Hierarchical B-frame Video Coding Using Two-Layer CANF without Motion Coding - Supplementary Material

David Alexandre[1], Hsueh-Ming Hang[2], Wen-Hsiao Peng[3]
[1]Electrical Engineering and Computer Science International Graduate Program
[2]Institute of Electronics
[3]Department of Computer Science
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
{davidalexandre.eed05g,hmhang}@nctu.edu.tw, wpeng@g2.nctu.edu.tw

1. Variants of TLZMC

In this section, we introduce two additional variants of TLZMC: TLZMC** and TLZMC+. They are built upon TLZMC* and offer improved rate-distortion (RD) performance at the cost of slightly increased computational complexity. Table 1 shows the comparison between TLZMC, TLZMC*, TLZMC**, and TLZMC+.

1.1. System Architecture

The TLZMC** and TLZMC+ models incorporate the efficient CANF backbone in B-CANF [4], which is a variant (termed ECANF) of CANF [5] having fewer network parameters, thereby, reducing computational complexity at the cost of slightly lower coding performance. These models also feature modifications in several aspects, including the downsampling factor (TLZMC**, TLZMC+), the downsampler and super-resolution network (TLZMC**, TLZMC+), and the merging operation (TLZMC+). Figures 1 and 2 show the architectures of TLZMC** and TLZMC+, respectively. Specifically, TLZMC** adopts a downsampling factor of 2 (instead of 4), employs Max Pooling as the downsampler (using the built-in library from torch.nn.MaxPool2D), and adopts CARN [2] for super-resolution. But, TLZMC** still uses the multi-frame merging network from TLZMC. The implementation of CARN is from [1]. Taking one step further, TLZMC+ incorporates the above modifications and replaces the multi-frame merging net (MFMN) with the frame synthesis network in B-CANF [4].

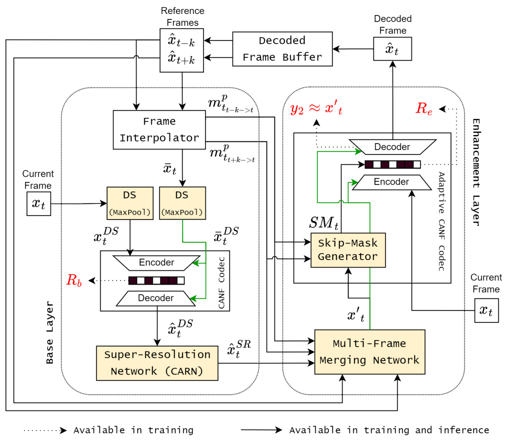


Figure 1. Framework of TLZMC**. As compared to its predecessor (TLZMC*), TLZMC** uses EfficientCANF as its base-layer and enhancement-layer codecs, max pooling as the downsampler, and CARN as the super-resolution network.

1.2. Training Procedure

For training TLZMC** and TLZMC+, we use the same training procedure of TLZMC*, except that the CARN has an initial model provided by its software [1].

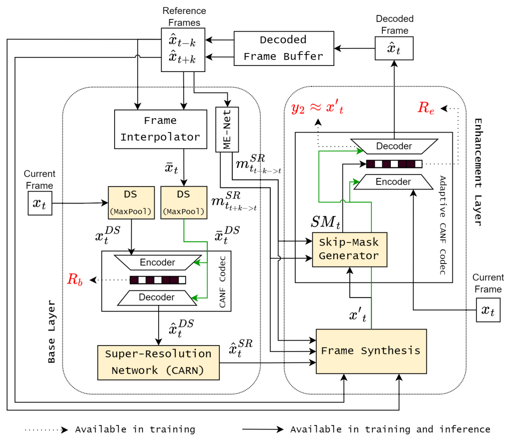


Figure 2. Framework of TLZMC+. It has the same architecture as TLZMC** except that the multi-frame merging network is replaced by the frame synthesis network in B-CANF [4].

Model	FA	Base Codec	Downsampler	Super Resolution	Merging	Size	Encode/Decode MACs (M/px)	UVG / HEVC-B BD-Rate Saving
B-CANF	Y	ECANF	-	-	FS	24.0M	2.70 / 1.97	-56.36 / -48.34
TLZMC	N	CANF	DS-Net (1/4x)	SR-Net	MFMN	39.9M	1.50 / 1.44	-42.39 / -24.15
TLZMC*	Y	CANF	DS-Net (1/4x)	SR-Net	MFMN	40.5M	1.57 / 1.48	-44.99 / -26.11
TLZMC**	Y	ECANF	Max Pooling (1/2x)	CARN	MFMN	27.5M	2.16 / 2.10	-51.50 / -36.36
TLZMC+	Y	ECANF	Max Pooling (1/2x)	CARN	FS	29.0M	2.39 / 2.33	-48.10 / -39.40

Table 1. Components, model size and computational complexity of B-CANF, TLZMC, TLZMC*, TLZMC**, and TLZMC+. FA: frame-type adaptive coding, ECANF [4]: EfficientCANF , MFMN: multi-frame merging network, CARN [2]: Super-resolution with Cascading Residual Network, FS [4]: Frame Synthesis.

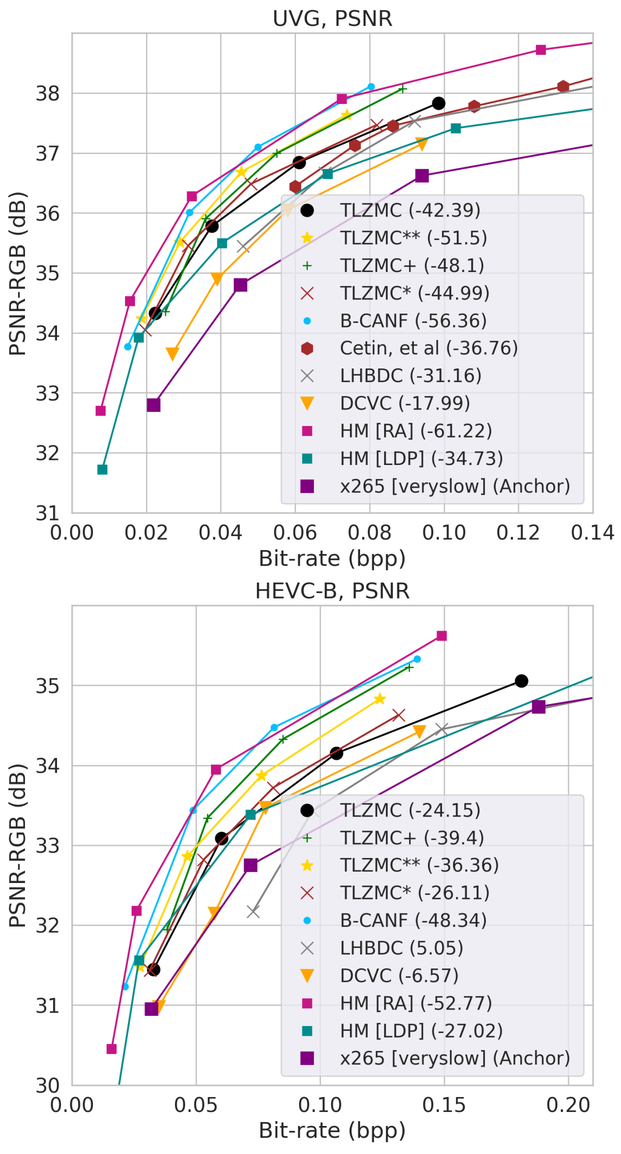


Figure 3. RD results for TLZMC variants compared to a few other high-performance benchmark schemes in terms of PSNR. We use the MSE model to evaluate TLZMC, TLZMC*, TLZMC**, and TLZMC+ under the same settings as the main paper.

Video Seq.	TLZMC	TLZMC*	TLZMC**	TLZMC+
(1) BasketballDrive	+45.67	+43.56	**+21.30**	+25.58
(2) BQTerrace	+84.24	+73.80	+51.43	**+2.27**
(3) Kimono1	+27.96	+26.59	**+13.46**	+13.84
(4) Cactus	+37.96	+22.65	**+6.79**	+13.85
(5) ParkScene	+61.37	+44.90	+30.21	**+17.09**

Table 2. BD-rate saving comparison on HEVC-B dataset. B-CANF is used as the anchor.

1.3. Experimental Results

We compare the rate-distortion (RD) performance of TLZMC, TLZMC*, TLZMC** and TLZMC+ in Figure 3, which includes also some high-performance B-frame and P-frame codecs. The test sequences are from UVG and HEVC Class-B datasets and the experiment settings are the same as those in the main paper. It is clear that the new variants provide higher BD-rate savings than TLZMC on both UVG and HEVC-B datasets. Particularly, their improvements on the HEVC-B dataset is more significant. When we examine individual videos, TLZMC** and TLZMC+ offer more improvements on the video sequences with a moving camera, such as ParkScene and BQTerrace in the HEVC-B dataset. In contrast, the videos in the UVG test dataset typically have slower movement and do not have complex textures as compared to HEVC-B videos. Table 2 and Table 3 present the BD-rate comparison among TLZMC, TLZMC*, TLZMC**, and TLZMC+, when B-CANF is the anchor. A negative BD-rate number indicates that the current variant outperforms B-CANF. On HoneyBee sequence, a stationary video with a small moving object, both TLZMC** and TLZMC+ outperform B-CANF. Also, we show the bitrate of the base layer in Table 4. We see that TLZMC** and TLZMC+ have much more bits spent on coding the base layer. This is attributed mainly to the smaller downsampling factor (i.e., 2) used in these two schemes. In other words, their base-layer images have larger size.

We adopt CANF as our image compressor and the system block diagram is shown in Figure 4. As Table 1 indicates, we use two types of CANF. The original CANF (referred to as CANF) was implemented according to [5]. The EfficientCANF (ECANF) was implemented according to [4], which has significantly smaller model size due to its reduced and trimmed network layers. CANF has 13.616 million parameters, while ECANF has only 7.339 million parameters. Hence, ECANF has lower computational complexity than and slightly inferior RD performance to the original CANF.

Video Seq.	TLZMC	TLZMC*	TLZMC**	TLZMC+
(1) Beauty	+19.39	+18.11	+14.67	**+14.40**
(2) Bosphorus	+28.21	+25.49	**+22.98**	+24.03
(3) HoneyBee	+4.01	+1.33	-11.10	**-13.40**
(4) Jockey	+34.77	+26.55	**+21.63**	+27.72
(5) ReadySteady	+30.24	+29.12	+27.29	**+27.13**
(6) ShakeNDry	+14.04	+13.11	+12.73	**+12.27**
(7) YachtRide	+18.02	+17.92	**+14.82**	+17.79

Table 3. BD-rate saving comparison on UVG dataset. B-CANF is used as the anchor.

1.4. Skip-Mode Coding Procedure

Figure 5 shows the operation of our skip-mode coding. The skip mask determines the locations of skipped feature samples. At the encoder, the skipped feature samples are not included in the coded bitstream. At the decoder, the same skip mask is generated; it guides the skipped sample insertion process, in which the skipped samples are inserted using the decoded μ values from the hyperprior. Figure 6 shows the structure of our adaptive CANF.

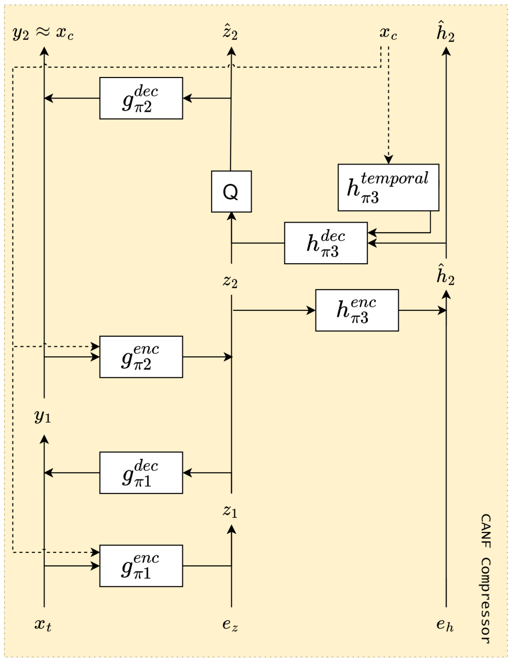


Figure 4. The architecture of basic CANF compressor. It takes x_t as the target signal and x_c as the condition signal. The reconstruction target is $\hat{x}_t = y_1 + g^{dec}_{\pi_1}(z_1)$

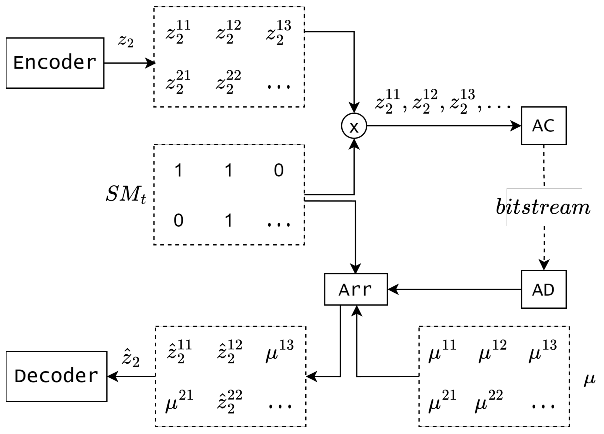

Encoder
z_2
z_2^{11} z_2^{12} z_2^{13}
z_2^{21} z_2^{22} ...
$z_2^{11}, z_2^{12}, z_2^{13}, \dots$
AC
SM_t
1 1 0
0 1 ...
bitstream
Arr
AD
Decoder
$\hat{z}_2$
$\hat{z}_2^{11}$ $\hat{z}_2^{12}$ μ^{13}
μ^{21} $\hat{z}_2^{22}$ ...
μ^{11} μ^{12} μ^{13}
μ^{21} μ^{22} ...
μ

Figure 5. The binary skip mask SM_t is used to determine the skipped samples in z_2. Only non-skipped samples go to the arithmetic coding units (AC and AD). The same SM_t is used to determine the locations of skipped samples on the decoder side. The received samples are arranged to recover their original locations, guided by SM_t, and the skipped samples are filled in with the μ values from the hyperprior to reconstruct $\hat{z}_2$ (cited from [3]).

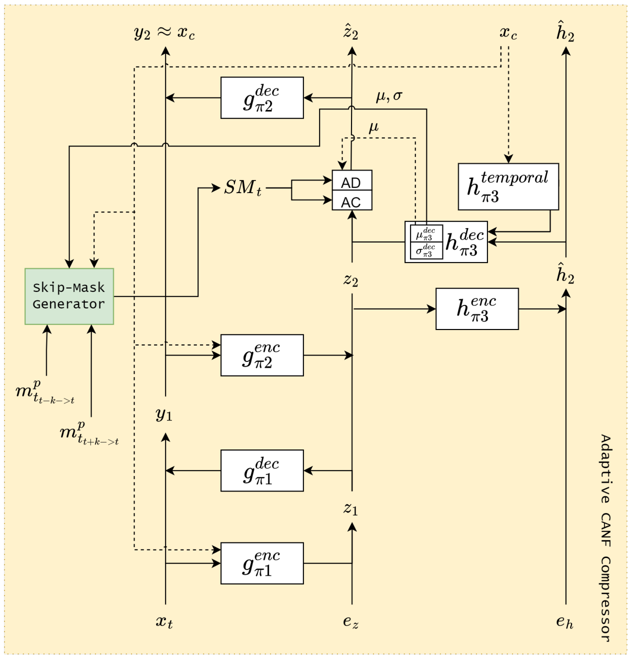


Figure 6. The architecture of adaptive CANF compressor. The arithmetic coder (AC) and decoder (AD) in CANF are modified to execute the skip-mode coding process. The decoder uses the μ values produced by the predictor (in $h^{dec}_{\pi 3}$) to replace the skipped samples.

1.5. Frame-type Adaptive Coding

The frame-type adaptive coding tool in B-CANF [4] classifies frames in a GOP into “reference B-frames” and ”non-reference B-frames”. Reference B-frames are used

λ	TLZMC	TLZMC* (DS4 & MFMN)	TLZMC** (DS2 & MFMN)	TLZMC+ (DS2 & FS [4])
	R / NR / AVG	R / NR / AVG	R / NR / AVG	R / NR / AVG
256	5.38 / 8.71 / 6.80%	18.05 / 15.96 / 17.41%	27.37 / 38.34 / 32.95%	51.94 / 70.38 / 59.31%
512	4.30 / 7.55 / 5.53%	11.58 / 8.45 / 10.52%	14.12 / 19.77 / 16.59%	35.79 / 43.93 / 38.93%
1024	2.99 / 6.43 / 4.51%	8.08 / 5.16 / 7.01%	8.21 / 10.82 / 9.67%	23.93 / 24.17 / 24.03%
2048	2.55 / 5.29 / 3.46%	5.31 / 2.83 / 4.31%	6.10 / 7.18 / 6.53%	15.75 / 13.13 / 14.56%
Average	3.80 / 6.99 / 5.07%	10.75 / 8.10 / 9.81%	13.95 / 19.03 / 16.44%	31.86 / 37.94 / 34.21%

Table 4. Percentages of the base-layer bitrate for 100 frames in all videos were calculated for the HEVC-B dataset. We use the MSE model for TLZMC, TLZMC*, TLZMC**, and TLZMC+. The percentages of the enhancement-layer bitrate can be derived by (100 - BL)%. The total bitrate excludes intra-frames. R: reference frames, NR: non-reference frames, and AVG: the average percentages of the base-layer bitrate over both reference and non-reference frames.

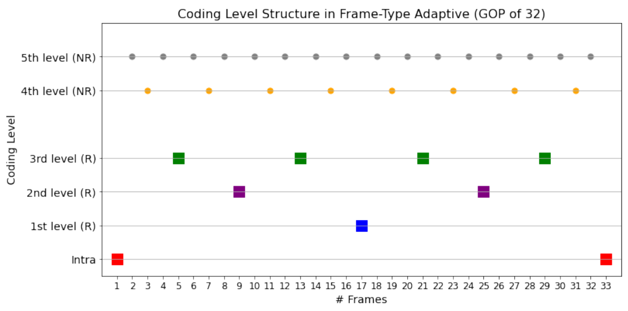


Figure 7. The specification of our frame-type adaptation in a GOP of 32 frames. The first three interpolation levels are reference frames and levels 4 and 5 are non-reference frames. R: Reference frame, NR: Non-Reference frame.

to generate interpolated frames; non-reference B-frames are never used to generate interpolated frames. If the GOP size is 32 and frame 1 and frame 33 are Intra-coded frames, then odd-indexed frames are reference frames and even-indexed frames are non-reference frames. We modify slightly the above classification in TLZMC* (and thus also in TLZMC** and TLZMC+). TLZMC* specifies the reference B-frames based on the levels of interpolation in a GOP. In a GOP of 32 frames, as shown in Figure 7, the first interpolation level is frame 17, which is generated (interpolated) by using frame 1 and frame 33. The second interpolation level includes frames 9 and 25, which are generated by using frames 1, 17, and 33. The third interpolation level consists of frames 5, 13, 21, and 29, generated using the frames of previous levels. The fourth level and the fifth level frames are similarly defined. In TLZMC*, the frames belonging to the first three interpolation levels are called reference B-frames. The remaining frames (levels 4 and 5) are non-reference B-frames. In contrast, only level 5 interpolated frames are non-reference frames in B-CANF. Higher bit rates are assigned to the reference-frames to improve the overall coding performance.

1.6. Merging-Map Generator Architecture

The merging-map generator architecture is shown in Figure 8. This architecture generates merging maps using a U-Net structure, taking three images as input: $\hat{x}_t^{SR}$, $warp(\hat{x}_{t-k}, m^p_{t_{t-k->t}})$, and $warp(\hat{x}_{t+k}, m^p_{t_{t+k->t}})$. The generator architecture uses max-pooling for down-sampling and transpose convolution for up-sampling, and generates three floating-point maps using softmax function.

1.7. Refinement Module Architecture

Figure 9 shows the refinement module architecture, which is also built on the U-Net structure and is made of convolution layers. This module takes the output generated by the merging-map generator, along with the reference frames $\hat{x}_{t-k}$ and $\hat{x}_{t+k}$, and generates a high-resolution image x'_t. The refinement module uses average-pooling for down-sampling and bi-linear interpolation for up-sampling.

2. Additional Experiment Results

2.1. Evaluation in x265 and HM-16.23

The x265 compressed videos in this paper were generated using the FFmpeg software with the x265 [veryslow] setting. The command line for x265 encoding is:

```
ffmpeg -pix_fmt yuv420p -s WxH -r FR -i input_file.yuv
-vframes N -c:v libx265 -preset veryslow -tune zerolatency
-x265-params "qp=QP:keyint=GOP" output
```

QP stands for the quality point, configured as 19, 22, 27, 32, 37. For the standard test protocol, GOP is set to 10 for HEVC Class B and 12 for UVG datasets.

To obtain HM-16.23 [LDP] results, we use the configuration file *encoder_lowdelay_P_main.cfg* with the following settings:

```
-i=input.yuv -fr=FR -wdt=W -hgt=H -f=N -ip=32 -g=8
-dr=2 -q=QP -b bistream -o output
```

For HM-16.23 [RandomAccess], we use the configuration file *encoder_randomaccess_main.cfg* with the following settings:

```
-i=input.yuv -fr=FR -wdt=W -hgt=H -f=N -ip=32 -g=16
-q=QP -b bistream -o output
```

The value of QP for [LDP] is set to 17, 22, 24, 27,

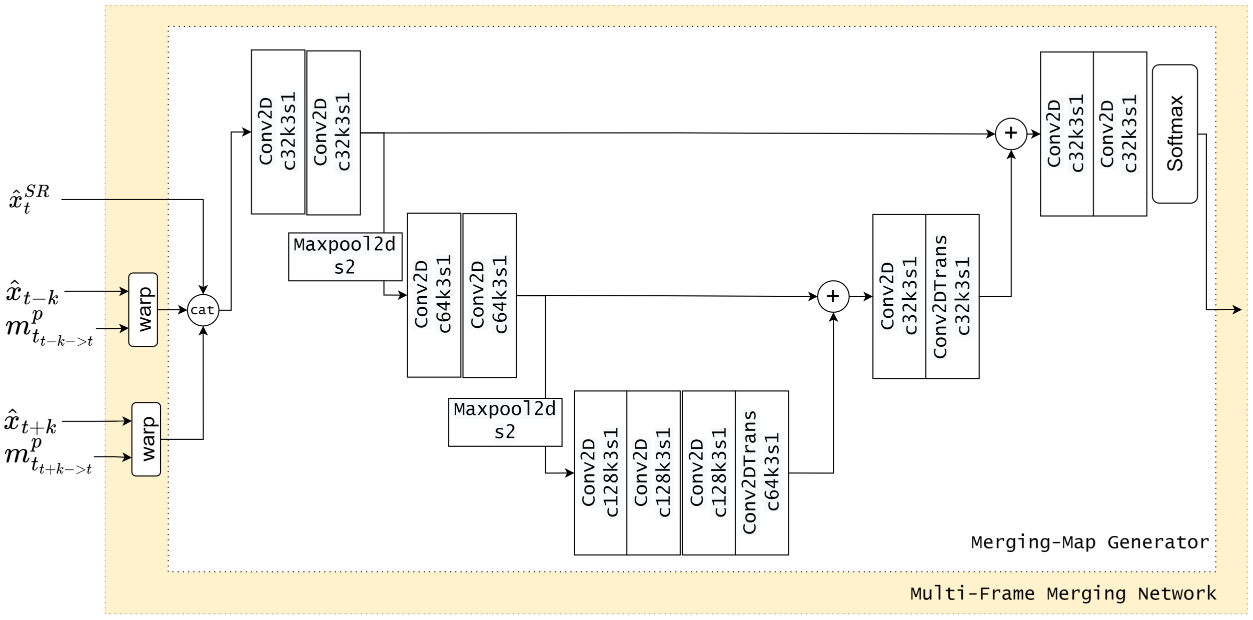


Figure 8. The merging-map generator takes three inputs: $\hat{x}_t^{SR}$, $warp(\hat{x}_{t-k}, m^p_{t_{t-k->t}})$, and $warp(\hat{x}_{t+k}, m^p_{t_{t+k->t}})$, and generates the merging-map using a U-Net structure. Downsampling is performed using the max-pool operation, and up-sampling is performed using transpose convolution. Finally, the softmax function is applied at the end of the network to generate floating-point maps.

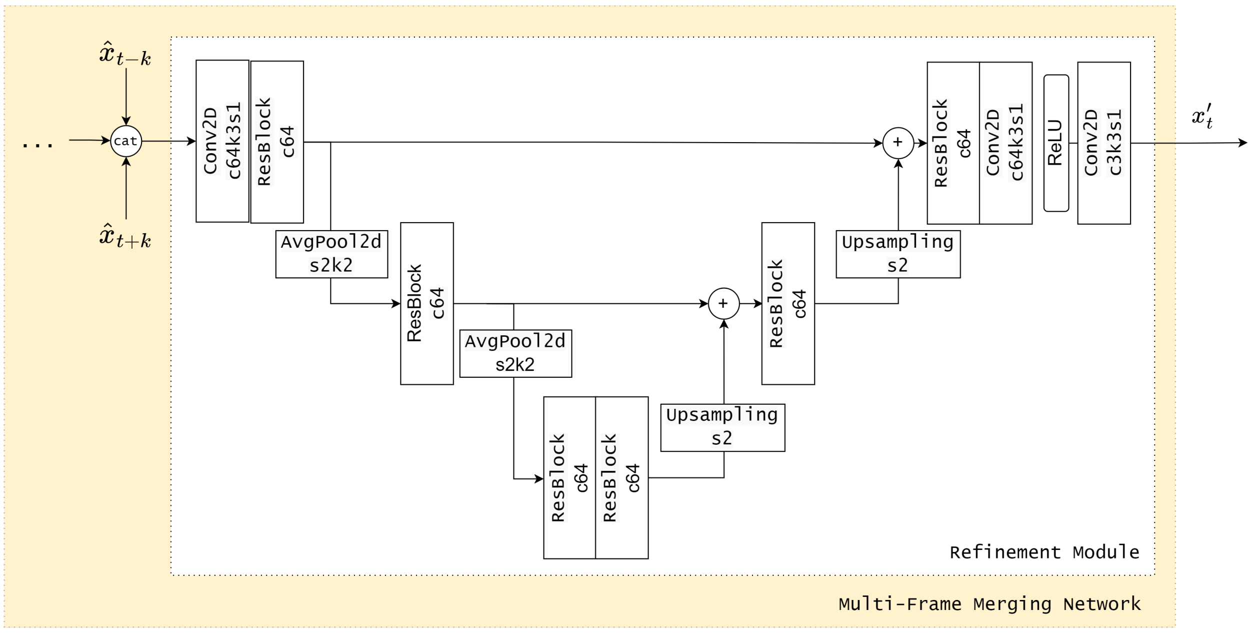


Figure 9. The refinement module of the multi-frame merging network (MFMN) uses a U-Net structure to produce output x'_t based on the inputs $\hat{x}_{t-k}$, $\hat{x}_{t+k}$, and the merging output from MFMN. This module consists of convolution layers, with average pooling used for downsampling and bilinear interpolation for upsampling. The output goes through a ReLU function before the construction of x'_t.

32, while for [RandomAccess] is set to 19, 22, 27, 32, 37 The symbols FR, N, and QP correspond to the frame rate, the number of frames to be encoded, and the quantization value.

2.2. Visual Quality

In Figure 10, we compare the visual quality of TLZMC coded images to that of the standard codecs (x265 [veryslow] and HM-16.23 [RandomAccess]) in low-rate settings. Our reconstructed image (10c) of the Jockey sequence in the UVG dataset use the MSE model with λ=256

(PSNR=35.47 dB, bitrate=0.0047 bpp). It has fewer reconstruction artifacts than x265 [veryslow] (10e) at a similar bitrate (PSNR=35.16 dB, bitrate=0.0047 bpp). However, the quality of the reconstructed image using HM-16.23 [RandomAccess] (10g) is higher; it is consistent with the fact that it has higher PSNR (PSNR=35.93 dB, bitrate= 0.0046 bpp). We also show the reconstructed image of the ShakeNDry sequence of the UVG dataset using our MS-SSIM model. Compared to x265 [veryslow] (10f) and HM-16.23 [RandomAccess] (10h), our TLZMC with MS-SSIM model (10d) provides better visual quality at similar bitrates (∼0.075 bpp). This is reflected in the MS-SSIM metrics, where TLZMC has MS-SSIM=0.9605 while x265 [veryslow] has MS-SSIM=0.9299 and HM-16.23 [RandomAccess] has MS-SSIM=0.9420. In general, our approach can preserve more details in the coded images compared to the traditional codecs.

References

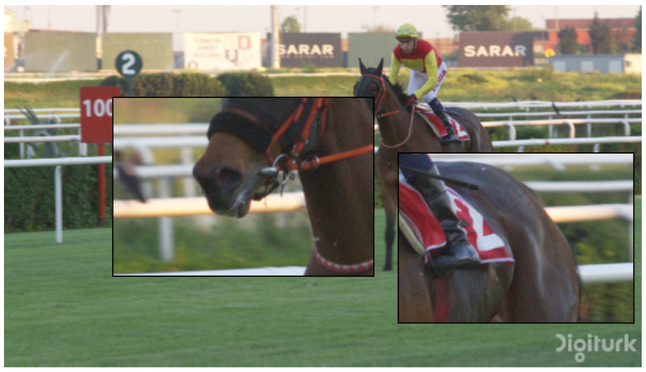
[1] Carn software, 2022. https://github.com/nmhkahn/CARN-pytorch. 1

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. *arXiv preprint arXiv:1803.08664*, 2018. 1, 2

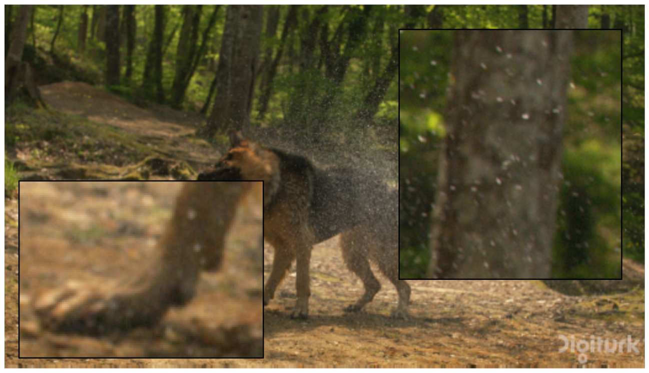
[3] David Alexandre, Hsueh-Ming Hang, and Wen-Hsiao Peng. Two-layer learning-based p-frame coding with super-resolution and content-adaptive conditional anf. In *Proceedings of the 4th ACM International Conference on Multimedia in Asia*, pages 1–7, 2022. 3

[4] Mu-Jung Chen, Yi-Hsin Chen, Peng-Yu Chen, Chih-Hsuan Lin, Yung-Han Ho, and Wen-Hsiao Peng. B-canf: Adaptive b-frame coding with conditional augmented normalizing flows. *arXiv:2209.01769v1*, 2022. 1, 2, 3, 4

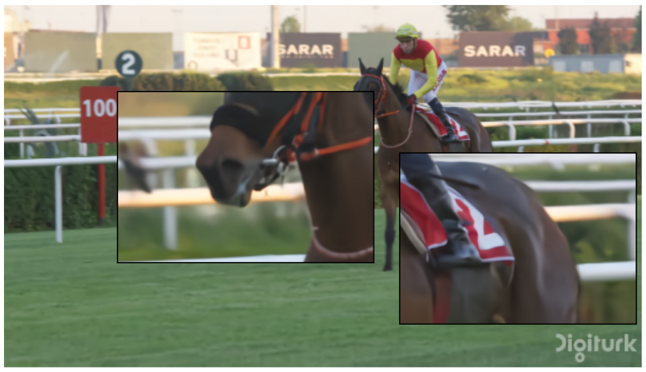
[5] Yung-Han Ho, Chih-Chun Chan, Wen-Hsiao Peng, Hsueh-Ming Hang, and Marek Domański. Anfic: Image compression using augmented normalizing flows. *IEEE OJCAS*, 2:613–626, 2021. 1, 2



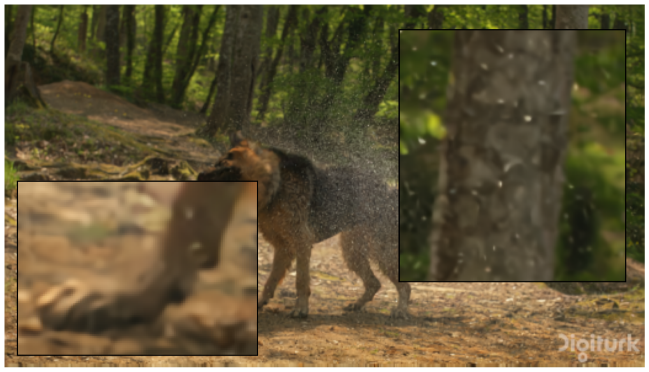
(a)



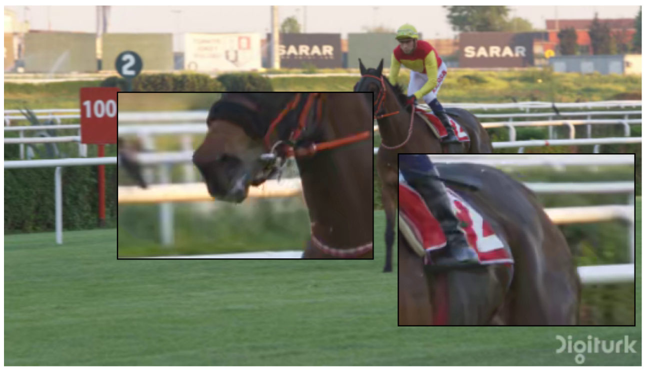
(b)



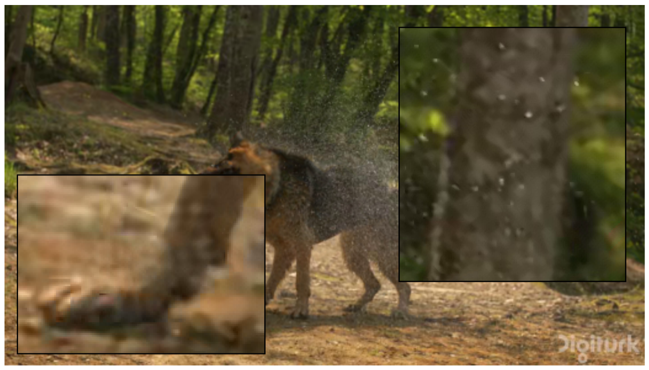
(c)



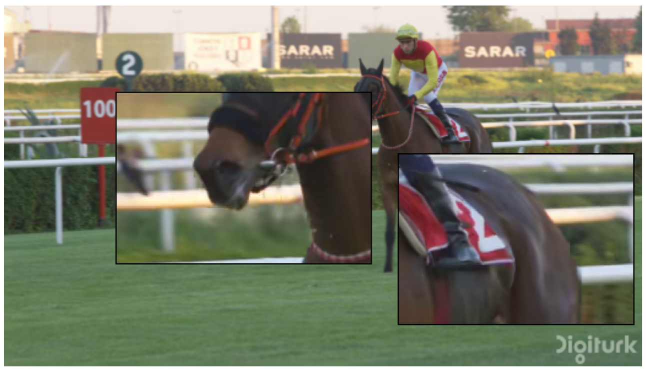
(d)



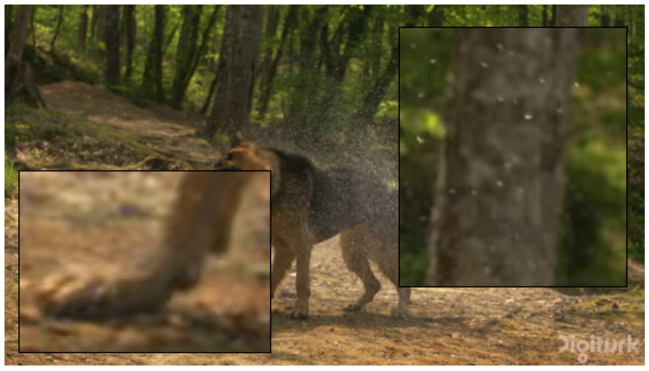
(e)



(f)



(g)



(h)

Figure 10. The visual quality of TLZMC. (a,b) show the original pictures of Jockey and ShakeNDry video sequences. (c) is produced using the MSE model of TLZMC with λ=256. (PSNR=35.47 dB, bitrate=0.0047 bpp) (d) is produced using the MS-SSIM model TLZMC with λ=4 (MS-SSIM=0.9605, bitrate=0.0762 bpp). (e,f) are produced by x265 [veryslow]: for (e), PSNR=35.16 dB, bitrate=0.0047 bpp; and for (f), MS-SSIM=0.9299, bitrate=0.0754 bpp. (g,h) are produced by HM-16.23 [RandomAccess]: for (g), PSNR=35.93 dB, bitrate=0.0046 bpp; and for (h), MS-SSIM=0.9420, bitrate=0.0765 bpp. The QP values of x265 and HM-16.23 were selected to reconstruct the frames at similar bpps to our codec.