A. Experimental Results

A.1. BDBR and BD-PSNR(BD-MSSSIM) Results

Table 1: BDBR and BD-PSNR(BD-MSSSIM) performances of H.265 and our DVC model when compared with H.264.

Sequences		H.265				Ours			
		PSNR		MS-SSIM		PSNR		MS-SSIM	
		BDBR (%)	BD-PSNR(dB)	BDBR (%)	BD-MS SSIM(dB)	BDBR (%)	BD-PSNR(dB)	BDBR (%)	BD-MS SSIM(dB)
ClassB	BasketballDrive	-44.37	1.15	-39.80	0.87	-23.17	0.59	-22.21	0.51
	BQTerrace	-28.99	0.68	-25.96	0.50	-25.12	0.54	-19.52	0.36
	Cactus	-30.15	0.68	-26.93	0.47	-39.53	0.94	-41.71	0.86
	Kimono	-38.81	1.19	-35.31	0.97	-40.70	1.23	-33.00	0.92
	ParkScene	-16.35	0.45	-13.54	0.29	-25.20	0.77	-29.02	0.77
Average		**-31.73**	**0.83**	**-28.31**	**0.62**	**-30.75**	**0.81**	**-29.09**	**0.68**
ClassC	BasketballDrill	-35.08	1.69	-34.04	1.41	-24.47	1.05	-27.18	1.18
	BQMall	-19.70	0.84	-17.57	0.60	26.13	-0.72	-18.85	0.67
	PartyScene	-13.41	0.60	-13.36	0.53	-9.14	0.29	-37.18	1.61
	RaceHorses	-17.28	0.69	-17.01	0.57	-8.06	0.19	-29.24	1.05
Average		**-21.37**	**0.96**	**-20.50**	**0.78**	**-3.88**	**0.20**	**-28.11**	**1.13**
ClassD	BlowingBubbles	-12.51	0.50	-10.28	0.35	-17.79	0.62	-35.44	1.53
	BasketballPass	-19.26	0.99	-17.98	0.85	-0.39	-0.01	-20.53	1.01
	BQSquare	-3.49	0.14	5.90	-0.19	-1.60	0.01	-23.67	0.84
	RaceHorses	-14.77	0.68	-13.23	0.56	-18.95	0.72	-29.79	1.30
Average		**-12.51**	**0.58**	**-8.89**	**0.39**	**-9.68**	**0.34**	**-27.36**	**1.17**
ClassE	Vidyo1	-37.12	1.11	-31.67	0.55	-36.05	1.20	-36.80	0.72
	Vidyo3	-34.99	1.23	-29.48	0.65	-32.58	1.25	-40.09	1.02
	Vidyo4	-34.71	1.05	-27.41	0.61	-30.84	1.03	-24.84	0.66
Average		**-35.61**	**1.13**	**-29.52**	**0.61**	**-33.16**	**1.16**	**-33.91**	**0.80**
Average Over All Sequences		**-25.06**	**0.85**	**-21.73**	**0.60**	**-19.22**	**0.61**	**-29.32**	**0.94**

In the video compression task, BDBR and BD-PSNR (BD-MSSSIM) are widely used to evaluate the performance [1] of different video compression systems. BDBR represents the average percentage of bit rate savings when compared with the baseline algorithm at the same PSNR (MS-SSIM). BD-PSNR (BD-MSSSIM) represents the gain (dB) when compared with the baseline algorithm at the same bit rate.

In Table 1, we provide the BDBR and BD-PSNR(BD-MSSSIM) results of H.265 and the our proposed method DVC when compared with H.264. When the distortion is measured by PSNR, our proposed model saves 19.22% bit rate, while H.265 saves 25.06% bit rate. When measured by MS-SSIM, our proposed method can save more than 29% bit rate, while H.265 only saves 21.73%. It clearly demonstrates that our proposed method outperforms H.264 in terms of PSNR and MS-SSIM. Meanwhile, the performance of our method is comparable or even better than H.265 in terms of PSNR and MS-SSIM, respectively.

A.2. Compression Performance on the HEVC Class C and Class D

In the submitted paper, we provide the performance on the HEVC Class B and Class E datasets. In this section, we also compare our proposed method with the traditional video compression algorithms H.264 and H.265 on the HEVC Class C and Class D datasets in Figure 1. Our proposed method still outperforms the H.264 algorithm in terms of PSNR and MS-SSIM. According to Table 1, the newly proposed DVC model saves 3.88% and 9.68% bit rates when compared with H.264 on the HEVC Class C and Class D datasets, respectively.

A.3. Result Using Kinetics Dataset as the Train Dataset

Wu *et al.* utilized the Kinetics datasets to train the video compression mdoel in [2]. In Figure 2, we also report the performance of our model when using the Kinetics dataset as the train dataset, which is denoted as *Ours_Kinetics* (see the

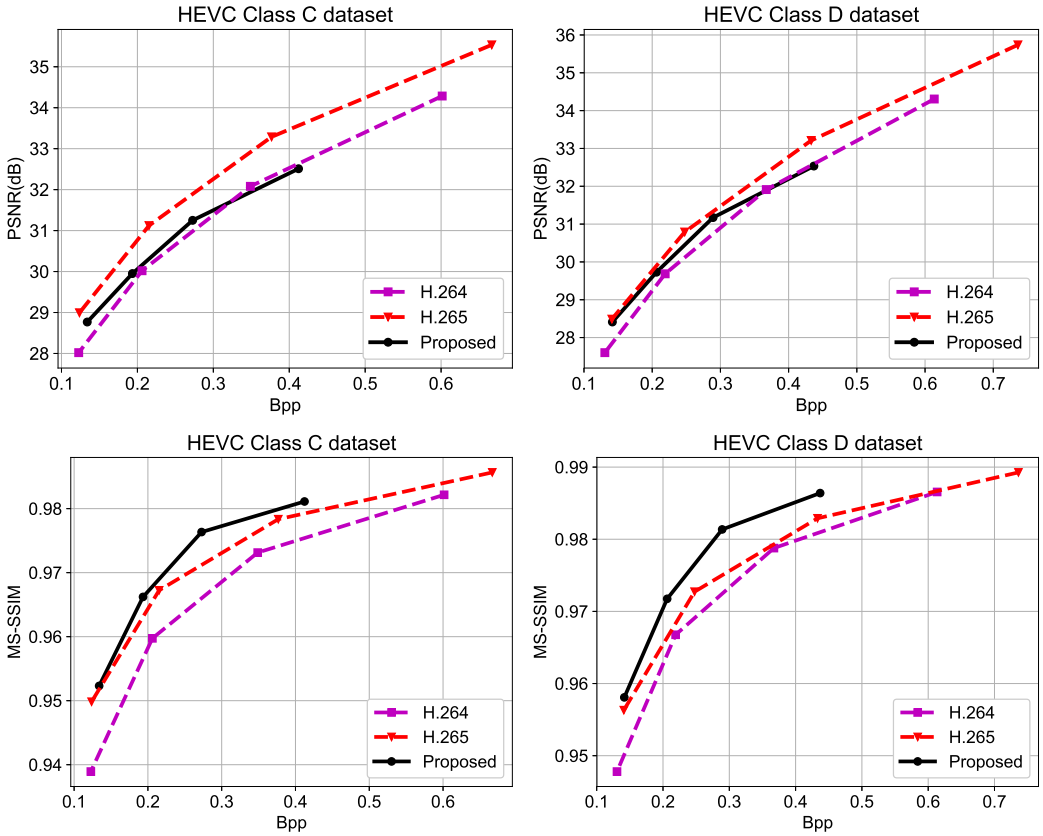

Figure 1: Compassion of our method with the traditional video compression methods H.264 and H.265 on the HEVC Class C and Class D datasets.

green curve). It is obvious that our method trained based on Kinetics dataset outperforms the H.264 and the baseline method [2].

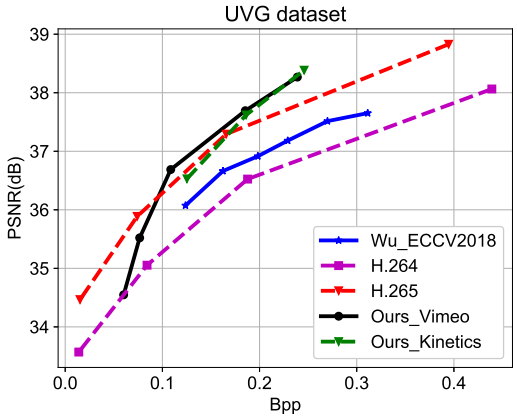

Figure 2: Compassion of our method with the traditional video compression methods H.264 and H.265 on the UVG dataset.

B. Experimental Settings

B.1. H.264 and H.265 Setting

In order to generate the compressed videos from H.264 and H.265, we follow the setting in [2] (confirmed from the first author) and use the FFmpeg with the *very fast* mode. Given the uncompressed video sequence *Video.yuv* with the resolution of $W \times H$, the command line for generating H.264 compressed video is provided as follows,

ffmpeg -y -pix_fmt yuv420p -s WxH -r FR -i Video.yuv -vframes N -c:v libx264 -preset veryfast -tune zerolatency -crf Q -g

GOP-bf 2 -b_strategy 0 -sc_threshold 0 -loglevel debug output.mkv

The command line for generating H.265 is provided as follows,

ffmpeg -pix_fmt yuv420p -s WxH -r FR -i Video.yuv -vframes N -c:v libx265 -preset veryfast -tune zerolatency -x265-params "crf=Q:keyint=GOP:verbose=1" output.mkv

Among them, FR, N, Q, GOP represents the frame rate, the number of encoded frames, quality, GOP size, respectively. N is set to 100 for the HEVC datasets. Q is set as 15,19,23,27 in our settings. GOP is set as 10 for the HEVC dataset and 12 for the UVG dataset.

B.2. PSNR, MS-SSIM and Bpp

In our experiments, PSNR and MS-SSIM are evaluated on the RGB channel. We average the PSNRs from all vdieo frames in each sequence to obtain the corresponding PSNR of this sequence. Given the compressed frame with the resolution of $W \times H$ and the corresponding bit cost R, the Bpp is calculated as follows,

$$Bpp = R/W/H \tag{1}$$

C. Network Architecture of Our Motion Compensation Network

The motion compensation network is shown in Figure 3. $w(\hat{x}_{t-1}, \hat{v}_t)$ represents the warped frame. We use residual block with pre-activation structure.

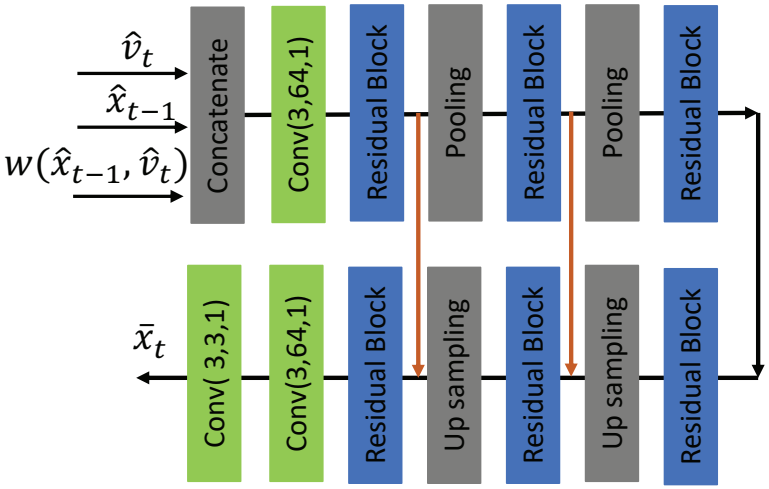

Figure 3: Network architecture of the motion compensation network.

References

[1] G. Bjontegaard. Calculation of average psnr differences between rd-curves. *VCEG-M33*, 2001. 1

[2] C.-Y. Wu, N. Singhal, and P. Krahenbuhl. Video compression through image interpolation. In *ECCV*, September 2018. 1, 2