.2 |
| ✗ | ✓ | ✓ | 19.5 | 9.1 | 11.9 | 27.2 | 20.9 | 17.7 |
| ✓ | ✗ | ✗ | 12.4 | 14.7 | 17.9 | 23.4 | 9.4 | 15.6 |
| ✗ | ✓ | ✗ | 33.8 | 29.3 | 31.6 | 54.3 | 36.2 | 37.0 |
| ✗ | ✗ | ✓ | 19.5 | 19.3 | 19.2 | 21.1 | 17.3 | 19.3 |

**Table 4: Effect of different spatial priors.**

| | B | C | D | E | RGB | Average |
|---|---|---|---|---|---|---|
| Proposed dual spatial prior | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Checkerboard prior | 5.6 | 4.1 | 6.0 | 11.0 | 5.5 | 6.2 |
| No spatial prior | 11.9 | 11.1 | 11.6 | 29.7 | 6.4 | 14.1 |
| Auto-regressive prior | −14.8 | −11.4 | −11.2 | −5.6 | −16.9 | −12.0 |

**Different spatial priors.** Table 4 compares the effect of different spatial prior modules on exploring the inner correlation in $\ddot{y}_t$. The tested spatial context models include our proposed dual spatial prior, checkerboard prior [16], and the parallel-unfriendly auto-regressive prior [36]. From this table, we can find that our dual spatial prior could bring 14.1% bitrate saving. And there is 6.2% improvement over the checkerboard prior. This verifies the benefit of enlarging the scope of spatial context and exploiting the cross-channel correlation. In addition, we also find that the auto-regressive prior could further save the bitrate by a large margin (12.0%). However, considering the very slow encoding and decoding speed, we still do not adopt it in our neural codec.

**Multi-granularity quantization.** As introduced in Section 3.3, the multi-granularity quantization powered by our entropy model not only helps us achieve rate adjustment in single model but also improves the final RD performance by content-adaptive bit allocation. Table 5 shows the BD-rate comparison. From this table, we can find that there is 11.8% loss if disabling the whole

**Table 5: RD comparison of multi-granularity quantization.**

| | B | C | D | E | RGB | Average |
|---|---|---|---|---|---|---|
| Multi-granularity quantization | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| w/o $qs_t^{sc}$ | 9.7 | 10.1 | 10.6 | 8.3 | 5.6 | 8.9 |
| w/o multi-granularity quantization | 9.9 | 15.1 | 14.6 | 11.7 | 7.8 | 11.8 |

**Table 6: Study on different frame generators.**

| | B | C | D | E | RGB | Average |
|---|---|---|---|---|---|---|
| W-Net | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| U-Net | 6.3 | 7.6 | 9.1 | 7.7 | 8.6 | 7.9 |
| 2 Residual Blocks [27, 41] | 11.7 | 9.9 | 11.0 | 13.4 | 9.1 | 11.0 |
| 1 Residual Block | 12.5 | 12.5 | 13.4 | 14.5 | 10.9 | 12.8 |

multi-granularity quantization. When only removing the spatial-channel-wise quantization step $qs_t^{sc}$, there is a significant 8.9% bit increase. It shows that the $qs_t^{sc}$ play an important role in multi-granularity quantization and it is necessary to design a dynamic bit allocation which can adapt to various video contents.

**Frame generator design.** To improve the generation ability of neural codec, our frame generator uses W-Net based structure to enlarge the receptive field. To verify its effectiveness, we compare the frame generator using different networks, as shown in Table 6. We compare W-Net, U-Net, and residual block [27, 41] based structures. From this table, we can find that W-Net has more than 10% bitrate saving than the plain network using residual blocks. The W-Net which ties two U-Nets is also 7.9% better than the single U-Net. Thus, it is possible that larger bitrate saving can be achieved by using more U-Nets. However, considering the complexity, we currently use W-Net.

## 4.4 Smooth Rate Adjustment in Single Model

Table 5 shows the RD improvement of multi-granularity quantization. This sub-section investigates whether our multi-granularity quantization can achieve smooth rate adjustment. For our codec, the global quantization step $qs^{global}$ can be flexibly adjusted during the testing. It serves the similar role of quantization parameter in traditional video codecs. Fig. 6 shows the results of our codec using the learned 4 $qs^{global}$ values which are guided by RD loss using the 4 $\lambda$ values during the training. In addition, we also manually generate 30 $qs^{global}$ values by the interpolation between the maximum and minimum of the learned $qs^{global}$ values. From the results at the 30 rate points, we can find that our single model could achieve smooth rate adjustment without any outlier. By contrast, DCVC [27] and Sheng 2021 [41] need different models for each rate point.

## 4.5 Model Complexity

We compare the model complexity in model size, MACs (multiply–accumulate operations), encoding time, and decoding time with previous SOTA neural video codecs, as shown in Table 7. We use 1080p frame as input to measure these numbers. For the encoding/decoding time, we measure the time on NVIDIA V100 GPU, including the time of writing to and reading from bitstream as in [41]. As aforementioned, our model supports rate adjustment in single model. Thus, we significantly reduce the model training and storage burden. Because DCVC [27] uses the parallel-unfriendly auto-regression prior model, its encoding/decoding time is quite slow. By contrast, [41] and our codec are much faster. Compare with [41], our encoding/decoding time is increased a little. However, the compression ratio is pushed into a new height (from Sheng 2021 surpassing HM to ours surpassing VTM). We believe this is a price worth paying.

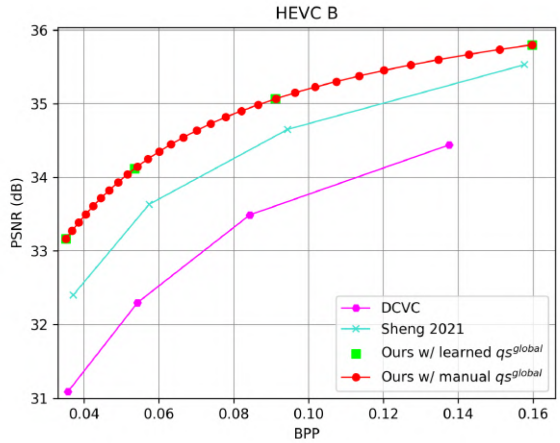


**Figure 6: Rate adjustment in single model. DCVC [27] and Sheng 2021 [41] need separate model for each rate point.**

**Table 7: Complexity comparison.**

| | Model size | MACs | Encoding time | Decoding time |
|---|---|---|---|---|
| DCVC [27] | 35.2MB $\times N$ | 2.4T | 12,260ms | 35,590ms |
| Sheng 2021 [41] | 40.9MB $\times N$ | 2.9T | 880ms | 470ms |
| Ours | 67.0MB $\times 1$ | 3.3T | 986ms | 525ms |

Note: $N$ is the number of rate points. 1080p is used for test.

## 5 CONCLUSION

In this paper, we have presented how to design an efficient entropy model which helps our neural codec not only achieve the higher compression ratio than VTM but also smoothly adjust the rates in single model. In particular, the latent prior is proposed to enrich the input of entropy model. Via building the the propagation chain, the model can exploit the correlation across the latent representation of multiple frames. The proposed dual spatial prior not only doubles the scope of spatial context in parallel-friendly manner but also further squeezes the redundancy across the channel dimension. In addition, our entropy model also generates the spatial-channel-wise quantization step. Such content-adaptive quantization mechanism helps the codec cope with various video consents well. As the core element in the multi-granularity quantization, it not only helps achieve the smooth rate adjustment but also improves the final RD performance by dynamic bit allocation. When compared with the best traditional codec VTM using the highest compression ratio configuration, an average of 18.2% bitrate saving is achieved on UVG dataset, which is an important milestone.

# REFERENCES

[1] 2022. HM-16.20. https://vcgit.hhi.fraunhofer.de/jvet/HM/. Accessed: 2022-03-02.

[2] 2022. PyTorchVideoCompression. https://github.com/ZhihaoHu/PyTorchVideoCompression. Accessed: 2022-03-02.

[3] 2022. Ultra video group test sequences. http://ultravideo.cs.tut.fi. Accessed: 2022-03-02.

[4] 2022. VTM-13.2. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/. Accessed: 2022-03-02.

[5] 2022. x265. https://www.videolan.org/developers/x265.html. Accessed: 2022-03-02.

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

[7] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. 2017. End-to-end Optimized Image Compression. In *5th International Conference on Learning Representations, ICLR 2017*.

[8] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. 2018. Variational image compression with a scale hyperprior. *6th International Conference on Learning Representations, ICLR* (2018).

[9] Gisle Bjontegaard. 2001. Calculation of average PSNR differences between RD-curves. *VCEG-M33* (2001).

[10] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J Sullivan, and Jens-Rainer Ohm. 2021. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology* 31, 10 (2021), 3736–3764.

[11] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. 2021. BasicVSR: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4947–4956.

[12] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. 2020. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7939–7948.

[13] Ze Cui, Jing Wang, Shangyin Gao, Tiansheng Guo, Yihui Feng, and Bo Bai. 2021. Asymmetric gained deep image compression with continuous rate adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10532–10541.

[14] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. 2019. Neural Inter-Frame Compression for Video Coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[15] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. 2019. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7033–7042.

[16] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. 2021. Checkerboard context model for efficient learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14771–14780.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[18] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7132–7141.

[19] Zhihao Hu, Zhenghao Chen, Dong Xu, Guo Lu, Wanli Ouyang, and Shuhang Gu. 2020. Improving deep video compression by resolution-adaptive flow coding. In *European Conference on Computer Vision*. Springer, 193–209.

[20] Zhihao Hu, Guo Lu, and Dong Xu. 2021. FVC: A New Framework towards Deep Video Compression in Feature Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1502–1511.

[21] Cong Huang, Jiahao Li, Bin Li, Dong Liu, and Yan Lu. 2022. Neural Compression-Based Feature Learning for Video Restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5872–5881.

[22] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. 2020. A lightweight optical flow CNN—Revisiting data fidelity and regularization. *IEEE transactions on pattern analysis and machine intelligence* 43, 8 (2020), 2555–2569.

[23] A Burakhan Koyuncu, Han Gao, and Eckehard Steinbach. 2022. Contextformer: A Transformer with Spatio-Channel Attention for Context Modeling in Learned Image Compression. *arXiv preprint arXiv:2203.02452* (2022).

[24] Théo Ladune, Pierrick Philippe, Wassim Hamidouche, Lu Zhang, and Olivier Déforges. 2020. Optical Flow and Mode Selection for Learning-based Video Coding. In *22nd IEEE International Workshop on Multimedia Signal Processing*.

[25] Théo Ladune, Pierrick Philippe, Wassim Hamidouche, Lu Zhang, and Olivier Déforges. 2021. Conditional Coding and Variable Bitrate for Practical Learned Video Coding. *CLIC workshop, CVPR* (2021).

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

[27] Jiahao Li, Bin Li, and Yan Lu. 2021. Deep contextual video compression. *Advances in Neural Information Processing Systems* 34 (2021).

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

[29] Haojie Liu, Ming Lu, Zhan Ma, Fan Wang, Zhihuang Xie, Xun Cao, and Yao Wang. 2020. Neural video coding using multiscale motion compensation and spatiotemporal context model. *IEEE Transactions on Circuits and Systems for Video Technology* (2020).

[30] Haojie Liu, Han Shen, Lichao Huang, Ming Lu, Tong Chen, and Zhan Ma. 2020. Learned video compression via joint spatial-temporal correlation exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 11580–11587.

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

[32] Guo Lu, Chunlei Cai, Xiaoyun Zhang, Li Chen, Wanli Ouyang, Dong Xu, and Zhiyong Gao. 2020. Content adaptive and error propagation aware deep video compression. In *European Conference on Computer Vision*. Springer, 456–472.

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

[34] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. 2020. An end-to-end learning framework for video compression. *IEEE transactions on pattern analysis and machine intelligence* (2020).

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

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

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

[38] Anurag Ranjan and Michael J Black. 2017. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4161–4170.

[39] Oren Rippel, Alexander G Anderson, Kedar Tatwawadi, Sanjay Nair, Craig Lytle, and Lubomir Bourdev. 2021. ELF-VC: Efficient Learned Flexible-Rate Video Coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 14479–14488.

[40] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G Anderson, and Lubomir Bourdev. 2019. Learned video compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3454–3463.

[41] Xihua Sheng, Jiahao Li, Bin Li, Li Li, Dong Liu, and Yan Lu. 2021. Temporal Context Mining for Learned Video Compression. *arXiv preprint arXiv:2111.13850* (2021).

[42] Wenyu Sun, Chen Tang, Weigui Li, Zhuqing Yuan, Huazhong Yang, and Yongpan Liu. 2020. High-Quality Single-Model Deep Video Compression with Frame-Conv3D and Multi-frame Differential Modulation. In *European Conference on Computer Vision (ECCV)*. Springer, 239–254.

[43] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. 2017. Lossy image compression with compressive autoencoders. *5th International Conference on Learning Representations, ICLR* (2017).

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

[45] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. 2003. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Vol. 2. Ieee, 1398–1402.

[46] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. 2018. Video compression through image interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 416–431.

[47] Xide Xia and Brian Kulis. 2017. W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint arXiv:1711.08506* (2017).

[48] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. 2019. Video Enhancement with Task-Oriented Flow. *International Journal of Computer Vision (IJCV)* 127, 8 (2019), 1106–1125.

[49] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. 2021. Learning for Video Compression with Recurrent Auto-Encoder and Recurrent Probability Model. *IEEE Journal of Selected Topics in Signal Processing* 15, 2 (2021), 388–401.

[50] Ruihan Yang, Yibo Yang, Joseph Marino, and Stephan Mandt. 2021. Hierarchical autoregressive modeling for neural video compression. *9th International Conference on Learning Representations, ICLR* (2021).

# Appendices

Appendices provide the supplementary material to our proposed hybrid spatial-temporal entropy modelling for neural video compression.

## A NETWORK ARCHITECTURE

**Contextual encoder and decoder.** The network design of our contextual encoder and decoder is shown in Fig. 7. For encoder, the inputs are the current original frame $x_t$ and the multi-scale contexts $\bar{C}_t^0$, $\bar{C}_t^1$,$\bar{C}_t^2$ at different resolutions (original, 2x downsampled, 4x downsampled) with channel 64. The output is the latent representation $y_t$ with channel 96 at 16x downsampled resolution. For decoder, the inputs contain the decoded latent representation $\hat{y}_t$. Conditioned on $\bar{C}_t^1$ and $\bar{C}_t^2$, the high-resolution feature $\hat{F}_t$ with channel 32 is decoded. For the convolutional and sub-pixel convolutional layers in Fig. 7, the $(K, Cin, Cout, S)$ indicate the kernel size, input channel number, output channel number, and stride, respectively. In Fig. 7, the bottleneck residual block comes from [41]. The kernel size and stride of convolutional layer therein are 3 and 1, respectively. Thus, we only show the input and output channel numbers for bottleneck residual block.

**Entropy model.** Fig. 8 shows the detailed network structure of our entropy model. The inputs include the the hyper prior with channel 192, the temporal context prior with channel 192, the latent prior (i.e., the decoded latent representation from the previous frame) with channel 96. The temporal context prior is generated by the temporal context encoder, as shown in Fig. 9. The network structures of hyper prior encoder and decoder are shown in Fig. 10.

In the first-step coding, the entropy model not only estimates the mean and scale values of the probability distribution for the quantized latent representations $\ddot{y}_{t,(i+j)\%2=0,k<\frac{C}{2}}$ and $\ddot{y}_{t,(i+j)\%2=1,k\geq\frac{C}{2}}$, but also generates the quantization step at spatial-channel-wise. During the second-step coding, the coded latent representations in previous step are fused to the input, and then the entropy model estimates the the mean and scale values of the probability distribution for $\ddot{y}_{t,(i+j)\%2=1,k<\frac{C}{2}}$ and $\ddot{y}_{t,(i+j)\%2=0,k\geq\frac{C}{2}}$.

**Motion vector encoder and decoder.** The encoder and decoder for motion vector are illustrated in Fig. 11. For encoder, the input is the original motion vector with channel 2. The output is the latent representation at 16x downsampled resolution with channel 64. The decoder follows the inverse structure. The multi-granularity quantization and entropy model for $mv\_y_t$ follow those of $y_t$. The only difference is the entropy model input. For motion vector, the inputs of entropy model are the corresponding hyper prior and the latent prior. In Fig. 11, the downsample and upsample residual blocks are from [41].

**Frame generator.** Different from DCVC [27] and [41] only using plain residual blocks, we proposing using the W-Net [47] based structure to build our frame generator. The W-Net can effectively enlarge the receptive field of model to improve the generation ability of model. The detailed network structure of our frame generator is presented in Fig. 14. The inputs include the the high-resolution feature $\hat{F}_t$ with channel 32 and the context $\bar{C}_t^0$ at original resolution with channel 64. The outputs are the final reconstructed frame and the feature used by the next frame. In Fig. 14, there are two same U-Nets to build the W-Net. The residual block with attention in the U-Net is illustrated in Fig.12.

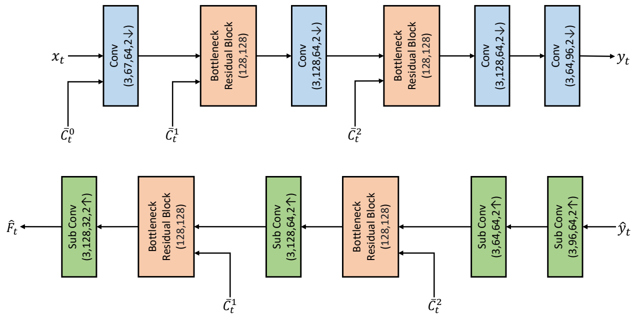


**Figure 7: Structure of contextual encoder and decoder.**

**Neural image codec.** As we target at single model handling multiple rates, we also need a neural image codec supporting such capability for intra coding. The structure of our neural image codec is presented in Fig. 13. The corresponding multi-granularity quantization and the entropy model are similar with those of $y_t$. The only difference is the input of entropy model. For neural image codec, the input of entropy model only includes the hyper prior. In Fig. 13, we also use one U-Net to improve the generation ability of neural image codec and its network structure is similar to that in Fig. 14. The compression ratio of our image codec is on par with that of Cheng 2020 [12], as shown in Table 8.

## B SETTINGS OF TRADITIONAL CODECS

For x265 [5], we use the *veryslow* preset. We also compare HM-16.20 [1] and VTM-13.2 [4], which represent the best encoder of H.265 and H.266, respectively. For HM and VTM, the low delay configuration with the highest compression ratio is used. 4 reference frames and 10-bit internal bit depth are used (settings in *encoder_lowdelay_main_rext.cfg* for HM and *encoder_lowdelay_vtm.cfg* for VTM). The comparison in [41] has shown that, although the final distortion is measured in RGB domain, using YUV 444 as the internal color space could boost the compression ratio. Thus, we also use this setting. More studies on codec setting can be found in [41]. The detailed settings of x265, HM, and VTM are:

- **x265**
  ffmpeg
  -pix_fmt yuv420p
  -s {*width*}x{*height*}
  -framerate {*frame rate*}
  -i {*input file name*}
  -vframes {*frame number*}
  -c:v libx265
  -preset veryslow
  -tune zerolatency
  -x265-params
  “qp={*qp*}:keyint=32:csv-log-level=1:
  csv={*csv_path*}:verbose=1:psnr=1"
  {*output video file name*}

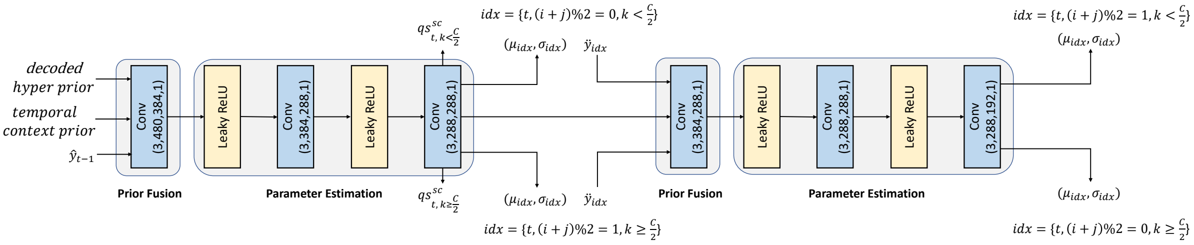

decoded hyper prior
temporal context prior
$\hat{y}_{t-1}$
Conv (3,480,384,1)
Prior Fusion
Leaky ReLU
Conv (3,384,288,1)
Leaky ReLU
Conv (3,288,288,1)
Parameter Estimation
$qs^{sc}_{t,k<\frac{C}{2}}$
$qs^{sc}_{t,k\geq\frac{C}{2}}$
$idx = \{t, (i+j)\%2 = 0, k < \frac{C}{2}\}$
$(\mu_{idx}, \sigma_{idx})$
$\ddot{y}_{idx}$
$(\mu_{idx}, \sigma_{idx})$
$\ddot{y}_{idx}$
$idx = \{t, (i+j)\%2 = 1, k \geq \frac{C}{2}\}$
Conv (3,384,288,1)
Prior Fusion
Leaky ReLU
Conv (3,288,288,1)
Leaky ReLU
Conv (3,288,192,1)
Parameter Estimation
$idx = \{t, (i+j)\%2 = 1, k < \frac{C}{2}\}$
$(\mu_{idx}, \sigma_{idx})$
$(\mu_{idx}, \sigma_{idx})$
$idx = \{t, (i+j)\%2 = 0, k \geq \frac{C}{2}\}$

**Figure 8: Structure of entropy model.**

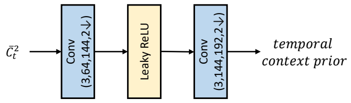


**Figure 9: Structure of temporal context encoder.**

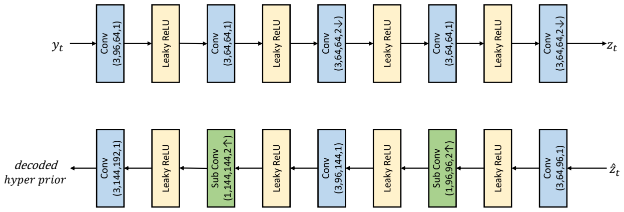


**Figure 10: Structure of hyper prior encoder and decoder.**

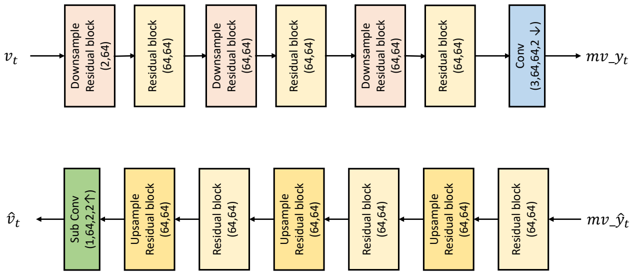


**Figure 11: Structure of motion vector encoder and decoder.**

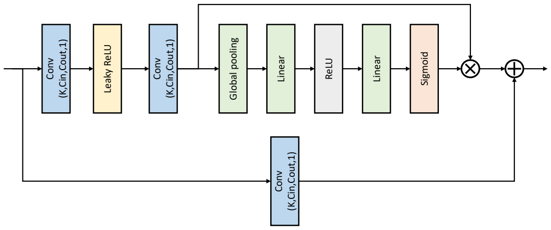


**Figure 12: Structure of residual block with attention.**

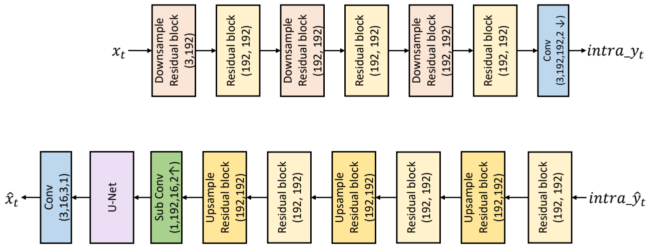

$x_t$
Downsample Residual block (3,192)
Residual block (192, 192)
Downsample Residual block (192, 192)
Residual block (192, 192)
Downsample Residual block (192, 192)
Residual block (192, 192)
Conv (3,192,192,2 ↓)
$intra\_y_t$
$\hat{x}_t$
Conv (3,16,3,1)
U-Net
Sub Conv (1,192,16,2↑)
Upsample Residual block (192,192)
Residual block (192, 192)
Upsample Residual block (192,192)
Residual block (192, 192)
Upsample Residual block (192,192)
Residual block (192, 192)
$intra\_\hat{y}_t$

**Figure 13: Structure of neural image codec.**

- **HM**<br>
  TAppEncoder<br>
  -c encoder_lowdelay_main_rext.cfg<br>
  --InputFile={*input file name*}<br>
  --InputBitDepth=8<br>
  --OutputBitDepth=8<br>
  --OutputBitDepthC=8<br>
  --InputChromaFormat=444<br>
  --FrameRate={*frame rate*}<br>
  --DecodingRefreshType=2<br>
  --FramesToBeEncoded={*frame number*}<br>
  --SourceWidth={*width*}<br>
  --SourceHeight={*height*}<br>
  --IntraPeriod=32<br>
  --QP={*qp*}<br>
  --Level=6.2<br>
  --BitstreamFile={*bitstream file name*}
- **VTM**<br>
  EncoderApp<br>
  -c encoder_lowdelay_vtm.cfg<br>
  --InputFile={*input file name*}<br>
  --BitstreamFile={*bitstream file name*}<br>
  --DecodingRefreshType=2<br>
  --InputBitDepth=8<br>
  --OutputBitDepth=8<br>
  --OutputBitDepthC=8<br>
  --InputChromaFormat=444<br>
  --FrameRate={*frame rate*}<br>
  --FramesToBeEncoded={*frame number*}<br>
  --SourceWidth={*width*}<br>
  --SourceHeight={*height*}

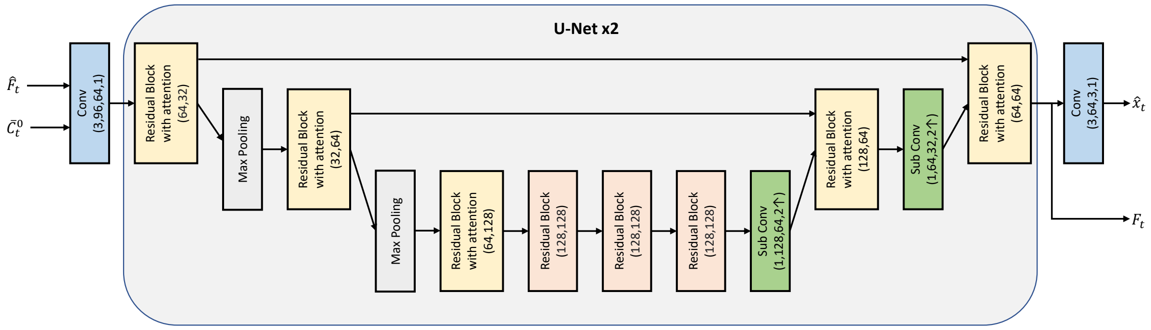


**Figure 14: Structure of frame generator.**

```
--IntraPeriod=32
--QP={qp}
--Level=6.2
```

## C RATE-DISTORTION CURVES

Fig. 15 and Fig. 16 show the RD (rate-distortion) curves on each dataset, including both the PSNR and MS-SSIM results. From these results, we can find that our codec can achieve SOTA compression ratio in wide rate range.

## D COMPARISON UNDER DIFFERENT INTRA PERIOD SETTINGS

Previous work Sheng 2021 [41] shows that intra period 12 is harmful to the compression ratio and is seldom used in the real applications. For example, when compared with intra period 12, intra period 32 has an average of 23.8% bitrate saving for HM (Table 3 in the supplementary material of [41]). Thus, to get closer to the practical scenario, we follow [39] and also use intra period 32 in our paper.

But it is also important to evaluate our codec under intra period 12. Table 9 shows the corresponding BD-rate (%) comparison in terms of PSNR. It is noted that, since the GOP size of VTM with *encoder_lowdelay_vtm* configuration is 8, it does not support intra period 12. Therefore, we configure VTM-13.2* by following [41] and use it as the anchor (more configuration details of VTM-13.2* can be found in [41]). We can see that, under intra period 12, our codec also significantly outperforms all previous SOTA neural and traditional codecs.

## E VISUAL COMPARISONS

We also conduct the visual comparisons with the previous SOTA neural and traditional video codecs. Two examples are shown in Fig. 17 and Fig. 18. From these comparisons, we can find that our codec can achieve much higher reconstruction quality without increasing the bitrate cost. For instance, in the example shown in Fig. 17, we can find that the frames reconstructed by previous neural codecs have obvious color distortion. By contrast, our codec can restore more accurate stripe color. In Fig. 18, our codec can generate clearer texture details.

## F OTHER DETAILS

Our codec is built upon previous developments DCVC [27] and its improved work Sheng 2021 [41]. When compared with Sheng 2021 [41], the motion estimation, motion vector encoder/decoder, temporal context mining, and contextual encoder/decoder are reused. The entropy model, quantization mechanism, and frame generator are redesigned. In addition, we add more details on our quantizer and MEMC (motion estimation and motion compensation) process, respectively.

**Quantizer**. The quantizer works as follows:

- First, the latent representation is divided by the quantization step (QS).
- Second, the mean value is subtracted.
- Third, the latent representation is rounded to the nearest integer and converted into bit-stream via the arithmetic encoder.

During the decoding, the rounded latent representation is first decoded by the arithmetic decoder and adds back the mean value. At last, the QS is multiplied. In this process, most existing image and video codecs use the fixed QS, i.e., 1. By contrast, we decide the QS at multi-granularity levels. First, the global QS is set by the user for the specific target rate. Then it is scaled by the channel-wise QS because different channels contain information with different importance. At last, the spatial-channel-wise QS generated by our entropy model is applied. This can help our codec cope with various video contents and achieve precise rate adjustment at each position.

**Motion estimation and motion compensation**. The MEMC process can be divided into the following steps:

- First, the original motion vector (MV) between the current frame and the reference frame is estimated via the optical flow estimation network. The MV is dense at full resolution.
- The original MV is then transformed into latent representation via the MV encoder.
- The latent representation of MV is quantized and converted into bit-stream. The quantization step therein is also determined at multi-granularity levels. The quantization and entropy model are similar with those of the latent representation of the current frame.
- The MV at full resolution is then reconstructed via arithmetic decoder and MV decoder.

**Table 8: Image codec BD-Rate (%) comparison for PSNR.**

| | UVG | MCL-JCV | HEVC B | HEVC C | HEVC D | HEVC E | HEVC RGB | Average |
|---|---|---|---|---|---|---|---|---|
| Cheng 2020 image codec [12] | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Our image codec | -0.9 | 0.2 | -1.9 | -0.2 | 2.5 | 1.9 | -1.2 | 0.1 |

**Table 9: BD-Rate (%) comparison for PSNR under intra period 12.**

| | UVG | MCL-JCV | HEVC B | HEVC C | HEVC D | HEVC E | HEVC RGB | Average |
|---|---|---|---|---|---|---|---|---|
| VTM-13.2* | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| HM-16.20 | 8.3 | 15.3 | 20.1 | 13.3 | 11.6 | 22.4 | 13.2 | 14.9 |
| x265 | 97.4 | 99.2 | 89.1 | 49.5 | 44.2 | 79.5 | 94.5 | 79.1 |
| DVCPro [34] | 54.8 | 63.0 | 67.3 | 70.5 | 47.2 | 124.2 | 50.8 | 68.3 |
| RLVC [49] | 74.0 | 103.8 | 87.2 | 86.1 | 53.0 | 98.7 | 66.2 | 81.3 |
| MLVC [28] | 39.9 | 65.0 | 57.2 | 99.5 | 75.8 | 84.1 | 64.3 | 69.4 |
| DCVC [27] | 21.0 | 28.1 | 32.5 | 44.8 | 25.2 | 66.8 | 22.9 | 34.5 |
| Sheng 2021 [41] | -20.4 | -1.6 | -1.0 | 15.7 | -3.4 | 7.8 | -13.7 | -2.4 |
| Ours | -38.4 | -24.2 | -19.9 | -10.5 | -23.9 | -18.7 | -34.2 | -24.3 |

- The reconstructed MV is used to warp the feature and extract the context via the temporal context mining module.

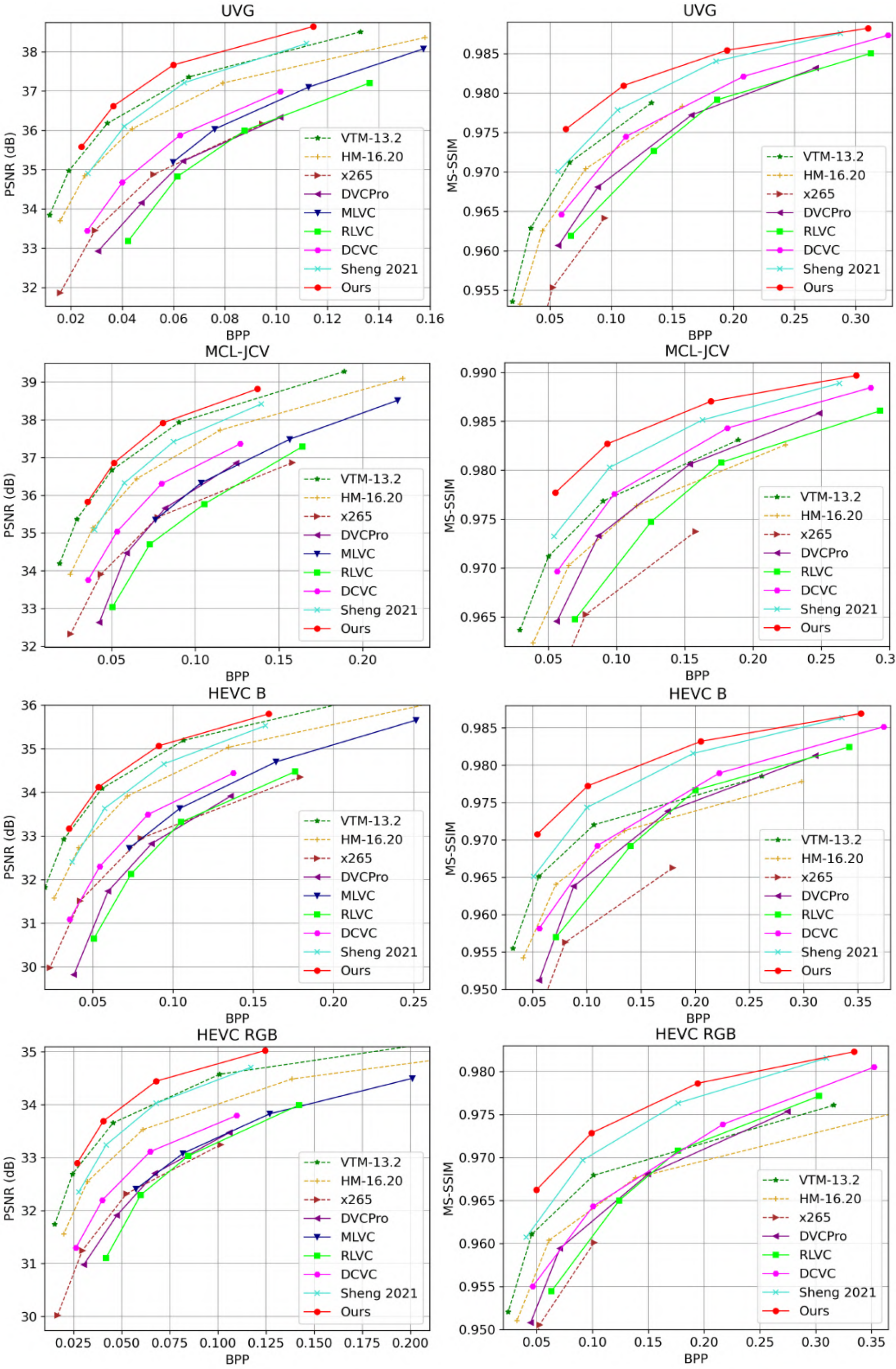


**Figure 15: RD curves of UVG, MCL-JCV, HEVC B and RGB.**

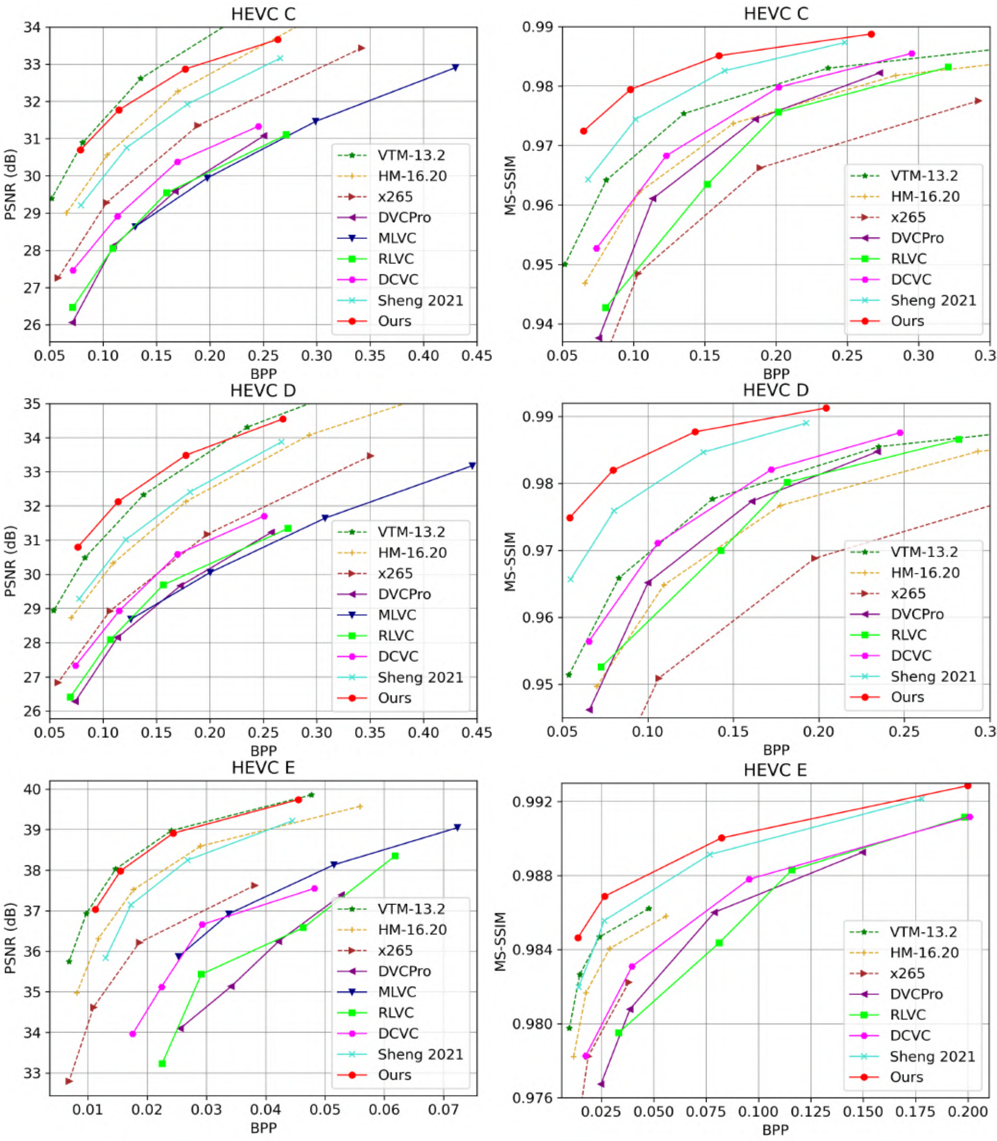
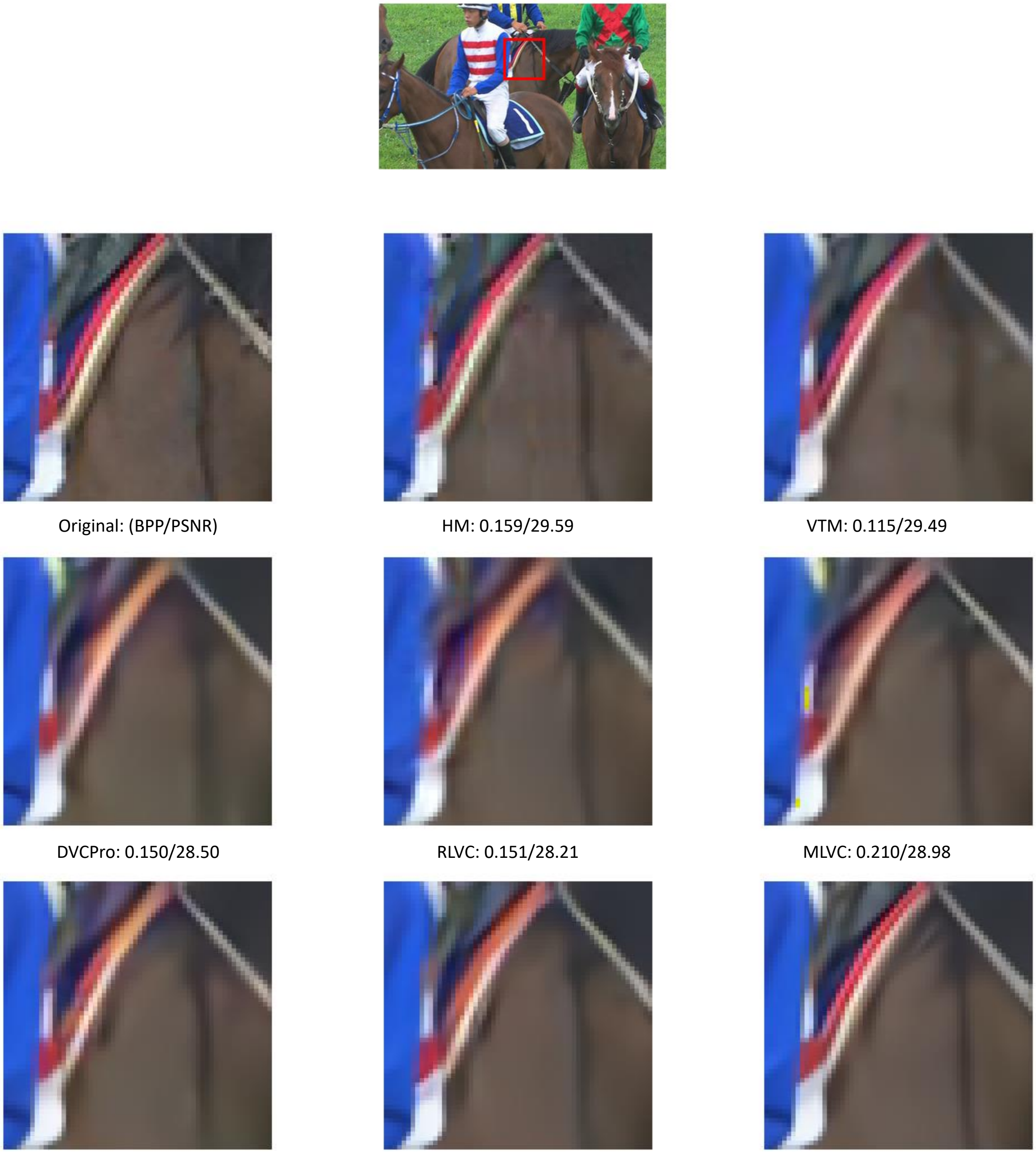


**Figure 16: RD curves of HEVC C, D and E.**



Original: (BPP/PSNR)

HM: 0.159/29.59

VTM: 0.115/29.49

DVCPro: 0.150/28.50

RLVC: 0.151/28.21

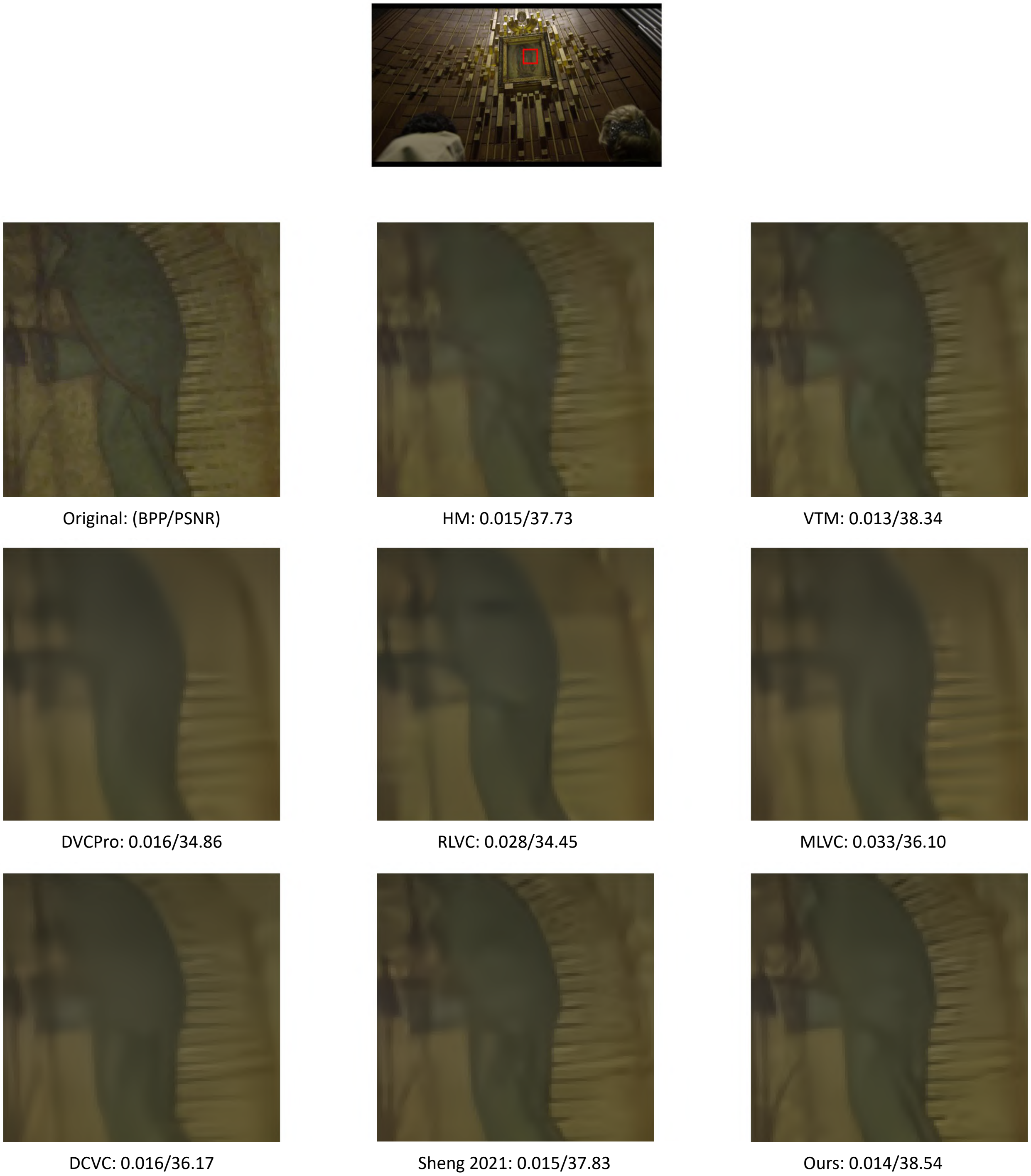
MLVC: 0.210/28.98

DCVC: 0.154/29.44

Sheng 2021: 0.115/29.57

Ours: 0.112/30.54

**Figure 17: *RaceHorses* from HEVC D dataset.**



Original: (BPP/PSNR)

HM: 0.015/37.73

VTM: 0.013/38.34

DVCPro: 0.016/34.86

RLVC: 0.028/34.45

MLVC: 0.033/36.10

DCVC: 0.016/36.17

Sheng 2021: 0.015/37.83

Ours: 0.014/38.54

**Figure 18: *videoSRC03* from MCL-JCV dataset.**