Supplementary Material of
Motion Information Propagation for Neural Video Compression

Linfeng Qi[1*] Jiahao Li [2] Bin Li[2] Houqiang Li[1] Yan Lu[2]
[1] University of Science and Technology of China [2] Microsoft Research Asia
qlf324@mail.ustc.edu.cn, lihq@ustc.edu.cn, {li.jiahao, libin, yanlu}@microsoft.com

1. Overview

This document provides the supplementary material for our paper of Motion Information Propagation for Neural Video Compression. First, we provide the information about model complexity in Section 2. Then we illustrate some network architectures in Section 3. The settings of traditional codecs can be found in Section 4. Finally, some additional results are given in Section 5, including the performance comparison with intra period 12, the visual comparisons, and more analyses on bit allocation.

2. Model Complexity

Table 1. Complexity comparison.

Method	MACs	Encoding time	Decoding time
DCVC-HEM [7]	3269G	766ms	530ms
Our DCVC-MIP	3386G	826ms	593ms

Table 1 provides the model complexity about the MACs (multiply–accumulate operations), encoding time, and decoding time. We use 1080p video frames as input for measurement. The encoding time and decoding time are tested on NVIDIA V100 GPU. The numbers include the time of bitstream writing/reading. Comparing with DCVC-HEM [7], our DCVC-MIP needs a little additional computation cost (3.6% extra MACs). We also need a little additional encoding/decoding time. The increase of complexity is due to the cost of offset diversity [4] and transformer based context refinement. We think that there is still much room for improving the model complexity, *e.g.,* using efficient attention [12] instead of the original self-attention.

3. Network Architecture

Transformer based context refinement. As shown in Figure 1 (a), the transformer based context refinement module consists of two convolution layers and a residual Swin Transformer block (RSTB) [8]. The RSTB contains two swin transformer layer (STL), where the window based multi-head self-attention could effectively enlarge the receptive field with acceptable complexity. The window size is set to 4. It does not bring too much computation cost because the computation is done at low resolution and the number of channels is only 64.

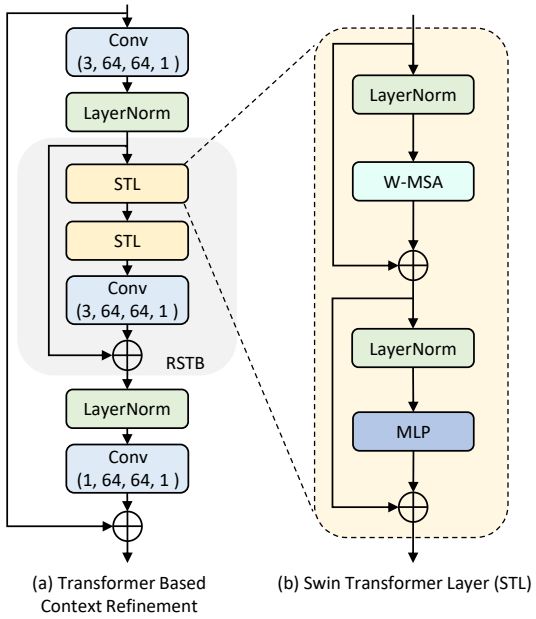

Figure 1. Illustration for transformer based context refinement.

Entropy model of motion coding. As mentioned in the manuscript, we introduce the propagated motion condition H_{t-1} to the entropy model of motion coding. We use a simple prior encoder to process H_{t-1} to get an extra prior, whose structure is shown in Figure 2. We denote this extra prior as motion condition prior. We use the same entropy model of motion coding as that in DCVC-HEM [7]. However, as shown in Figure 3, besides decoded hyper prior and previously decoded motion latent $mv\_\hat{y}_{t-1}$, we also use the motion condition prior as input. These three priors will help

*This work was done when Linfeng Qi was an intern at Microsoft Research Asia.

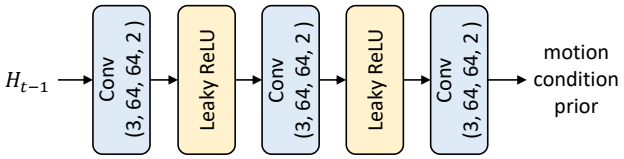

Figure 2. Structure of the motion condition prior encoder.

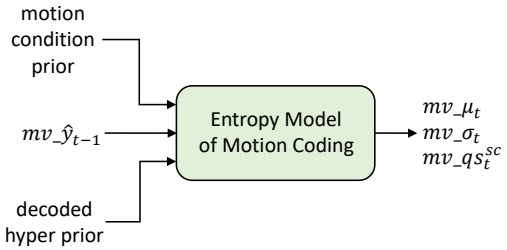

Figure 3. Illustration for the entropy model of motion coding.

the entropy model to estimate the quantization step, mean and scale values for the motion latent representations.

Motion encoder and motion decoder. Figure 4 illustrates the structure of motion encoder and motion decoder. The motion encoder takes the estimated motion v_t and the motion condition H_{t-1} as input, and will output the latent representation mv_y_t. The decoder will take the quantized latent representation $mv\_\hat{y}_t$ as input, and output the decoded motion $\hat{v}_t$. The feature G_t, which is of 1/2 resolution in the motion decoder, will serve as motion guidance to assist to mitigate the alignment errors in context generation.

4. Settings of Traditional Codecs

We follow DCVC-HEM [7] to configure the traditional codecs. We use the very flow preset for x265 [3]. As for HM [1] and VTM [2], we use the low delay configuration with the highest compression ratio. 4 reference frames is used. The settings are as follows:

- x265
  ffmpeg
  -pix_fmt yuv420p
  -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*}

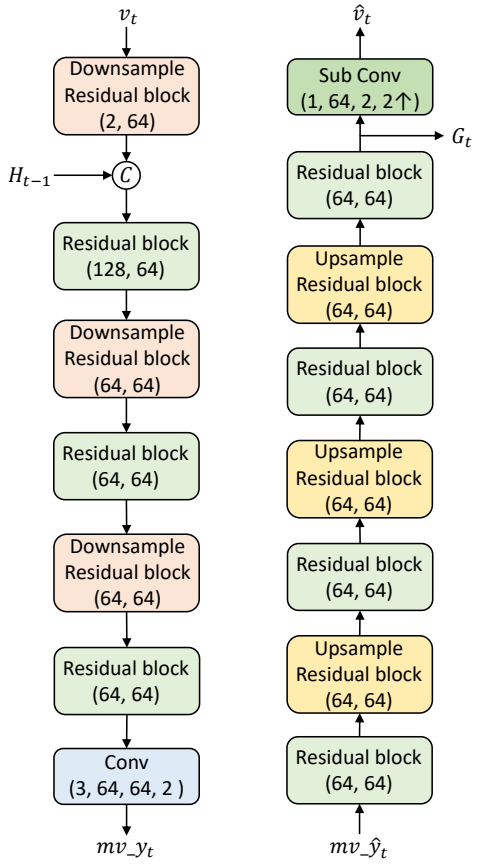

Figure 4. Structure of the motion encoder and motion decoder.

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

- VTM
  EncoderApp
  -c encoder_lowdelay_vtm.cfg
  --InputFile={*input file name*}
  --BitstreamFile={*bitstream file name*}

Table 2. BD-Rate (%) results for PSNR in comparison with VTM-13.2 with intra period 12

Method	UVG	MCL-JCV	HEVC B	HEVC C	HEVC D	HEVC E	HEVC RGB	Average
VTM-13.2 [2]	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
HM-16.20 [1]	8.3	15.3	20.1	13.3	11.6	22.4	13.2	14.9
x265 [3]	97.4	99.2	89.1	49.5	44.2	79.5	94.5	79.1
DVCPro [10]	54.8	63.0	67.3	70.5	47.2	124.2	50.8	68.3
M-LVC [9]	39.9	65.0	57.2	99.5	75.8	84.1	64.3	69.4
CANF-VC [5]	-3.9	10.5	12.7	15.3	7.8	19.0	14.6	10.9
DCVC [6]	21.0	28.1	32.5	44.8	25.2	66.8	22.9	34.5
DCVC-TCM [11]	-20.4	-1.6	-1.0	15.7	-3.4	7.8	-13.7	-2.4
DCVC-HEM [7]	-38.4	-24.2	-19.9	-10.5	-23.9	-18.7	-34.2	-24.3
Our DCVC-MIP	**-43.8**	**-29.8**	**-28.1**	**-20.1**	**-32.4**	**-28.3**	**-39.5**	**-31.7**

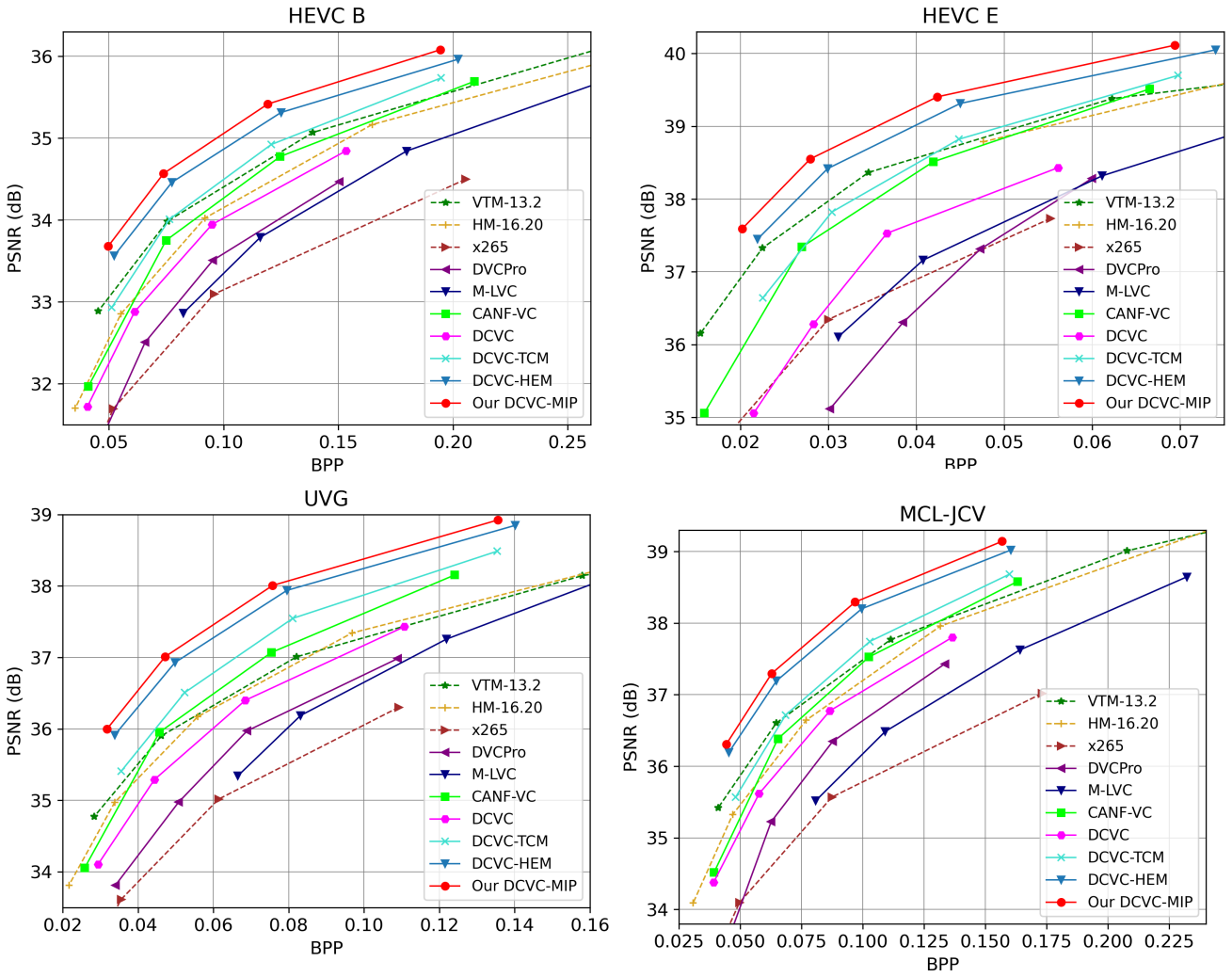

Figure 5. RD-Curves with intra perid 12.

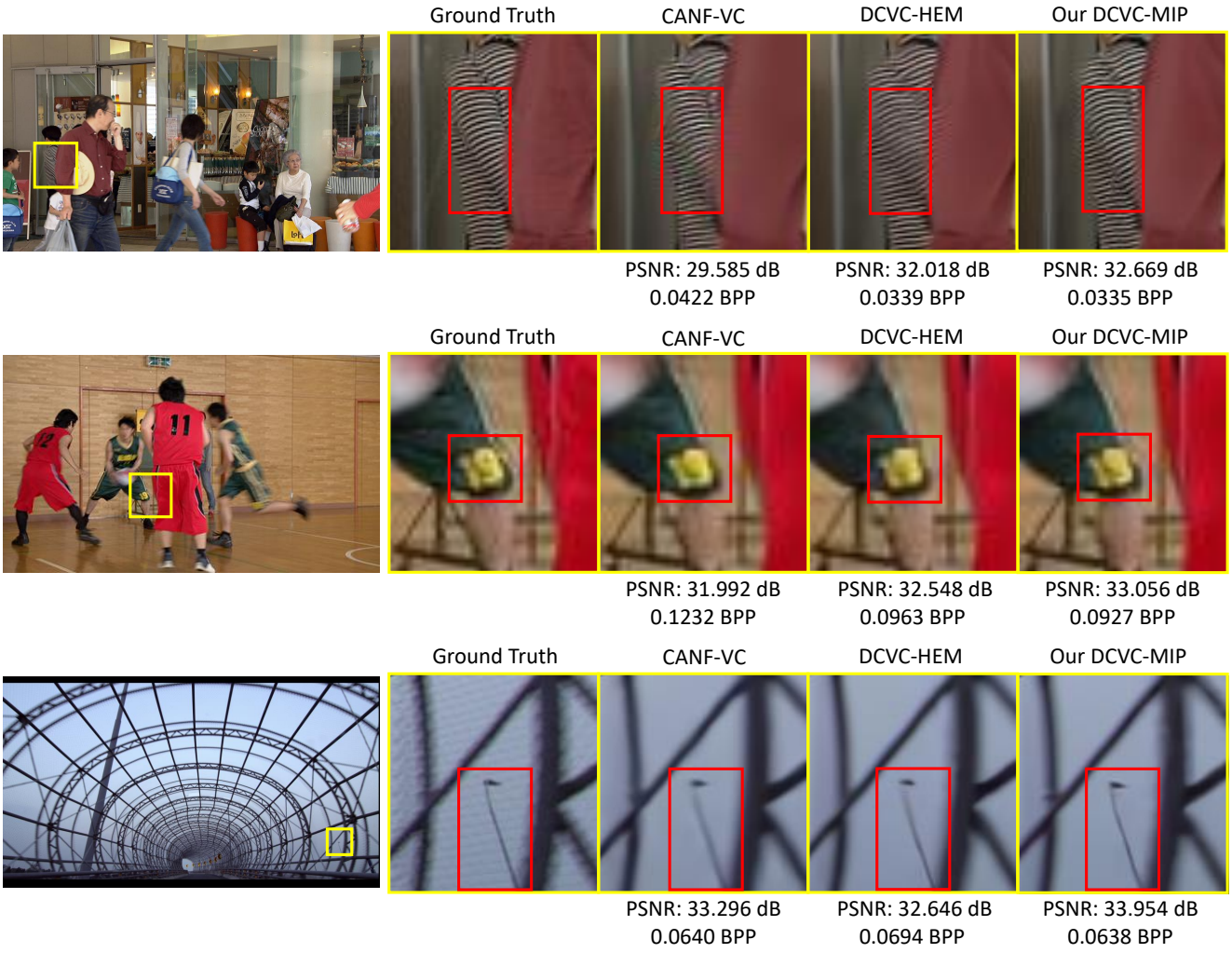

Figure 6. Visual Comparisons.

--DecodingRefreshType=2
--InputBitDepth=8
--OutputBitDepth=8
--OutputBitDepthC=8
--InputChromaFormat=444
--FrameRate={*frame rate*}
--FramesToBeEncoded={*frame number*}
--SourceWidth={*width*}
--SourceHeight={*height*}

5. Additional Results

Performance comparison with intra period 12. In the manuscript, we have provided the performance comparison with intra period 32. We also give comparison results with intra period 12 here, which is shown in Table 2. We follow DCVC-HEM [7] to configure VTM-13.2, which serves as the anchor. It is shown that our method can achieve an average of 31.7% bit rate saving over VTM-13.2. Our model still outperforms the previous SOTA neural codecs. Figure 5 illustrates the RD curves when the intra period is set to 12. It is shown that under the same PSNR, our method will consume less bitrate. These experimental results demonstrate the effectiveness of the proposed method.

Visual comparisons. In Figure 6, we provide some visual comparisons with the recent methods CANF-VC [5] and DCVC-HEM [7]. It is shown that our DCVC-MIP can preserve more structure details and achieve higher reconstruction quality. For example, in the first row, we can see that while other neural codecs suffer from losing texture details and color distortion, our DCVC-MIP can still output a high-quality result. In the other two examples, our DCVC-MIP can also reconstruct the frame details better without increasing the bit cost.

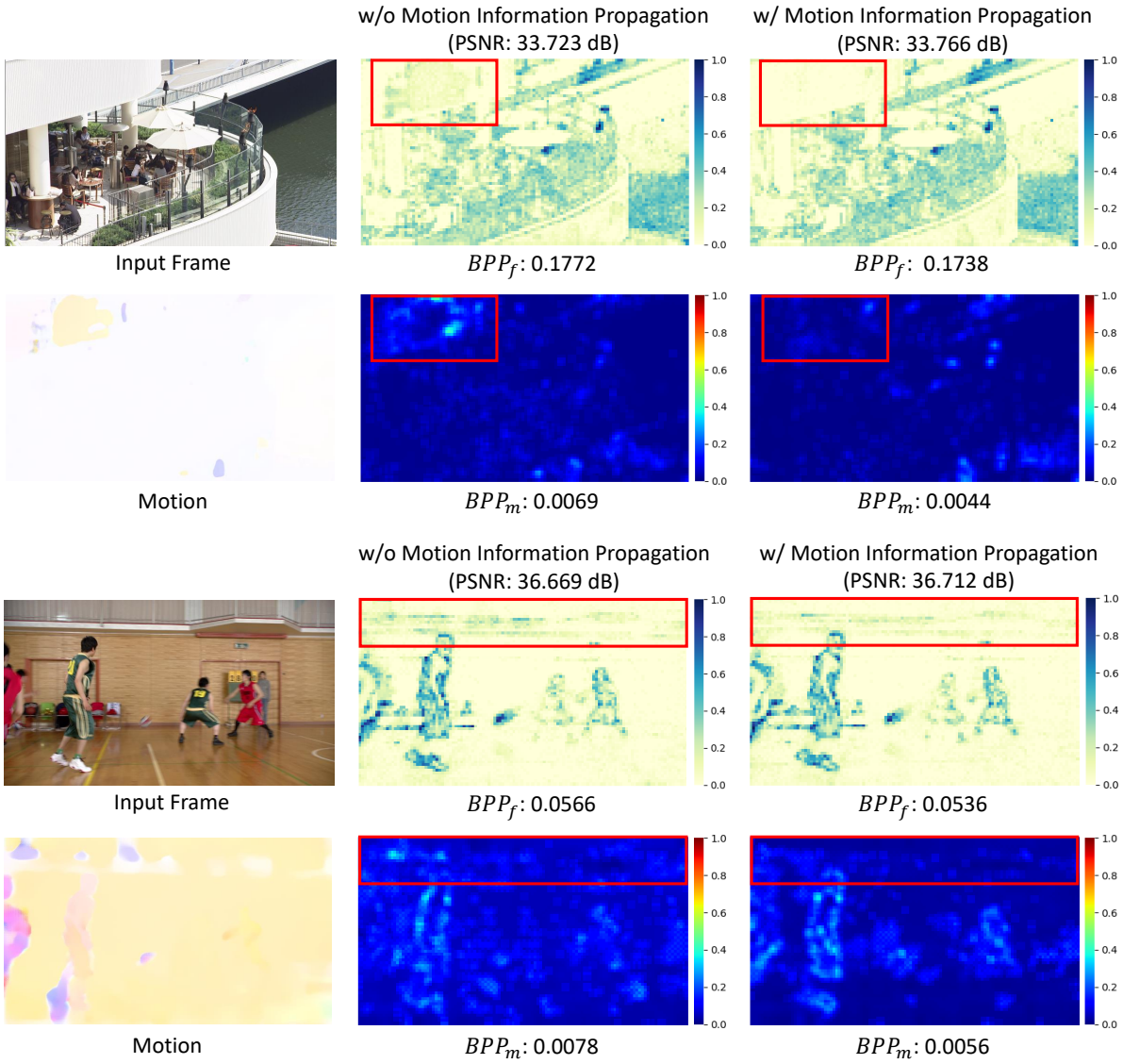

Figure 7. Visualization for the pixel-level bit allocation of frame coding and motion coding. For each example, the bit allocation of frame coding is in the top row and the bit allocation of motion coding is in the bottom row. When using Motion Information Propagation, frame coding and motion coding share similar regions where the bit cost is reduced, which could show that the bi-directional information interactions assist to achieve the synergy between frame coding and motion coding.

Visualization for bit allocation. We also visualize the pixel-level bit allocation of frame coding and motion coding in Figure 7. We compare the bit allocation of frame coding and motion coding when using model with or without the proposed Motion Information Propagation. The bit allocation of frame coding is depicted in the top row, where BPP_f means bits per pixel for frame coding (coding latent representation y_t and hyperprior representation z_t). The bit allocation of motion coding is depicted in the bottom row, where BPP_m means bits per pixel for motion coding (coding motion latent representation mv_y_t and motion hyperprior representation mv_z_t). In the first example, the incorrect estimated motion leads to an unnecessary bit increase, which is annotated by the red box in the second column. When employing Motion Information Propagation, we can see that the bit cost is reduced obviously for both motion coding and frame coding in the annotated region. In the second example, it is also shown that Motion Information Propagation could help to save bit cost. We can see that the regions with obvious bit cost reduction of frame coding and motion coding are similar. It shows that through the proposed Motion Information propagation, the bi-directional information interactions can assist to achieve the synergy between frame coding and motion coding.

References

[1] HM-16.20. https://vcgit.hhi.fraunhofer.de/jvet/HM/-/tree/HM-16.20. Accessed: 2022-09-01. 2, 3

[2] VTM-13.2. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tree/VTM-13.2. Accessed: 2022-09-01. 2, 3

[3] x265. https://www.videolan.org/developers/x265.html. Accessed: 2022-09-01. 2, 3

[4] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Understanding deformable alignment in video super-resolution. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 973–981, 2021. 1

[5] Yung-Han Ho, Chih-Peng Chang, Peng-Yu Chen, Alessandro Gnutti, and Wen-Hsiao Peng. Canf-vc: Conditional augmented normalizing flows for video compression. *arXiv preprint arXiv:2207.05315*, 2022. 3, 4

[6] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[7] 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*, 2022. 1, 2, 3, 4

[8] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021. 1

[9] 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

[10] 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. 3

[11] 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

[12] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 1