Deep frame interpolation for video compression

Jean Bégaint*†, Franck Galpin*, Philippe Guillotel* and Christine Guillemot†

*Technicolor
Av. des Champs Blancs
35576 Cesson-Sévigné, France
firstname.lastname@technicolor.com

† INRIA
Campus de Beaulieu
35042 Rennes, France
firstname.lastname@inria.fr

Abstract

Deep neural networks have been recently proposed to solve video interpolation tasks. Given a past and future frame, such networks can be trained to successfully predict the intermediate frame(s). In the context of video compression, these architectures could be useful as an additional inter-prediction mode. Current inter-prediction methods rely on block-matching techniques to estimate the motion between consecutive frames. This approach has severe limitations for handling complex non-translational motions, and is still limited to block-based motion vectors. This paper presents a deep frame interpolation network for video compression aiming at solving the previous limitations, *i.e.* able to cope with all types of geometrical deformations by providing a dense motion compensation. Experiments with the classical bi-directional hierarchical video coding structure demonstrate the efficiency of the proposed approach over the traditional tools of the HEVC codec.

Introduction

Inter-prediction is crucial to efficiently compress video content. By capturing temporal redundancies between consecutive frames, inter-prediction methods are able to generate accurate predictions of the frames to encode. The motion between frames is traditionally estimated and compensated via block-based translational motion vectors. To correct local inaccuracies, a residue may then be added to the prediction. Inter-prediction methods can use past and future frames to predict current frames, which is known as bi-directional prediction. This bi-directional prediction is generally used in a hierarchical manner to encode a group of pictures (GOP).

Video frame interpolation is used to generate intermediate frames between two consecutive input frames. Recently, several works have demonstrated improvements in solving frame interpolation tasks by using deep neural networks [1, 2, 3, 4]. These approaches usually estimate an optical flow, *i.e.* a per pixel translational displacement, then warp and blend the original input frames to generate the interpolated frame. Video frame interpolation thus shares some characteristics with inter-prediction methods. In the context of video compression, such deep architectures could be useful as an additional inter-prediction technique. Indeed these deep learning approaches are able to estimate the motion between frames in a spatially coherent manner, at the pixel level, which leads to good interpolation results. Moreover, these methods benefit from the fact that no motion information need to be transmitted to the decoder.

This paper introduces a novel inter-prediction method for video compression based on deep frame interpolation networks. For each frame that can be coded with bi-directional inter-prediction from past and future frames, an additional reference frame

is provided by interpolating the original reference frames. The method was integrated within the HEVC video codec [5]. BD-rates improvements are measured using the Common Test Condition (CTC) sequences [6].

Related Work

Video frame interpolation algorithms have benefited from recent advances with deep neural networks. Several deep architectures have been proposed in the last few years [7, 8, 1, 2, 3, 4], constantly improving the quality of the interpolation results. These networks usually operate in two stages. First a kind of motion estimation is performed, such as optical flow, then the frame is interpolated from the reference input frames and the estimated motion.

Liu *et al.* [1] were the first to introduce an architecture with an unsupervised learning of the optical flow. Similarly to the classical fully convolutional approaches, their *Deep Voxel Flow* (DVF) network is trained end-to-end to interpolate an intermediate frame from two consecutive input frames. However, the last layer of the network is a non-trainable interpolation function interpolating the previous layers output and the input frames. The loss function is computed directly as the $l1$-norm between the interpolated frame and the ground truth. The network thus learns by itself to estimate a kind of optical flow and a temporal mask for the trilinear interpolation layer. The network outputs a *voxel flow* which represents the per pixel displacement (dx, dy) and a temporal mask dt weighting the trilinear blending.

Niklaus *et al.* proposed in [9] an adaptive convolution approach to video frame interpolation (*AdaConv*). Instead of relying on a two-steps approach, *i.e.* first a motion estimation then a pixel synthesis, the authors proposed to implement the pixel synthesis as a convolution over the two input frames. They trained a deep neural network to predict a per pixel spatially adaptive convolution kernel. Thus for each pixel, parameters for a 2D convolutional kernel are predicted, then the input frames are convolved at the current pixel location to predict its interpolated value. Their method is able to handle occlusions, blur and brightness change. However the large kernel to be estimated for each pixel is computationally expensive, as they need a 51×51 pixel kernel to handle large motions. This drawback is addressed in their following work [2] in which they introduced separable convolutions (*SepConv*) using 1D kernels for faster processing.

In the image compression domain, several deep learning approaches have also been recently proposed [10, 11, 12, 13]. Current performances already exceed classical image codecs like JPEG [14] and JPEG2000 [15], and can reach state-of-the-art codecs such as HEVC [5] (PSNR). Fewer works have been published on video compression. Han *et al.* [16] proposed an end-to-end trainable video codec based on a variational auto-encoder which incorporates both global and per frame latent variables to capture temporal redundancies. Wu *et al.* introduced in [17] an end-to-end architecture based on an image interpolation network to exploit temporal redundancies. A compressed residual information is also learned jointly to correct interpolation inaccuracies.

Similarly, this paper builds upon existing deep frame interpolation methods to target video compression applications. However, the frame interpolation method is

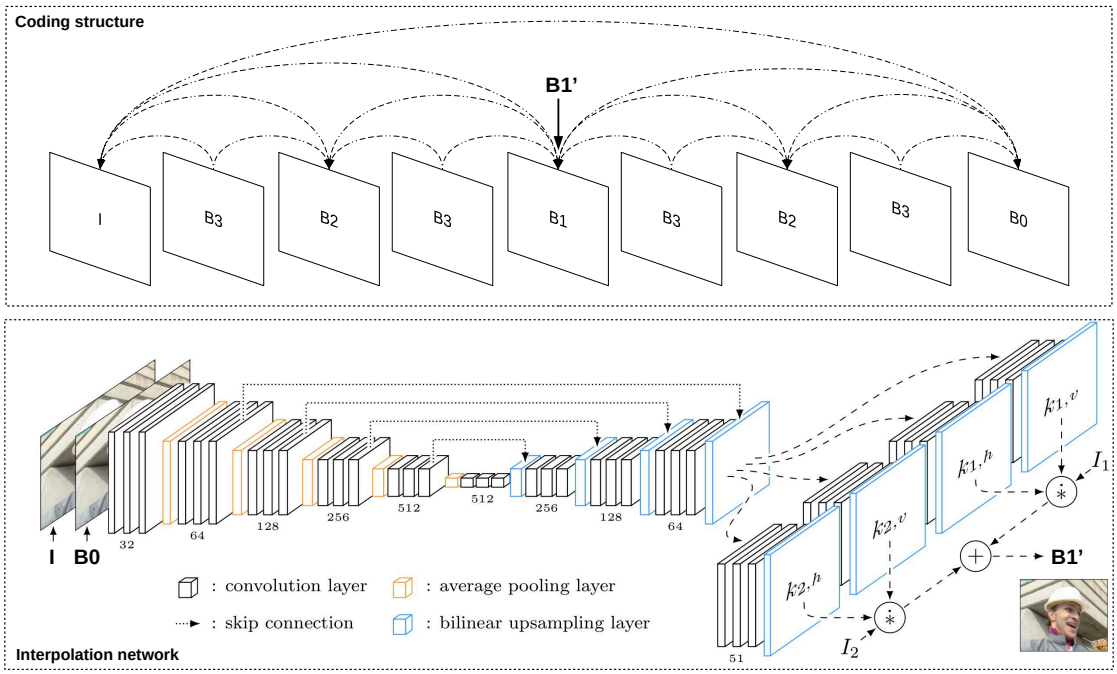


Figure 1: Coding scheme: illustration for a GOP of 8 frames with the default *random access* configuration. The *SepConv* [2] network is used here as the interpolation network. The dashed lines represent the default references, the continuous line represent the extra reference from the deep interpolation. Example is shown for the B1 frame.

introduced here within the classical hybrid video codec architecture, which allows to take advantage of the efficient residual coding and entropy signaling of HEVC.

Coding scheme

To test deep frame interpolation networks in a video compression context, the HEVC HM reference software [6] was modified. For each frame hierarchically coded in the GOP, an extra reference frame, interpolated by the network, is appended to the default reference frames list and can be selected as reference by the rate-distortion optimization process. This new GOP structure is represented in Figure 1. For example, the *B1* frame can be predicted with the interpolated reference from the *I* and *B0* frames, in addition to the default HEVC reference frames.

Using extra reference frames has two advantages over fully implementing a new inter-prediction mode in HEVC. First, the design change on the legacy decoder is relatively straightforward, but mostly it allows the encoder to use additional inter-prediction modes from the interpolated reference. Although the best configuration would be to only have blocks with SKIP mode as prediction mode (only motion vector predictions, no residual), MERGE or even INTER from the interpolated frames can still be beneficial compared to a prediction from the default reference frames. To support the use of extra reference frames, several flags need to be inserted into the bitstream. The decoder will then re-interpolate the selected frames when needed, and

thus needs to know if an extra reference frame was used as prediction or not for each coded block.

At the frame level, a flag is first set to signal the use or not of the extra reference frames. At the Prediction Unit (PU) level, an integer is added to the motion vector information to store the use of the selected extra reference frame. The sentinel value 0 is set as a signaling method to inform the decoder that no extra reference frames are used for the current PU. The interpolated frame flag is entropy coded with CABAC [18] in order to reduce the syntax size. A CABAC context is also determined from the top and left PU blocks, if they are available for the current block. It should be noted that no specific study was performed on the CABAC context or its initialization, study that would be beneficial once more statistics about the proposed method are obtained.

Experimental results

The coding experiments are performed on the common test sequences [6]. The HEVC HM software version 16.16 was used for all the experiments. The rate-distortion performance are computed with the Bjontegaard metric [19] using the common 22, 27, 32, 37 Quantization Parameter (QP). Unless specified otherwise, the PSNR is reported for the Y channel only. The default HM *random access* configuration mode [6] is used as a baseline for the following tests. The default fixed GOP size of 16 is used. Experiments are run with sequences of the classes B, C and D of the CTC [6], affine sequences class [20] and the legacy *Foreman* sequence. HEVC is used as the baseline reference. Experiments were conducted with the *SepConv* approach [2], respectively trained with a $l1$-norm (*SepConv-l1*) and a perceptual loss (*SepConv-lf*), the *Deep Voxel Flow* (DVF) approach [1]. A non deep learning approach, *Celiu Flow* [21] is used as an additional baseline to measure the effect on using a dense optical flow at the encoder and decoder, compared to the traditional block-based motion compensation.

BD-rate results

The BD-rate results are reported in **??**. First, one can note that all the experiments bring improvements over the HEVC codec. The best performances are obtained for the *SepConv-l1* method with a mean BD-rate reduction of -2.46%, compared to -2.38% for the *SepConv-lf* network, -0.09% for DVF and -0.89% for the *CeliuFow* method.

The *SepConv-l1* method outperforms the other methods. This network was specifically trained with the $l1$-norm, which minimizes the interpolation error energy, explaining the greater average performance of -0.08% over the *SepConv-lf* network. The $l1$-norm is often preferred over the $l2$-norm, as it has been repeatedly shown that it leads to less blurry results [7]. Surprisingly, the *SepConv-lf* performs better on three of the sequences, and although it was trained with a perceptual norm the performance loss compared to the one trained with the $l1$-norm is relatively small. Both *SepConv* approaches outperform the classical optical flow approach by -1.45%. However the *Deep Voxel Flow* network has the worst bit-rate distortion performances, with a mean gain of -0.09%. Some values could also not be reported due to memory limitations

Table 1: BD-rate performances comparison for different frame interpolation methods compared to HEVC.

Class	Sequence	SepConv-l1	SepConv-lf	DVF	CeLiu Flow
B	BQTerrace	**-6.38%**	-5.94%	–	-2.77%
	BasketballDrive	**-1.43%**	-1.05%	–	-0.35%
	Cactus	**-7.36%**	-7.07%	–	-4.56%
	Kimono1	**-2.05%**	-1.71%	–	-0.23%
	ParkScene	**-2.36%**	-2.14%	–	-0.41%
C	BQMall	**-3.44%**	-2.89%	-0.10%	-0.34%
	BasketballDrill	**-3.87%**	-3.77%	-0.71%	-1.17%
	PartyScene	-2.53%	**-3.08%**	-0.14%	-0.37%
	RaceHorses	**-1.30%**	-1.01%	-0.03%	-0.47%
D	BQSquare	-3.31%	**-5.48%**	0.10%	0.06%
	BasketballPass	**-2.93%**	-2.51%	-0.13%	-0.34%
	BlowingBubbles	-2.08%	**-2.29%**	-0.13%	-0.24%
	RaceHorses	**-2.21%**	-1.84%	-0.06%	-0.29%
Affine	CStoreGoods	**-0.45%**	-0.29%	–	-0.25%
	DrivingRecorder1	**-1.37%**	-0.93%	0.04%	-0.07%
	DrivingRecorder2	**-1.24%**	-1.03%	0.01%	-0.24%
	LakeWalking	**-0.08%**	0.00%	0.03%	-0.05%
	ParkSunny	**-1.34%**	-1.34%	–	-0.82%
	ParkWalking	**-0.54%**	-0.32%	–	-0.03%
Other	foreman	**-2.92%**	-2.82%	0.08%	-0.32%
Mean		**-2.46%**	-2.38%	-0.09%	-0.89%

in the published implementation. It is also important to note that the deep architectures were trained on RGB images, whereas HEVC encodes YCbCr frames with a spatial 4:2:0 sampling. The input frames need to be up-sampled (chroma wise) and converted to RGB before being processed by the network, the inverse operation is performed before the HEVC encoding. There is some loss of information during these conversions. Training the networks to process YCbCr 4:2:0 frames should lead to better results.

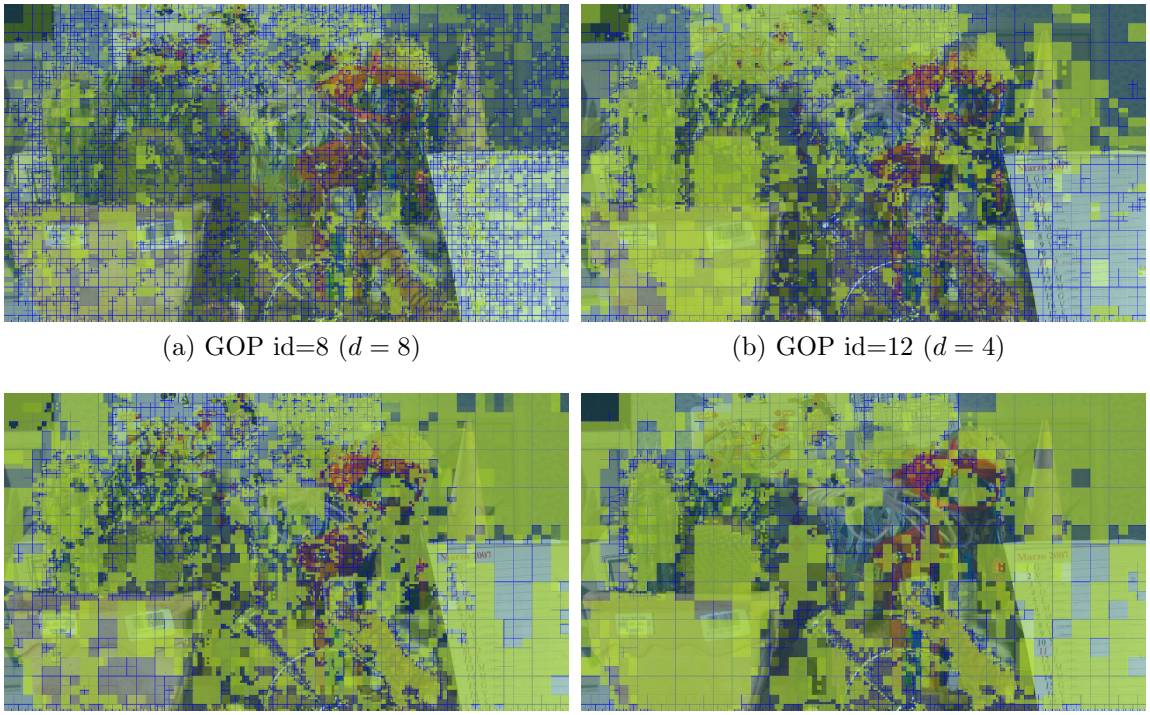
Affine sequences benefit less from the interpolation methods as they mostly display large global scene motions (due to camera zooms or shakes for example). Such motions are more difficult to estimate for these methods, as they were designed and trained for small local motions estimation. An average bit-rate reduction of -0.84% is obtained while homography models based on local descriptors have already been demonstrated to have superior gains, up to -3.31% [22, 23]. It would be interesting to investigate recently proposed deep architectures designed for homography estimation [24, 25].

Cost of signaling

To signal the use of the extra interpolated reference frame, a flag is set at the frame level and the PU level. In order to evaluate the cost of the signaling, we performed the same experiments but disabled the flag coding at the PU level and the global flag

Signaling	Class B	Class C	Class D	Affine
Yes	-3.93%	-2.60%	-2.62%	-0.86%
No	-4.15%	-2.90%	-3.00%	-1.04%

Table 2: BD-rate results comparison with and without signaling of the extra reference use at the prediction unit level. BD-rates were measured for the *SepConv-l1* network.



(a) GOP id=8 ($d = 8$)

(b) GOP id=12 ($d = 4$)

(c) GOP id=14 ($d = 2$)

(d) GOP id=15 ($d = 1$)

Figure 2: Examples of references selection on the *Cactus* sequence at QP=22. Blocks in green are encoded from the interpolated reference by the *SepConv-l1* network.

at the frame level. Results are reported in Table 2 on the CTC sequences. Without signaling, increased performance gains are obtained, with respective improvements of -0.22%, -0.29%, -0.38 % and -0.18% for classes B, C, D and affine. This demonstrates that our current CABAC coding of the flags does not add up to a significant part of the gains, although the flag signaling could still be optimized for greater bit-rate reductions.

The highest bit-rate reduction, -7.36%, is obtained on the *"Cactus"* sequence with the *SepConv-l1* network, which can be explained by the large number of small object motions in this sequence. Estimating and compensating efficiently these small local motions require indeed a lot of syntax signaling (quad-tree splitting, motion vectors information, residual). This costly prediction is avoided by interpolating directly the intermediate blocks with the proposed approach. Figure 2 shows, for some frames of the *"Cactus"* sequence, the blocks for which the reference interpolated by the deep

Table 3: Comparison of the mean interpolation times for each method, on the three classes of the CTC sequences [6].

Method	**Platform**	**Interpolation time (seconds)**		
		Class B	Class C	Class D
CeliuFow	CPU	52.53	9.90	2.43
DVF	GPU	–	–	0.13
SepConv	GPU	0.99	0.30	0.10

neural network is selected by the encoder.

When the frame distance between the reference frames is smaller, the network prediction is more accurate and more blocks are encoded with the proposed method. As such the traditional hierarchical coding structure might not benefit best the proposed interpolation method, as a frame distance of 8 or 4 frames is challenging for the currently implemented networks.

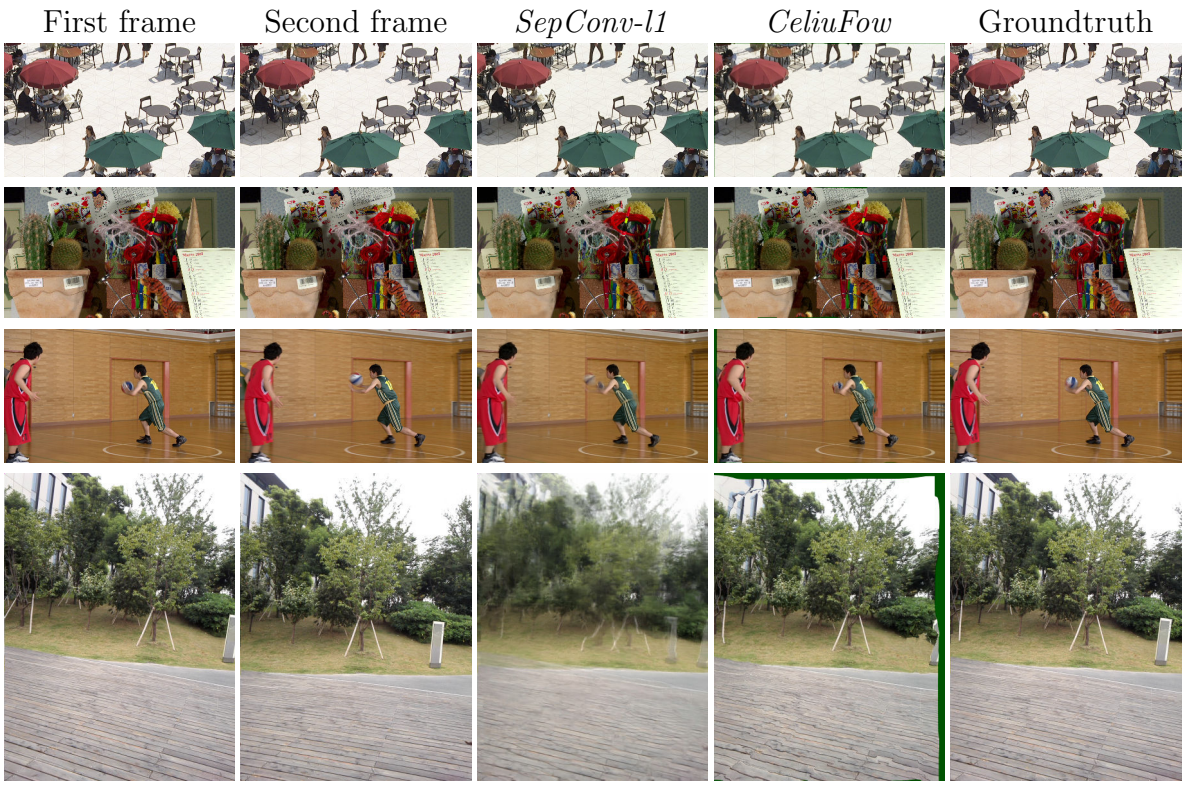
Qualitative results

Qualitative results are presented in Table 4. Interpolated frames are computed for different distances in a group of pictures. This section provides visual results for interpolations obtained for possible GOP distances between reference frames: 2 and 8. Results are shown for four sequences of the test set: *"BQSquare"*, *"Cactus"*, *"BasketBallPass"*, and *"LakeWalking"*.

One may note that small local motions are well interpolated, for example the people walking in *"BQSquare"*, the objects rotating in *"Cactus"* and the players and their ball in *"BasketBallPass"*. The interpolation methods perform better for small GOP distances (1, 2), indeed when the GOP distance between reference frames is too large, blurry outputs are generated. This might be explained by the fact that distant reference frames do not have a motion field that is still locally linear, as assumed by the networks, for example the curved trajectory of the basket ball in the *"BasketBallPass"* sequence. Moreover both methods struggle with the *"LakeWalking"* sequence which displays strong scene motion due to the camera movements (the lack of padding for the *CeliuFow* method is also clearly visible), and also illumination changes.

Complexity

When using deep convolutional networks in conjunction with classical coding architectures, it is difficult to provide a meaningful complexity study as some algorithms are designed to run efficiently on GPU and not on CPU. Instead the interpolation times for each methods are compared for 3 classes from the CTC test sequences (see Table 3). The experiments were run with a single core on an Intel Xeon X5650 CPU, and a Nvidia GeForce GTX 1070 GPU. The *SepConv* approach is at least 25 times faster than the *CeliuFow* method, and about 1.3 times faster that *DVF*. It benefits from running on a GPU, and an efficient CUDA implementation. However the network *SepConv* network requires 21 million parameters, which takes an approximate size of 82 mega-bytes on the disk, and a single forward pass of a 256x256



GOP distance: 2

GOP distance: 8

Table 4: Qualitative evaluation of the interpolation methods. Sequences (top to bottom): *"BQSquare"*, *"Cactus"*, *"BasketBallPass"*, and *"LakeWalking"*.

pixels patch requires around 216 mega-bytes of memory during the execution. This constitute a major drawback for practical hardware implementations of these deep neural networks. However, considering that coming integrated circuit architectures are integrating deep learning capabilities, this may be less of a problem in the coming years.

Conclusion and perspectives

This paper presented a novel inter prediction method based on a deep neural network. The efficiency of different deep architectures was demonstrated against classical methods and the latest HEVC video codec. The prediction accuracy is improved compared to traditional approaches as deep-learning architectures are able to learn a good generalization of motion estimation tasks, with both better modeling and higher spatial resolution. We see this work as a first step in leveraging deep neural networks for frame inter-prediction. The next step is to research, design and train deep interpolation networks specifically in a video compression context, *i.e.* within a rate distortion optimization (RDO) constraint scheme.

A lot of research remains to be performed to improve the design of deep neural networks for video compression. Networks are usually trained, and operate, on 32-bit floating point values. However video codec standards rely on integer operations for the sake of reproducibility. Works have already been proposed for implementing network coefficients quantization [26], such schemes will need to be studied and adapted for video compression. The interpolation filters used by the network could also be tailored for video inter-prediction. A closer integration in the codec would bring improvements thanks to the RDO loop. Finally, the estimated motion range for these deep architectures is limited to small local motions for now, which limits the efficiency on affine sequences. Learning both a global motion and a local motion field constitutes an important perspective.

References

[1] Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala, "Video frame synthesis using deep voxel flow," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 4473–4481.

[2] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive separable convolution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 261–270.

[3] Simon Niklaus and Feng Liu, "Context-aware synthesis for video frame interpolation," *arXiv preprint arXiv:1803.10967*, 2018.

[4] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," *arXiv preprint arXiv:1712.00080*, 2017.

[5] Gary J. Sullivan, Jens-Rainer Ohm, Woojin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[6] Franck Bossen, "Common test conditions and software reference configurations," in *Proc. 12th JVT-VC Meeting, JCTVC-K1100*, Shanghai, CN, October 2012.

[7] Michael Mathieu, Camille Couprie, and Yann LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[8] Joost Van Amersfoort, Anitha Kannan, Marc'Aurelio Ranzato, Arthur Szlam, Du Tran, and Soumith Chintala, "Transformation-based models of video sequences," *arXiv preprint arXiv:1701.08435*, 2017.

[9] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive convolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 670–679.

[10] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell, "Full resolution image compression with recurrent neural networks.," in *CVPR*, 2017, pp. 5435–5443.

[11] Oren Rippel and Lubomir Bourdev, "Real-time adaptive image compression," *arXiv preprint arXiv:1705.05823*, 2017.

[12] 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.

[13] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool, "Generative adversarial networks for extreme learned image compression," *arXiv preprint arXiv:1804.02958*, 2018.

[14] Gregory K. Wallace, "The JPEG still picture compression standard," *Commun. ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[15] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi, "The jpeg2000 still image compression standard," *IEEE Signal Proc. Mag*, 2001.

[16] Jun Han, Salvator Lombardo, Christopher Schroers, and Stephan Mandt, "Deep probabilistic video compression," *arXiv preprint arXiv:1810.02845*, 2018.

[17] Chao-Yuan Wu, Nayan Singhal, and Philipp Krähenbühl, "Video compression through image interpolation," *arXiv preprint arXiv:1804.06919*, 2018.

[18] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand, "Context-based adaptive binary arithmetic coding in the h. 264/avc video compression standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 620–636, 2003.

[19] G. Bjontegaard, "Calculation of average psnr differences between rd-curves," in *ITU-T SG16/Q6 VCEG document VCEG-M33*, Austin, TX, USA, Apr 2001.

[20] Huanbang Chen, Fan Liang, and Sixin Lin, "Affine SKIP and MERGE modes for video coding," in *17th IEEE International Workshop on Multimedia Signal Processing, MMSP 2015, Xiamen, China, October 19-21, 2015*, 2015, pp. 1–5.

[21] Ce Liu et al., *Beyond pixels: exploring new representations and applications for motion analysis*, Ph.D. thesis, Massachusetts Institute of Technology, 2009.

[22] S. Parker, Y. Chen, D. Barker, P. de Rivaz, and D. Mukherjee, "Global and locally adaptive warped motion compensation in video compression," in *IEEE International Conference on Image Processing, ICIP 2017*, 2017.

[23] Jean Begaint, Franck Galpin, Philippe Guillotel, and Christine Guillemot, "Region-based models for motion compensation in video compression," in *2018 Picture Coding Symposium, PCS 2018, San Francisco, CA, USA, June 24-27, 2018*, 2018, pp. 154–158.

[24] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich, "Deep image homography estimation," *arXiv preprint arXiv:1606.03798*, 2016.

[25] Farzan Erlik Nowruzi, Robert Laganiere, and Nathalie Japkowicz, "Homography estimation from image pairs with hierarchical convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 913–920.

[26] Shuang Wu, Guoqi Li, Feng Chen, and Luping Shi, "Training and inference with integers in deep neural networks," *arXiv preprint arXiv:1802.04680*, 2018.