DEEP INTER PREDICTION VIA PIXEL-WISE MOTION ORIENTED REFERENCE GENERATION

*Sifeng Xia, Wenhan Yang, Yueyu Hu, and Jiaying Liu**

Institute of Computer Science and Technology, Peking University, Beijing, China

ABSTRACT

Inter prediction is an important module in video coding for temporal redundancy removal, where the reference blocks are searched from the previously coded frames and employed to predict the block to be coded. However, apart from regular block-wise shift motion, there usually exists inconsistent pixel-wise motion such as rotation and deformation between blocks, which will largely degrade the prediction performance. In this paper, we propose a **M**ultiscale **A**daptive **S**eparable **C**onvolutional **N**eural **N**etwork (MASCNN) to generate pixel-wise closer reference frames for inter prediction. A multiscale network is built to interpolate the target frame from coarse to fine. Reconstruction losses are enforced on each scale to make the network infer the main structure at small scales, which improves the interpolation accuracy of the network. Furthermore, a sum of absolute transformed difference (SATD) loss function is proposed to regularize the network training, which further improves the coding performance. Compared with HEVC, our method can obtain on average 5.7% BD-rate saving and up to 9.9% BD-rate saving for the luma component under the random access configuration.

***Index Terms*—** Inter prediction, frame interpolation, deep learning, video coding

1. INTRODUCTION

Video codecs like MPEG-4 AVC/H.264 [1] and High Efficiency Video Coding (HEVC) [2] exploit temporal similarities among video frames to improve the compression efficiency by the inter prediction module. For a block to be coded (to-be-coded block), the motion estimation technique is used to search reference blocks among the reconstructed frames. Motion compensation technique then obtains the prediction of the to-be-coded block based on reference blocks. After that, only the position of the reference blocks and the residue between the prediction and the original to-be-coded-block need to be coded. Consequently, many bits can be saved.

In inter prediction, the motion estimation process can mitigate the effect of the block level shift motion between the reference blocks and the to-be-coded block. However, in most cases, apart from the block-wise shift motion, there is additional pixel-wise motion like rotation and deformation between blocks. In the motion compensation process of conventional approaches [1, 2], the prediction is derived directly from one individual reference block or a linear combination of the reference blocks. Thus, the pixel-wise motion cannot be fully modeled and will result in a large residue between the prediction and the to-be-coded block, which costs a lot bits for coding.

Some methods [3–5] have explored to model the pixel-wise motion with optical flow. Alshin *et al.* [3, 4] calculated bi-directional optical flow between reference frames to apply pixel-wise motion refinement to motion compensation process. In [5], by estimating bi-directional optical flow between reference frames, a co-located reference frame is interpolated for motion compensation. Although successfully modelling pixel-wise motion to some extent, these methods heavily rely on the accuracy of optical flow estimation. However, owing to the time efficiency requirement in video coding tasks, the optical flow estimation accuracy of the above methods is usually not satisfying.

Recently, some methods introduce deep learning techniques to address motion related problems, *i.e.* optical flow estimation [6–8] and frame interpolation [9, 10]. In [8], a compact but effective optical flow estimation network PWC-Net is designed and outperforms state-of-the-art optical flow estimation methods. Different from image interpolation [11, 12], frame interpolation models the inter-frame motion and exploit temporal redundancies to synthesize the target intermediate frame. In [9, 10], Niklaus *et al.* used the deep convolutional neural network (CNN) to extract motion information between input frames and then generate adaptive kernels to interpolate the intermediate frame. These works have demonstrated the potential of deep learning techniques to model the pixel-wise motion in inter prediction of video coding. Furthermore, Zhao *et al.* [13] explored to directly copy blocks generated by deep frame interpolation as the reconstruction blocks at coding tree unit (CTU) level. Their method obtains BD-rate reduction over HEVC. However, the copy operation usually moves prediction artifacts of frame interpolation to the reconstruction result. Moreover, they directly applied a pretrained video frame interpolation mod-

*Corresponding author. This work was supported in part by National Natural Science Foundation of China under contract No. 61772043, and in part by Beijing Natural Science Foundation under contract No. L182002 and No. 4192025.

el to the video coding scenario without any adjustment or optimization.

In this paper, we propose to use deep learning techniques to generate additional reference samples which have smaller pixel-wise motion to the to-be-coded blocks for motion compensation. For a frame which is to be coded (to-be-coded frame), a pixel-wise closer frame (PC-frame) is interpolated from existing reconstructed reference frames and serves as an additional reference frame. We design a **M**ultiscale **A**daptive **S**eparable **C**onvolutional **N**eural **N**etwork (MASCNN) for the PC-frame generation. A multiscale network is built to ensure the interpolation quality. In order to improve the modeling capacity in the video coding scenario, a sum of absolute transformed difference (SATD) loss function is used. Experimental results show that the coding performance can be significantly improved by additionally using the PC-frames generated by our MASCNN as reference.

The rest of the paper is organized as follows. Sec. 2 introduces the proposed method. We will first briefly illustrate the hierarchical B coding structure where we implement and test our method. Then, the network architecture and the SATD loss function are presented. Details of how to integrate the generated PC-frame into the codec is introduced later. Experimental results are shown in Sec. 3 and concluding remarks are given in Sec. 4.

2. PROPOSED METHOD

2.1. Hierarchical B Coding Structure

We implement and test our method on the HEVC reference software HM-16.15 under the random access (RA) configuration. The size of the group of pictures (GOP) is 16 and the frames are coded in the hierarchical B coding structure, where the coding order of the frames is not decided by their picture order count (POC) value but systematically redesigned. As shown in Fig. 1, frames are assigned to different temporal layers. The frames are coded successively according to their temporal layers. Frames in higher levels can utilize the reconstructed lower level frames as the reference for inter prediction. In addition to frames in the same GOP, coded frames in other GOPs can also be adopted as the reference.

In this paper, we choose to generate the PC-frame for frames whose temporal layers are greater than 1 with available reconstructed reference frames. Specifically, for a to-be-coded frame I_t, we denote its temporal layer as $\tau(I_t)$ and the PC-frame can be generated as follows:

$$I_m = \begin{cases} f\left(\hat{I}_{t-4}, \hat{I}_{t+4}\right), \tau(I_t) = 2, \\ f\left(\hat{I}_{t-2}, \hat{I}_{t+2}\right), \tau(I_t) = 3, \\ f\left(\hat{I}_{t-1}, \hat{I}_{t+1}\right), \tau(I_t) = 4, \end{cases} \quad (1)$$

where I_m is the desired PC-frame and $\hat{I}_{t+*}$ means the reconstructed reference frame in lower temporal layers. $f(\cdot)$ represents the mapping function which infers the PC-frame from two-sided coded reference frames. In this paper, we use MASCNN for inference.

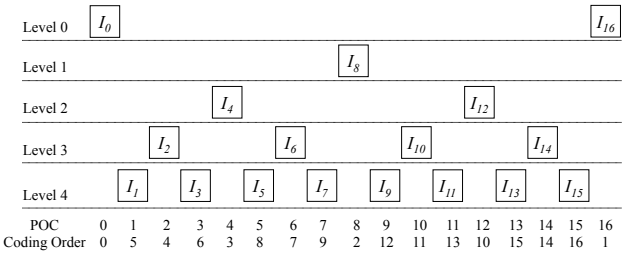

Fig. 1. Illustration of the hierarchical B coding structure in HM-16.15.

2.2. Architecture of MASCNN

For a frame I_t, we uniformly denote its two-sided reference frames as $\hat{I}_l$ and $\hat{I}_r$. With $\hat{I}_l$ and $\hat{I}_r$ as input, MASCNN extracts the bi-directional motion information from them and then generates the PC-frame. The adaptive separable convolution was first proposed in [10], which achieves considerable frame interpolation performance without too much memory consumption. We follow their basic route and construct a more effective network for frame interpolation. In the adaptive separable convolution architecture, the network infers four $1 \times n$ one-dimension filters for each pixel in the target frame I_m. The pixel $I_m(x, y)$ can be obtained by:

$$I_m(x,y) = w_l(x,y) * \hat{P}_l(x,y) + w_r(x,y) * \hat{P}_r(x,y), \quad (2)$$

where $w_l(x,y)$ and $w_r(x,y)$ are $n \times n$ weighting filters. $\hat{P}_l(x,y)$ and $\hat{P}_r(x,y)$ are $n \times n$ patches in $\hat{I}_l$ and $\hat{I}_r$ centered at the position (x, y). The $n \times n$ weighting filters are derived from the one-dimensional vectors by $w_l(x,y) = k'_{l,v}(x,y) * k_{l,h}(x,y)$ and $w_r(x,y) = k'_{r,v}(x,y) * k_{r,h}(x,y)$. In adaptive separable convolution, the three color channels are treated equally and are multiplied by same filters for deriving I_m.

Fig. 2 shows the architecture of MASCNN. An encoder-decoder structure is employed to extract features. The progressive down-sampling and up-sampling operations effectively enlarge the receptive fields. Kernel size of all convolutional layers is set to 3×3 and the rectified linear unit (ReLU) is utilized as the activation function. At the encoder side, average pooling is used for down-sampling. Bilinear interpolation is used for up-sampling at the decoder side. Skip connection is used here to bypass low-level information from the encoder side to the decoder side to accelerate training.

By utilizing the bi-directional motion features of different levels at the decoder side, we interpolate intermediate frames of different scales from coarse to fine. Reconstruction losses can be enforced on the multiscale interpolated frames to make the network at small scales concentrate on inferring the main structure. With the coarse-to-fine architecture, more accurate frame interpolation results can be obtained.

In the adaptive separable kernels estimation unit, four branches are built to separately infer four adaptive separable kernels from the extracted feature map. For a frame of size $H \times W$, four feature maps of size $H \times W \times n$ will be inferred. n is set to 13, 25 and 51 respectively for interpolating frames at scales $1/4$, $1/2$ and 1.

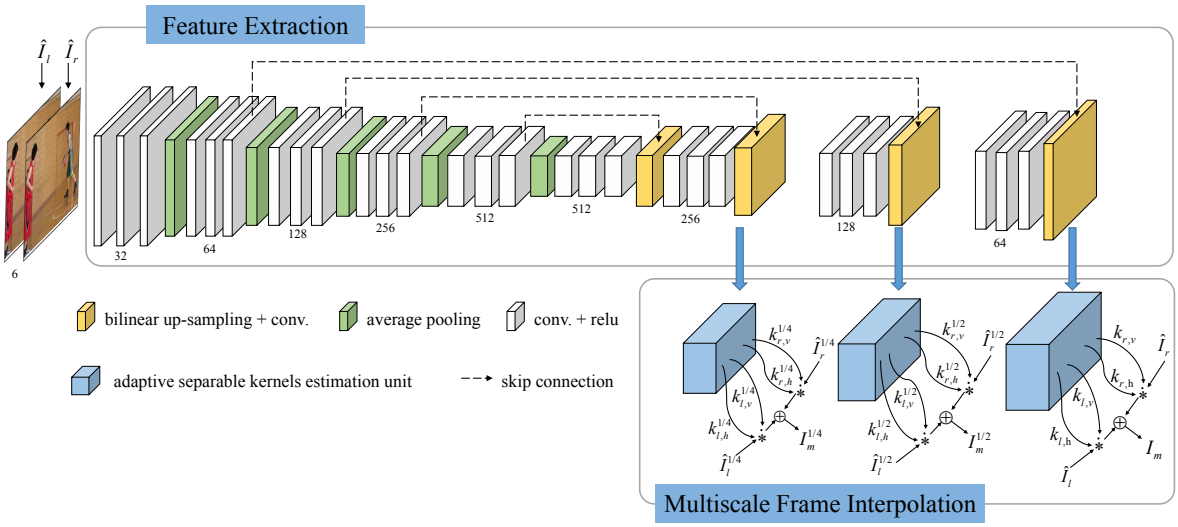

Fig. 2. Architecture of MASCNN. Numbers below the feature maps indicate channel numbers. $1/2$ and $1/4$ mean the relative scale of the filters and images. $\dot{*}$ represents the adaptive separable local convolution.

2.3. Multiscale SATD Loss

In the training process, the network parameters are optimized by back-propagating the gradient of the loss calculated between the interpolated frame I_m and the ground truth I_t. In deep video frame interpolation methods, the ℓ_1 loss function is commonly adopted [9, 10] for training so as to obtain better objective performance. However, ℓ_1 loss function can not fully measure the model capacity from the view of video coding performance since it cannot measure the bits needed for coding the residue.

In the fractional motion estimation process, SATD is adopted as the matching criterion because it is empirically proven that the numerical value of SATD after frequency transformation is more consistent with the number of bits to be spent for residue signals coding. Inspired by this, we choose SATD as the loss function ℓ_S to train our network.

We calculate the ℓ_S loss on 8×8 blocks. By dividing the residue $I_t - I_m$ into T non-overlapping 8×8 residue blocks, we transform each residue block $\mathbf{B}_j$ with the Hadamard transformation matrix $\mathbf{H}$ by:

$$\tilde{\mathbf{B}}_j = \mathbf{H} \times \mathbf{B}_j \times \mathbf{H}, \tag{3}$$

where $\tilde{\mathbf{B}}_j$ is the transformed residue block. Then, $\ell_S(I_m, I_t)$ can be obtained by sum of the absolute values of all the transformed residue signals:

$$\ell_S\left(I_m, I_t\right) = \sum_{j=1}^{T} \sum_{x=1}^{8} \sum_{y=1}^{8} \left| \tilde{\mathbf{B}}_j\left(x, y\right) \right|. \tag{4}$$

The final multiscale loss $\mathcal{L}$ is calculated by:

$$\mathcal{L} = \alpha \ell_S(I_m^{1/4}, I_t^{1/4}) + \beta \ell_S(I_m^{1/2}, I_t^{1/2}) + \gamma \ell_S(I_m, I_t), \tag{5}$$

where α, β, γ are the weighting parameters which are empirically set to $0.2, 0.3, 0.5$. The down-scaled images $I_t^{1/4}$ and $I_t^{1/2}$ are derived from I_t with bilinear interpolation.

2.4. Integration into HEVC

In the coding process of RA configuration, two reference frame lists $List0$ and $List1$ will be maintained. For most frames, 2 frames in $List0$ and 2 frames in $List1$ are available as reference for inter prediction. Reference indexes are allocated to the reference frames so that the prediction units (PU) at the decoder side can find selected reference frames with the coded reference indexes. Directly adding the generated PC-frame to the reference list as the fifth reference frame will cost more bits to code the reference index. Alternatively, we choose an existing reference frame who is temporally farthest from the to-be-coded frame to share its reference index.

We implement a coding unit (CU) level rate distortion optimization (RDO) to decide which reference frame to use if the shared reference index is selected. The PC-frame will be used for reference if the flag is set to true otherwise the original reference frame will be used. All the PUs in the CU share the same flag for indication. If all PUs in a CU do not choose the shared reference index, we will not code the flag since neither the PC-frame nor the original frame are selected as the reference.

3. EXPERIMENTS

3.1. Implementation Details

We use the Vimeo-90K dataset [14] to generate the training data. In total, 103488 samples are generated from the dataset for training the network. We also refer to [10] for data augmentation. Moreover, the two-sided reference frames are additionally coded by HM-16.15 under the all intra configuration with random quantization parameter (QP) values to make the model more robust for the coding artifacts. The network is implemented on PyTorch and AdaMax [15] is used as the optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$. The learning rate is initially set to 0.0001 and changed to 0.00001 after 50 epochs.

Table 1. BD-rate reduction of the proposed method.

Class	Sequence	BD-rate		
		Y	U	V
Class B	Kimono	-4.2%	-5.9%	-4.0%
	BQTerrace	-0.5%	-0.6%	0.1%
	BasketballDrive	-2.7%	-4.5%	-3.5%
	ParkScene	-5.3%	-4.9%	-4.0%
	Cactus	-6.5%	-8.5%	-8.1%
	Average	-3.8%	-4.9%	-3.9%
Class C	BasketballDrill	-4.9%	-7.9%	-7.5%
	BQMall	-9.3%	-11.4%	-11.2%
	PartyScene	-6.3%	-9.9%	-8.0%
	RaceHorsesC	-2.7%	-4.9%	-4.4%
	Average	-5.8%	-8.5%	-7.8%
Class D	BasketballPass	-9.1%	-11.7%	-13.7%
	BlowingBubbles	-6.3%	-9.2%	-7.4%
	BQSquare	-9.9%	-4.5%	-7.3%
	RaceHorses	-5.7%	-8.7%	-9.3%
	Average	-7.8%	-8.5%	-9.4%
All Sequences	Overall	-5.7%	-7.1%	-6.8%

We test our model in HM-16.15 under the RA configuration. BD-rate is used to measure the coding performance. The QP values are set to 27, 32, 37 and 42. It should be noted that we only need to train one model for all QPs. We also compare with a method proposed in [13], which also introduces deep frame interpolation to inter coding but directly use the interpolated block as the reconstruction block. For simplicity, we call it DVRF. Besides, more details about our work can be found on our website[1].

3.2. Results and Analysis

Table 1 shows the overall performance of our method for classes B, C, D. Our method has obtained on average 5.7%, 7.1% and 6.8% BD-rate savings respectively for the Y, U, V components. For the test sequence *BQSquare*, up to 9.9% BD-rate saving can be obtained for the luma component.

In order to validate the effectiveness of the multiscale frame interpolation architecture, we train another deep frame interpolation network ASCNN. It directly estimates the adaptive separable kernels at the original size without the auxiliary multiscale inference and constraint. The comparison results are shown in Table 2. With the multiscale architecture, on average 0.8% BD-rate reduction can be obtained for the luma component.

Moreover, we train MASCNN with ℓ_1 loss function (defined as MASCNN-ℓ_1) to identify the superiority of SATD loss function. The results are also shown in Table 2. By optimizing the parameters with SATD loss function, we can obtain on average 1.1% more BD-rate redcution.

[1]https://pigundermoon.github.io/MASCNN/MASCNN.html

Table 2. Effectiveness validation of multiscale architecture and SATD loss function.

Class	ASCNN	MASCNN-ℓ_1	MASCNN
Class C	-5.3%	-5.3%	-5.8%
Class D	-6.7%	-6.0%	-7.8%
All	-6.0%	-5.7%	-6.8%

Table 3. BD-rate reduction comparison between DVRF and MASCNN.

Class	Sequence	DVRF	MASCNN
Class C	BasketballDrill	-3.2%	-4.8%
	BQMall	-6.0%	-8.4%
	PartyScene	-3.0%	-4.8%
	RaceHorsesC	-0.8%	-2.4%
	Average	-3.2%	-5.1%
Class D	BasketballPass	-5.4%	-9.5%
	BlowingBubbles	-4.1%	-5.1%
	BQSquare	-7.1%	-7.3%
	RaceHorses	-2.2%	-5.9%
	Average	-4.7%	-7.0%
All Sequences	Overall	-4.0%	-6.0%

Furthermore, we compare our MASCNN with DVRF [13], which also uses the deep frame interpolation method to facilitate inter coding. DVRF is implemented on HM-16.6. For fair comparison, we also implement our method on HM-16.6 and test it under the same conditions with DVRF. In the RA configuration of HM-16.6, the GOP size is 8 and the frames are divided into four temporal layers. Following DVRF, we also only handle frames of level 2 and level 3 and directly replace the temporally farthest reference frame without CU level RDO. It should be noted that our method will achieve better performance if we also generate PC-frames for frames of level 1 and integrate the PC-frames by CU level RDO. As shown in Table 3, our method obtains on average 2.0% more BD-rate saving for the luma component compared with DVRF.

4. CONCLUSION

In this paper, we design a pixel-wise motion oriented reference generation deep learning network to improve the inter prediction performance of HEVC. By exploiting the bi-directional motion information between two-sided reference frames, we obtain an additional reference frame which is pixel-wise closer to the to-be-coded frame. A multiscale network is built to make the network learn to predict the main structure of the target frame at small scales so as to improve the prediction accuracy. For model training, SATD loss is used to measure the model capacity from the view of video coding performance. After adding the generated PC-frame under the hierarchical B coding structure, significant BD-rate reduction can be obtained. Extensive experiments identify the effectiveness of each component in MASCNN and demonstrate the superiority of MASCNN to the previous method.

5. REFERENCES

[1] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[2] Gary J. Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[3] Alexander Alshin, Elena Alshina, and Tammy Lee, "Bi-directional optical flow for improving motion compensation," in *Proc. Picture Coding Symposium*, 2010.

[4] Alexander Alshin and Elena Alshina, "Bi-directional optical flow for future video codec," in *Proc. Data Compression Conference*, 2016.

[5] Bohan Li, Jingning Han, and Yaowu Xu, "Co-located reference frame interpolation using optical flow estimation for video compression," in *Proc. Data Compression Conference*, 2018.

[6] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2017.

[7] Anurag Ranjan and Michael J. Black, "Optical flow estimation using a spatial pyramid network," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2017.

[8] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2018.

[9] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive convolution," in *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2017.

[10] Simons Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive separable convolution," in *Proc. IEEE Int'l Conf. Computer Vision*, 2017.

[11] Jie Ren, Jiaying Liu, Wei Bai, and Zongming Guo, "Similarity modulated block estimation for image interpolation," in *Proc. IEEE Int'l Conf. Image Processing*, 2011.

[12] Mading Li, Jiaying Liu, Jie Ren, and Zongming Guo, "Adaptive general scale interpolation based on weighted autoregressive models," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 2, pp. 200–211, 2015.

[13] Lei Zhao, Shanshe Wang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao, "Enhanced ctu-level inter prediction with deep frame rate up-conversion for high efficiency video coding," in *Proc. IEEE Int'l Conf. Image Processing*, 2018.

[14] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman, "Video enhancement with task-oriented flow," *arXiv preprint arXiv:1711.09078*, 2017.

[15] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *Proc. Int'l Conf. Learning Representations*, 2015.