Supplementary Material for
M-LVC: Multiple Frames Prediction for Learned Video Compression

Jianping Lin Dong Liu Houqiang Li Feng Wu
CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System,
University of Science and Technology of China, Hefei 230027, China
ljp105@mail.ustc.edu.cn, {dongeliu,lihq,fengwu}@ustc.edu.cn

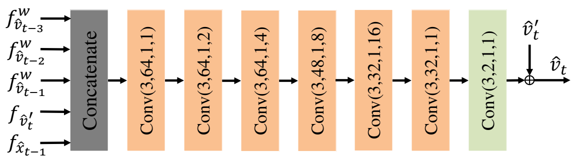


Figure 1. The MV refinement network. Conv(3,64,1,1) represents the convolutional layer with the kernel size of 3×3, the output channel of 64, the stride of 1, and the dilation constant of 1. Each convolutional layer is followed by a leaky ReLU except the last layer (indicated by green).

1. Proposed Method

1.1. Details of Our MV Refinement Network

The architecture of our MV refinement network is presented in Fig. 1. We first use a two-layer CNN to extract the features of $\hat{v}_{t-3}$, $\hat{v}_{t-2}$, $\hat{v}_{t-1}$, $\hat{v}'_t$, and $\hat{x}_{t-1}$, respectively. And then, the features of $\hat{v}_{t-3}$, $\hat{v}_{t-2}$ and $\hat{v}_{t-1}$ are warped towards v_t with the help of $\hat{v}'_t$,

$$\begin{aligned} \hat{v}^w_{t-k} &= Warp(\hat{v}_{t-k}, \hat{v}'_t + \sum_{l=1}^{k-1} \hat{v}^w_{t-l}), k = 1, 2 \\ f^w_{\hat{v}_{t-i}} &= Warp(f_{\hat{v}_{t-i}}, \hat{v}'_t + \sum_{k=1}^{i-1} \hat{v}^w_{t-k}), i = 1, 2, 3 \end{aligned} \tag{1}$$

where $\hat{v}^w_{t-k}$ is the warped version of $\hat{v}_{t-k}$ towards $\hat{v}'_t$. Finally, the warped features, and the features of $\hat{v}'_t$ and $\hat{x}_{t-1}$ are fed into a dilated convolution-based network, which can capture larger receptive field, to obtain the final reconstructed MV,

$$\hat{v}_t = H_{mvr}(f^w_{\hat{v}_{t-3}}, f^w_{\hat{v}_{t-2}}, f^w_{\hat{v}_{t-1}}, f_{\hat{v}'_t}, f_{\hat{x}_{t-1}}) + \hat{v}'_t \tag{2}$$

where H_{mvr} denotes the function of the network.

1.2. Details of Our Residual Refinement Network

Fig. 2 shows the architecture of our residual refinement network. First, we use a two-layer CNN to extract the features of $\hat{x}_{t-4}$, $\hat{x}_{t-3}$, $\hat{x}_{t-2}$, and $\hat{x}_{t-1}$ and warp them towards the current frame. This warping operation is the same with Eq. (4) in the paper. Then, the warped features and the features of $\bar{x}_t$ and $\hat{r}'_t$ are fed into a CNN, which is based on the U-Net structure [8] and integrates multiple residual blocks, to obtain the refined residual $\hat{r}_t$,

$$\hat{r}_t = H_{res}(f^w_{\hat{x}_{t-4}}, f^w_{\hat{x}_{t-3}}, f^w_{\hat{x}_{t-2}}, f^w_{\hat{x}_{t-1}}, f_{\bar{x}_t}, f_{\hat{r}'_t}) \tag{3}$$

where H_{res} represents the function of the network.

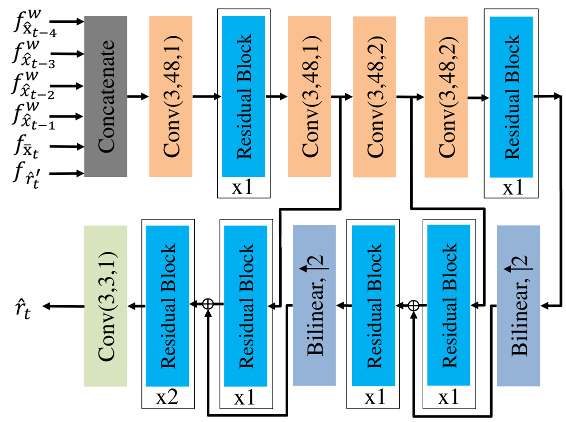


Figure 2. The residual refinement network. Each convolutional layer outside residual blocks is followed by a leaky ReLU except the last layer (indicated by green). Each residual block consists of two convolutional layers, which are configured as follows: kernel size is 3×3, output channel number is 48, the first layer has ReLU.

2. Experiments

2.1. Ablation Study of Our MAMVP-Net

To verify the effectiveness of the components in MAMVP-Net, we conduct experiments to compare the proposed MAMVP-Net (denoted by multi-scale w/ alignment) with its simplified versions: (1) single-scale w/o alignment, (2) single-scale w/ alignment, (3) multi-scale w/o align-

Table 1. Bit-rates (bpp) and reconstruction quality (PSNR) for ablation study of the MAMVP-Net

Network	single-scale w/o alignment	single-scale w/ alignment	multi-scale w/o alignment	multi-scale w/ alignment
bpp	0.297	0.290	0.287	0.285
PSNR (dB)	31.250	31.198	31.196	31.290

Figure 3. Visualized results of compressing the Kimono sequence using `Add MVRefine-Net` model. From left to right: the original MVD d_6, the decoded MVD $\hat{d}_6$, and the refined MVD, *i.e.* $\hat{v}_6 - \bar{v}_6$.

ment. These models are tested on HEVC Class D dataset and the reconstruction quality and bit-rates are shown in Table 1. It can be observed that the proposed MAMVP-Net achieves the highest reconstruction quality with the lowest bit-rates.

2.2. Visual Results of Our MV Refine-Net

In Fig. 3, we visualize the original MVD d_6, the decoded MVD $\hat{d}_6$, and the MVD after refinement, *i.e.* $\hat{v}_6 - \bar{v}_6$, when compressing the Kimono sequence using `Add MVRefine-Net` model. After compression, there are more zeros in $\hat{d}_6$ than d_6 due to the bit rate constraint. Our MV Refine-Net can restore some non-zero MVDs and thus improve the accuracy.

2.3. Visual Results of Our MMC-Net

In Fig. 4, we visualize the original frame x_9 (a), the predicted frame $\bar{x}_9$ obtained by `Add MVRefine-Net` model with $\lambda = 64$ (b), and the predicted frame $\bar{x}_9$ obtained by `Add MMC-Net` model with $\lambda = 64$ (c), when compressing the BasketballPass sequence. We can observe that the image in Fig. 4 (b) is much more smooth than (c), *e.g.* in the area of the wall. Quantitatively, the PSNR of the predicted frame in Fig. 4 (c) is 31.97dB, while the PSNR of the predicted frame in Fig. 4 (b) is 31.42dB. Therefore, our MMC-Net can obtain a more accurate prediction with more details by using multiple reference frames.

2.4. Visual Results of Our Residual Refine-Net

In Fig. 5, we visualize the original residual r_6, the decoded residual $\hat{r}'_6$, and the refined residual $\hat{r}_6$, when compressing the RaceHorses sequence using `Proposed` model. We can observe that $\hat{r}'_6$ is much more smooth than r_6 due to the rate constraint. Our Residual Refine-Net can restore some image details and thus improve the reconstruction quality.

2.5. Compression Performance on the HEVC Class C and E Datasets

We provide the compression results on the HEVC Class B and D datasets in the paper. In Fig. 8, we also present the compression results on the HEVC Class C and E datasets using H.264, H.265, DVC [5], and the proposed method. It can be observed that our method outperforms DVC [5] by a large margin. When compared with H.265, our method achieves on par or better compression performance in PSNR and MS-SSIM.

2.6. Comparison with Other Learned Video Compression Methods

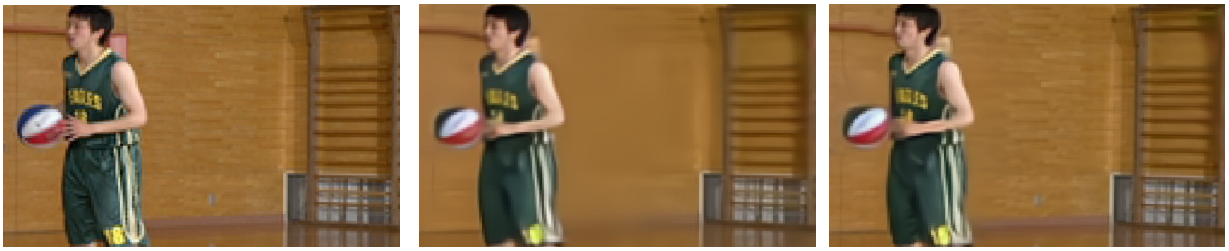
In the paper, we compare with two learned video compression methods of the state-of-the-art, *i.e.* Wu_ECCV2018 [10] and DVC [5]. Here, we also compare with other two latest learned methods, i.e. Djelouah_ICCV2019 [4] designed for random-access scenarios and Rippel_ICCV2019 [7] targeting low-latency scenarios. From Fig. 7 (b), we can observe that Djelouah_ICCV2019 [4] achieves better performance of $0.25 \sim 0.7$dB gain than our method in terms of PSNR on the MCL_JCV dataset [9]. Note that, their method is designed for random-access scenarios and integrates the autoregressive prior, proposed in [6], to predict the probabilities of quantized representations in entropy model. This autoregressive model has an obvious disadvantage of high decoding complexity even in parallel devices like GPU/TPU. From Fig. 7 (c), we can observe that Rippel_ICCV2019 [7] outperforms our method by about 0.005 in terms of MS-SSIM on the Xiph 1080p video dataset [1]. Note that, their method is optimized directly for MS-SSIM, but ours is optimized for MSE. It requires our future work to optimize our model for MS-SSIM to achieve a better performance in MS-SSIM.

2.7. Comparison with H.264 and H.265 in Other Settings

In the paper, we compare with the results of H.264 and H.265 where the results are directly cited from [5]. Note that the results are obtained by using the `veryfast` mode of x264 and x265 codecs, respectively. Here, we also compare with the results of H.264 and H.265 using other settings. Specifically, we use the following command lines for compressing a sequence *Video.yuv* whose resolution is $W \times H$ using x264 and x265 codecs,

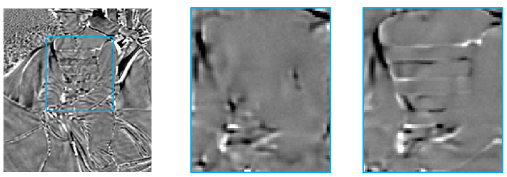
ffmpeg -y -pix_fmt yuv420p -s WxH -r FR -i Video.yuv -vframes N -c:v libx264 -crf Q -loglevel debug output.mkv
ffmpeg -y -pix_fmt yuv420p -s WxH -r FR -i Video.yuv -vframes N -c:v libx265 -x265-params "crf=Q" output.mkv

where *FR*, *N*, *Q* stand for the frame rate, the number of frames to be encoded, and the quality level, respectively.



(a) (b) (c)

Figure 4. Visualized results of compressing the BasketballPass sequence. (a) The original frame x_9. (b) The predicted frame $\bar{x}_9$ obtained by `Add MVRefine-Net` model with $\lambda = 64$. (c) The predicted frame $\bar{x}_9$ obtained by `Add MMC-Net` model with $\lambda = 64$. There are much more details in (c) than (b).



r_6 $\hat{r}'_6$ $\hat{r}_6$

Figure 5. Visualized results of compressing the RaceHorses sequence using `Proposed` model. From left to right: the original residual r_6, the decoded residual $\hat{r}'_6$, and the refined residual $\hat{r}_6$.

Fig. 8 presents the compression results on the UVG dataset and the HEVC Class B and Class D datasets. It can be observed that our proposed method achieves competitive results than x264 in PSNR, and is on par with x265 in MS-SSIM.

References

[1] Xiph test sequences. `http://media.xiph.org/video/derf`.

[2] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016.

[3] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018.

[4] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *ICCV*, pages 6421–6429, October 2019.

[5] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An end-to-end deep video compression framework. In *CVPR*, pages 11006–11015, June 2019.

[6] David Minnen, Johannes Ballé, and George D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018.

[7] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G. Anderson, and Lubomir Bourdev. Learned video compression. In *ICCV*, pages 3454–3463, October 2019.

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.

[9] 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 *ICIP*, pages 1509–1513. IEEE, 2016.

[10] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *ECCV*, pages 416–431, 2018.

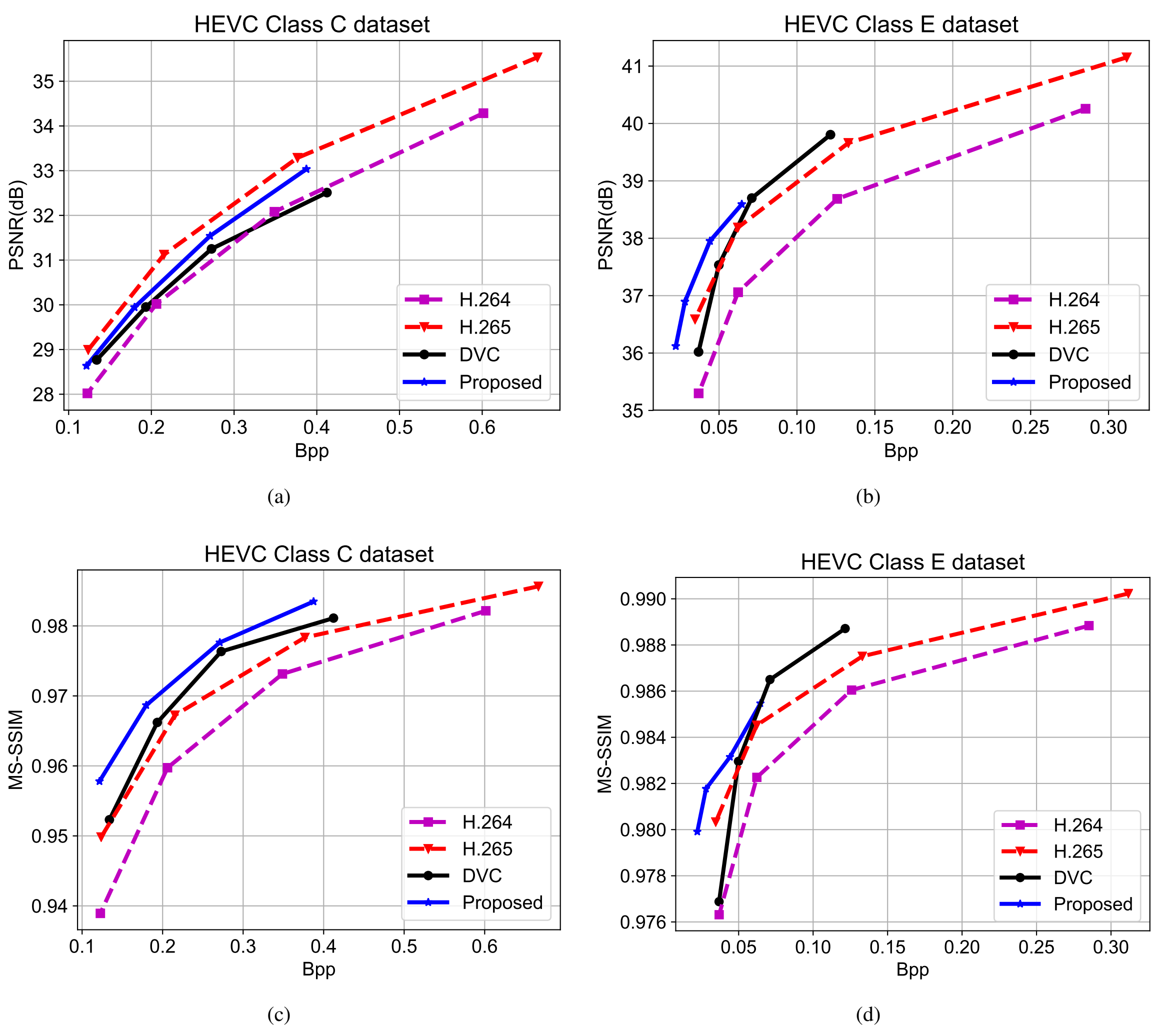
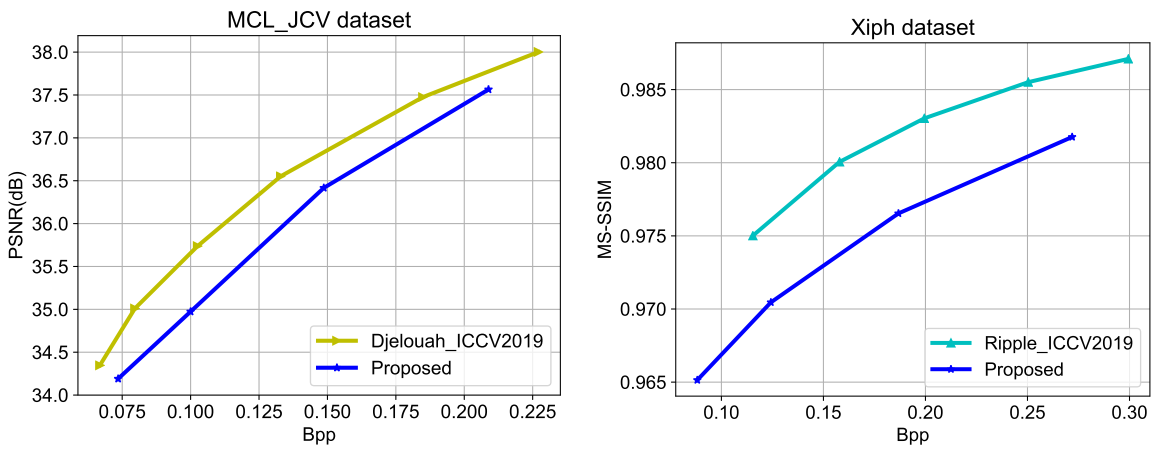


Figure 6. Compression results of H.264, H.265, DVC [5], and the proposed method on the HEVC Class C and E datasets. The results of H.264 and H.265 are cited from [5].



(a) (b)

Figure 7. Compression results of Djelouah_ICCV2019 [4], Rippel_ICCV2019 [7], and the proposed method on two different datasets. We directly cite the results reported in [4] and [7]. Please note that Djelouah_ICCV2019 [4] is designed for random-access scenarios and uses the autoregressive entropy model proposed in [6], while our method targets low-latency scenarios and just uses the fully-factorized ([2]) and hyperprior ([3]) entropy model. Rippel_ICCV2019 [7] is optimized for MS-SSIM but ours is optimized for MSE, PSNR results were not reported in [7].

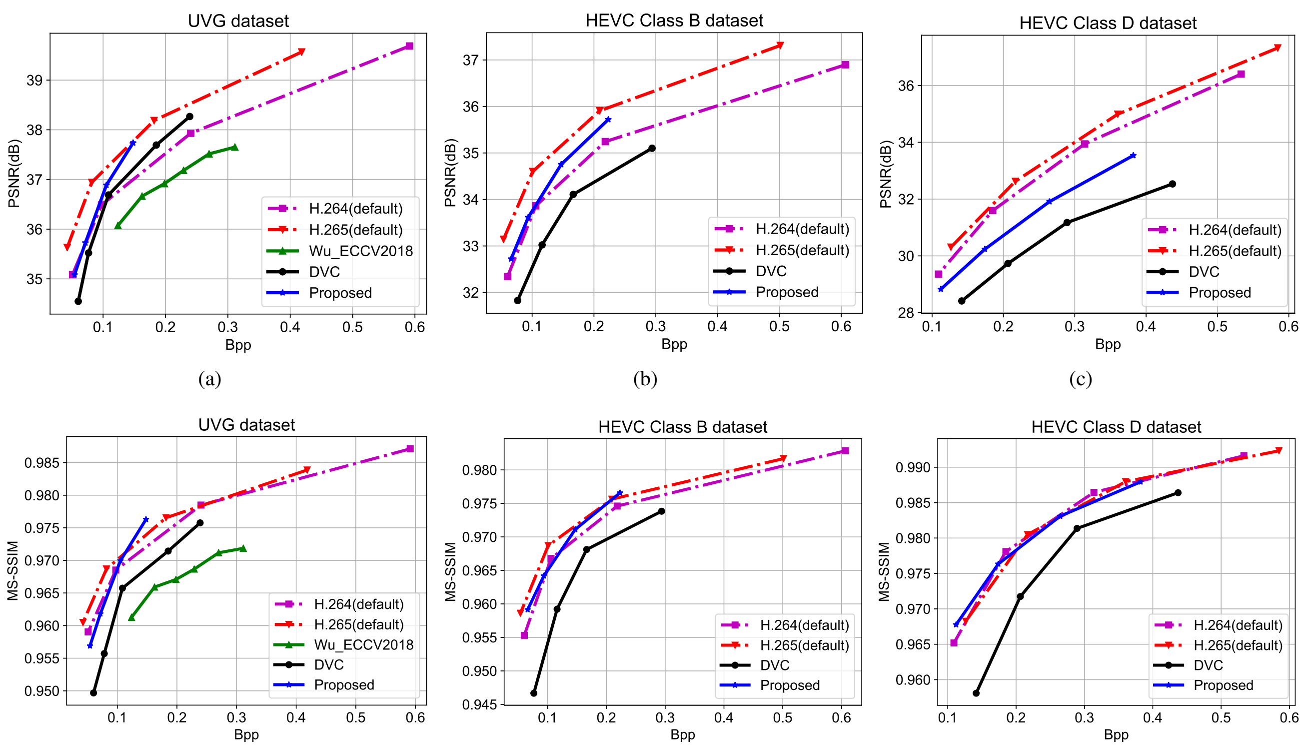


Figure 8. Compression results on the three datasets using H.264, H.265, DVC [5], Wu's method [10] and the proposed method. The settings of H.264 and H.265 are specified in the text. Top row: PSNR. Bottom row: MS-SSIM.