




Deep Reference Generation With Multi-Domain Hierarchical Constraints for Inter Prediction

Jiaying Liu , Senior Member, IEEE, Sifeng Xia , and Wenhan Yang , Member, IEEE

Abstract—Inter prediction is an important module in video coding for temporal redundancy removal, where similar reference blocks are searched from previously coded frames and employed to predict the block to be coded. Although existing video codecs can estimate and compensate for block-level motions, their inter prediction performance is still heavily affected by the remaining inconsistent pixel-wise displacement caused by irregular rotation and deformation. In this paper, we address the problem by proposing a deep frame interpolation network to generate additional reference frames in coding scenarios. First, we summarize the previous adaptive convolutions used for frame interpolation and propose a factorized kernel convolutional network to improve the modeling capacity and simultaneously keep its compact form. Second, to better train this network, multi-domain hierarchical constraints are introduced to regularize the training of our factorized kernel convolutional network. For spatial domain, we use a gradually down-sampled and up-sampled auto-encoder to generate the factorized kernels for frame interpolation at different scales. For quality domain, considering the inconsistent quality of the input frames, the factorized kernel convolution is modulated with quality-related features to learn to exploit more information from high quality frames. For frequency domain, a sum of absolute transformed difference loss that performs frequency transformation is utilized to facilitate network optimization from the view of coding performance. With the well-designed frame interpolation network regularized by multi-domain hierarchical constraints, our method surpasses HEVC on average 3.8% BD-rate saving for the luma component under the random access configuration and also obtains on average 0.83% BD-rate saving over the upcoming VVC.

Index Terms—High efficient video coding (HEVC), inter prediction, frame interpolation, deep learning, multi-domain hierarchical constraints, factorized kernel convolution.

I. INTRODUCTION

WITH the booming multimedia social networking and consumer electronics markets, an increasing amount of images and videos are uploaded to the community everyday. The new trend calls for new coding techniques to further

Manuscript received May 14, 2019; revised October 8, 2019; accepted December 8, 2019. Date of publication December 23, 2019; date of current version September 23, 2020. This work was supported in part by National Natural Science Foundation of China under Contract 61772043, in part by the Beijing Natural Science Foundation under Contract L182002, and in part by Huawei Technologies. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Marta Mrak.

The authors are with the Wangxuan Institute of Computer Technology, Peking University, Beijing 100871, China (e-mail: liujiaying@pku.edu.cn; xsfatpku@pku.edu.cn; yangwenhan@pku.edu.cn).

Color versions of one or more of the figures in this article are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TMM.2019.2961504

improve the compression efficiency. Successive video frames are usually continuous in the temporal dimension and capture the same scene. Therefore, existing video codecs like MPEG-4 AVC/H.264 [1] and High Efficiency Video Coding (HEVC) [2] seek to improve the video coding performance with inter prediction by removing temporal redundancy between video frames. Specifically, in the inter prediction module, for a block which is to be coded (to-be-coded block), the motion estimation technique is first used to search for reference blocks among the reconstructed frames. Then, motion compensation technique predicts the to-be-coded block from reference blocks. After that, only the block-level motion information and the prediction residue between the predicted result and the original to-be-coded-block need to be coded. Consequently, temporal redundancies are largely removed and many bits can be saved.

However, there are lots of obstacles to performing the inter prediction. Even for continuous frames, content changes and complex local motions are quite common, which lead to large residues. Thus, many bits are used to code these residues between the prediction and the to-be-coded block. Many researches are conducted to better estimate global and local motions, namely capturing inter-frame correspondences, for better motion compensation and temporal redundancies removal. The early works [1]–[3] perform *block-level motion estimation and compensation*. In these methods, the prediction is derived directly from one individual reference block or a linear combination of the reference blocks. In real videos, besides block-level translational motion, there exist complex local motions caused by non-translational camera and object movements, which are called *inconsistent pixel-wise displacement*, like rotation and deformation between the matched blocks. These kinds of inconsistent pixel-wise displacement cannot be modeled only by block-level motion estimation and compensation. The residues are still large and cost a lot of bits for coding.

In the upcoming Versatile Video Coding (VVC) [4], some new techniques are proposed to capture more fine-grained motions beyond the block-level translational motion. The affine motion compensated prediction [5] is proposed to model block-level affine motion such as rotation and zooming. Moreover, the bi-directional optical flow (BDOF) tool [6]–[8] is designed for pixel-wise refinement on the bi-directional predicted signal. Illumination consistency along motion trajectory is applied in BDOF estimation. Furthermore, the BDOF is used in [9] for interpolating a co-located reference frame as the additional reference for motion compensation. Although alleviating inconsistent pixel-wise displacement to some extent, the above

hand-crafted methods all model the inter-frame correspondences with limited human priors such as modelling the complex motions as block-level affine motion or calculating BDOF based on illumination consistency of local samples. Therefore, they still cannot fully model different kinds of inconsistent pixel-wise displacement.

Recently, with the rise and development of deep learning-based image processing, some researchers begin to devote their efforts to utilizing deep learning techniques to address motion-related problems, e.g. optical flow estimation [10]–[13] and frame interpolation [14]–[16]. Besides improving the performances in these tasks, these works bring in new insights and methodologies for pixel-level motion modeling, which provide new foundations for the successive works. Meanwhile, more and more works explore to introduce deep learning techniques to the video coding scenario. Various modules in the video coding structure have been touched, e.g., mode decision [17]–[19], intra prediction [20]–[22], inter prediction [23]–[25] and loop filtering [26], [27]. Owing to the powerful representation learning and nonlinear mapping capability of deep learning, significant improvements have been achieved by these deep learning-based video coding methods.

We follow both trends, deep learning-based motion modeling and deep learning-based video coding optimization, and offer optimized video coding techniques to better model the pixel-wise inconsistent displacement. Specifically, we choose to use deep learning techniques to interpolate a pixel-wise closer frame (PC-frame) from existing reconstructed frames. Here, “pixel-wise closer” means that the inconsistent pixel-wise displacement between the interpolated frame and the frame which is to be coded (to-be-coded-frame) is smaller than that between the reconstructed frames and the to-be-coded frame. The PC-frame will be utilized as an additional reference frame for the to-be-coded frame. Thereby reference blocks with smaller pixel-wise displacement may be retrieved for the to-be-coded blocks in inter prediction. Compared to the individual video frame interpolation task, frame interpolation in video coding faces more issues. 1) In the lossy compression, reconstructed reference frames are heavily degraded so less reference information could be used for interpolation. Moreover, the interpolation of the detail will be disturbed by compression artifacts in the prediction process. 2) Coding performance should be considered as the metric for coding-oriented frame interpolation. However, the existing coding pipeline is very complex and not end-to-end trainable. So introducing proper objective functions to train the coding-oriented frame interpolation network is also a great challenge. 3) There are various kinds of dependencies in different domains which can be utilized for PC-frame interpolation, e.g. spatial domain, frequency domain, *et al.* There is not a unified framework to consider these dependencies and their potential interactions jointly.

In our work, we tackle the above issues by building a multi-scale quality attentive factorized kernel convolutional neural network (MQ-FKCNN). The network exploits an encoder-decoder convolutional neural network (CNN) to generate factorized kernels for synthesizing the target frame from compressed frames.

Compared with a single large kernel or separable kernels, the proposed network is both flexible and economic to model video frame signals with factorized kernels. Meanwhile, we introduce multi-domain hierarchical constraints to train the network. 1) To reduce the disturbance of compression noise, we introduce a quality attentive mechanism which guides the network to make choices in the *quality domain* to use more information from high quality frames for inter prediction. 2) For the metric to train such a network, inspired by HEVC, a sum of absolute transformed difference (SATD) loss function that integrates measurements in both *spatial* and *frequency domains* is used. 3) To better utilize dependencies in the *spatial domain* and model *the joint interdependencies across different domains*, our network takes a multi-scale structure to exploit the spatial dependencies and model the multi-domain dependencies in a unified way. Benefiting from our well-designed factorized kernel CNN and the multi-domain hierarchical constraints, the proposed network can be trained not only for better interpolation quality but also greater coding performance.

Our contributions are summarized as follows:

- We propose to utilize deep frame interpolation to generate an additional pixel-wise closer reference frame for inter prediction. A coding-oriented frame interpolation network MQ-FKCNN is specially designed to flexibly synthesize the target frame from input frames with factorized kernels. MQ-FKCNN significantly alleviates the inconsistent pixel-wise displacement between existing reference frames and the to-be-coded frame.
- To better train our network, multi-domain hierarchical constraints are designed for the coding-oriented frame interpolation. The hierarchical dependencies in spatial, quality and frequency domains are considered jointly to obtain more abundant reference information and achieve better interpolation results.
- To additionally deal with compression artifacts, the multi-scale quality attentive mechanism is designed to make the network pick up more information from high quality frames and further exploit spatial dependencies for prediction, which further improves the interpolation accuracy.
- In order to improve the modeling capacity of our network in the video coding scenario, a multi-scale SATD loss function is implemented to guide the network optimization in the joint spatial and frequency domain, which can better indicate the coding cost of the prediction residue and lead to better coding performance.

The rest of the paper is organized as follows. Section II introduces recently proposed deep learning-based methods which solve motion-related problems. Some recent works that use deep learning techniques to improve video coding performance are also presented. Our proposed coding-oriented frame interpolation method will be introduced in Section III. Implementation details about the training data preparation and how to integrate generated PC-frames into HEVC are shared in Section IV. Experimental results and analyses are shown in Section V and concluding remarks are given in Section VI.

II. RELATED WORKS

A. Deep Learning-Based Motion-Related Works

Recently, deep learning-based motion estimation works have been widely proposed and show impressive results compared with traditional methods. In [28], an end-to-end optical flow estimation network FlowNet is first proposed and achieves comparable estimation accuracy with traditional methods. A succeeding network FlowNet 2.0 is later designed to progressively estimate the optical flow and perform on par with state-of-the-art methods at higher frame rates. Hu *et al.* [12] proposed a recurrent spatial pyramid network for effective and efficient optical flow estimation. Recently, Sun *et al.* [13], [29] proposed a compact but effective PWC-Net integrating pyramid processing, warping, and the cost volume. The proposed PWC-Net successfully outperforms previous works on the KITTI benchmark [30].

Meanwhile, some motion-related applications like frame interpolation are also greatly facilitated by deep learning techniques. Niklaus *et al.* [15] formulated video frame interpolation as two steps, *i.e.* motion estimation and pixel synthesis, and they proposed an end-to-end deep learning framework to solve these two tasks. Spatially adaptive kernels are estimated for synthesizing target frames. In [16], adaptive separable kernels are successively proposed to largely reduce the model parameters. Liu *et al.* [31] choosed to directly synthesize the target frame from the input by learning pixel displacement with the network. In [32], flows between the target frame and two input frames are also estimated and utilized for warping. The warped contextual information which is extracted from the response of ResNet-18 [33] is additionally used for blending intermediate frames warped from two-sided input frames. In [34], bi-directional optical flows between input frames are inferred by U-Net [35] and then linearly combined at each time step for interpolating target frames at arbitrary time points.

The successes of all the above deep learning-based methods have identified the ability of deep learning techniques in handling motion related problems. Thus, based on the meaningful experiences of previous methods, we make deeper explorations to estimate the pixel-wise displacement between compressed video frames and generate better temporal reference samples for inter prediction in video coding with deep neural networks.

B. Deep Learning-Based Video Coding

There have been lots of works exploiting deep learning techniques to improve video coding performance by optimizing modules in the coding structure.

CNN has brought significant performance gain to many image restoration tasks like super-resolution [36]–[38], denoising [39] and compression artifacts removal [40], [41]. The success of CNN on these image restoration tasks has promoted the development of deep learning-based loop filtering methods. In [26], Kang *et al.* proposed a multi-modal/multi-scale convolutional neural network to replace existing deblocking filter and sample adaptive offset for loop filtering, which obtains considerable gains over HEVC. A content-aware mechanism [42] is designed to use different CNN models for the adaptive loop filtering in

different regions. In addition to improving loop filtering performance, Laude and Ostermann [43] proposed to replace the conventional Rate Distortion Optimization (RDO) with CNN for the intra prediction mode decision. Furthermore, coding unit (CU) partition mode decision can also be predicted by CNN [17] and Long and Short-Term Memory (LSTM) network [18], [19].

Considering the strong nonlinear mapping ability of deep learning techniques, it is also very promising to predict more reference signals from existing reconstructed signals for intra and inter prediction by deep learning. Li *et al.* [20] firstly adopted fully connected network (FCN) to learn an end-to-end mapping from neighboring reconstructed pixels to the to-be-coded block in the intra coding of HEVC. Moreover, Hu *et al.* [21] used a recurrent neural network to explore the correlations between reconstructed reference pixels and predict the to-be-coded block in a progressive manner. As for inter prediction, Wang *et al.* [24] additionally used spatially neighboring pixels of both reference blocks and current to-be-coded blocks to refine initial predicted blocks with an FCN and a CNN. In [23] and [25], CNNs are used for fractional interpolation in the motion compensation process, which provide better sub-pixel level reference samples for inter prediction. Zhao *et al.* [44] first tried to apply deep frame interpolation to video coding by directly using interpolated blocks as the reconstructed blocks at coding tree unit (CTU) level. Furthermore, Haub *et al.* [45] used a frame extrapolation network to generate artificial reference frames to improve the inter prediction performance under the low delay configuration. They added the generated frame to the reference picture list by replacing one of the existing reference frames for inter prediction. Both these two methods directly use a frame interpolation or extrapolation network for inter coding without any specific optimization for the video coding scenario. So the performance of these methods may be limited. Comparatively, in our work, we carefully analyze new issues faced by frame interpolation in the video coding scenario and specifically design multi-domain hierarchical constraints to improve the robustness and generalization ability of our method in the video coding scenario. Extensive experiments have also shown the superior performance of our method and demonstrated the effectiveness of the specially designed coding oriented modules.

III. PIXEL-WISE CLOSER REFERENCE GENERATION WITH HIERARCHICAL CONSTRAINTS IN MULTIPLE DOMAINS

In this section, we first illustrate the frame interpolation in HEVC and analyze several issues faced in the coding scenario. Then, we build an MQ-FKCNN for deep frame interpolation. At last, we present several well-designed constraints to regularize the training of our MQ-FKCNN to address the above issues for better interpolation.

A. Frame Interpolation in HEVC

We implement and test our method on the HEVC reference software HM-16.15 under the RA configuration. For a to-be-coded frame I_t , two-sided frames I_l and I_r are previously coded and used as the input of MQ-FKCNN. The PC-frame I_m will be interpolated to facilitate inter prediction. In the video

coding scenario, only reconstructed reference frames \hat{I}_l and \hat{I}_r are available for reference. Compared to frame interpolation of high-quality videos, frame interpolation in the coding scenario inevitably faces four issues:

- 1) *High frequency detail loss* of reference frames caused by the quantization operation leads to difficulty in the implicit motion estimation and inaccuracy of local details inference.
- 2) *Compression artifacts*, i.e. the blockiness, in reference frames originate from block-based quantization. The artifacts are easy to be brought into the generated interpolation results.
- 3) *Inconsistent quality of input frames*. Due to the design of coding configurations, I_l and I_r may be coded with different QPs. A desirable frame interpolation model should consider the quality of the input frames and utilize this information adaptively.
- 4) *The purpose of video coding* is to maintain the quality of decoded frames with fewer bits. Thus, training a coding oriented frame interpolation model should pay attention to both distortion and bit cost.

B. Overview of the Proposed Method

To tackle the above mentioned issues, we explore possible models and potential constraints to effectively infer temporally intermediate frames from noisy and inconsistent input frames. To build an effective interpolation model, we start from raw adaptive kernel CNN and separate kernel CNN [16], analyze their correlation with a unified viewpoint, and develop the proposed MQ-FKCNN for better interpolation. To better train this model, the hierarchical constraints in several domains are introduced:

- 1) *Spatial Domain*. Our feature extraction network takes an encoder-decoder structure that first down-samples features and then up-samples features. In this network, the kernels and interpolation results are inferred from small to large progressively. At a small scale, the motion information is easier to be learnt and details can be better inferred in this progressive way even with the high frequency detail loss. Furthermore, at the small scale, compression artifacts are suppressed, and more clean and accurate interpolation results are obtained.
- 2) *Quality Domain*. To handle the inconsistent quality of input frames, we make our model be aware of the quality differences. The factorized kernel is modulated with quality-related features, which guides the model to utilize more information of the high quality input frame.
- 3) *Frequency Domain*. To better regularize the training of our frame interpolation network for better video coding performance, a loss considering both distortion as well as the bit cost is implemented by frequency transformation.

In the following sections, we will introduce our MQ-FKCNN and the multi-domain hierarchical constraints in details.

C. Multi-Scale Quality Attentive Factorized Kernel CNN

For a to-be-coded frame I_t , the frame interpolation method based on adaptive convolutions uses the two-sided reference

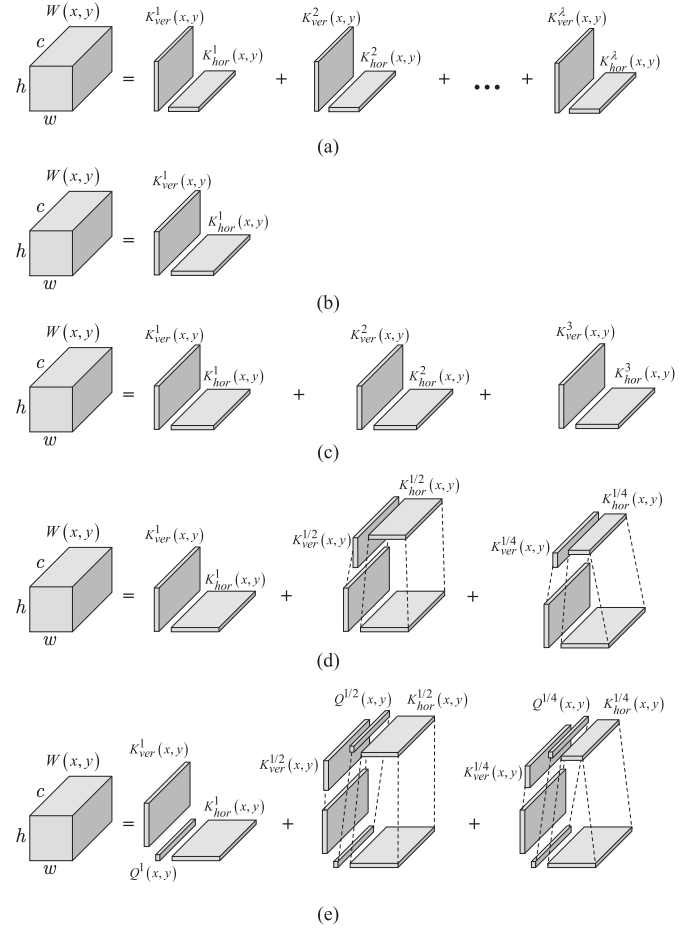


Fig. 1. Architecture of different adaptive kernel models for frame interpolation. (a) Raw adaptive convolution and its factorized form. (b) Adaptive separable convolution. (c) Factorized convolution. (d) Multi-scale factorized convolution. (e) Multi-scale quality attentive factorized convolution.

frames \hat{I}_l and \hat{I}_r as input. The bi-directional motion feature is first extracted and then used for inferring adaptive kernels to reconstruct the temporally intermediate frame. The adaptive convolutions used in previous works, the related variants and our newly proposed one are discussed as follows.

Adaptive Convolution. Adaptive convolution works in this way. To predict a pixel $I_m(x, y)$ in the target frame, two $n \times n$ adaptive 2D kernels $w_l(x, y)$ and $w_r(x, y)$ will be first estimated respectively for two-sided input reference frames \hat{I}_l and \hat{I}_r . $I_m(x, y)$ is then interpolated via local adaptive convolution on \hat{I}_l and \hat{I}_r as follows:

$$I_m(x, y) = w_l(x, y) \otimes \hat{p}_l(x, y) + w_r(x, y) \otimes \hat{p}_r(x, y), \quad (1)$$

where $\hat{p}_l(x, y)$ 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) . For better illustration in the following parts, we first introduce a factorized form of the adaptive convolution as shown in Fig. 1(a). We concatenate the 2D kernels $w_l(x, y)$ and $w_r(x, y)$ together to form a 3D adaptive kernel $W(x, y)$ for each pixel $I_m(x, y)$. We assume the size of $W(x, y)$ to be $c \times h \times w$, where h and w represent heights and weights of the adaptive kernel and are set equally to n . c belongs to the

temporal dimension and corresponds to the number of input reference frames, which is set to 2 through the whole paper. Then, the kernel can be respectively factorized along the temporal dimension as follows:

$$W(x, y, \theta) = \sum_{i=1}^{\lambda} K_{ver}^i(x, y, \theta)' * K_{hor}^i(x, y, \theta), \quad (2)$$

where θ indicates the sequence number of the input reference frame, and λ is the rank number of the adaptive kernel. $K_{ver}^i(x, y, \theta)$ and $K_{hor}^i(x, y, \theta)$ are a pair of separable vertical and horizontal kernels of size $1 \times n$ and the $n \times n$ adaptive kernel $W(x, y, \theta)$ can be derived by summing up the multiplicative results of λ pairs of separable kernels.

Adaptive Separable Convolution. The raw adaptive convolution with a large kernel size leads to a huge amount of parameters, which makes the model training less promising. The adaptive separable convolution [16] addresses the problem by estimating the separable form of the convolutions. In fact, the adaptive separable convolution can be viewed as the special case of the factorized form of the adaptive convolution when $\lambda = 1$, as shown in Fig. 1(b). For each pixel $I_m(x, y)$ in the target frame, four $1 \times n$ one-dimensional kernels $K_{ver}(x, y, 1)$, $K_{hor}(x, y, 1)$, $K_{ver}(x, y, 2)$ and $K_{hor}(x, y, 2)$ will be first estimated. Then, two $n \times n$ adaptive kernels $W(x, y, 1)$ and $W(x, y, 2)$ are obtained by $W(x, y, 1) = K_{ver}(x, y, 1)' * K_{hor}(x, y, 1)$ and $W(x, y, 2) = K_{ver}(x, y, 2)' * K_{hor}(x, y, 2)$. Promising frame interpolation results can be achieved by estimating the adaptive separable kernels.

Factorized Kernel Convolution. When we relax the approximate rank number λ and set λ to an intermediate value, we can get the convolutions with different number of model parameters and modeling capacities, as shown in Fig. 1(c) with $\lambda = 3$.

Multi-Scale Factorized Kernel Convolution. After the quantization operation, lots of high-frequency signals in input reference frames will be lost especially under high QPs. Thus, it is difficult to predict high-frequency details in the target frame with the degraded input frames. Furthermore, the high-frequency prediction error will additionally mislead the training of the network. It is intuitive that high-frequency signals will take a smaller part at lower scales. In this case, prediction quality of the main structures will be considered more in the training phase. Consequently, this kind of negative effects brought by the coding artifacts can be alleviated by the down-scaling operation. By constraining the interpolation process at small scales, the main structures of the target frame are better learned and the frame interpolation quality can be further improved.

In conjunction with multi-scale frame interpolation, we project the factorized kernels to different scales and build a multi-scale factorized kernel convolution as shown in Fig. 1(d) as follows:

$$W(x, y, \theta) = \sum_{\{s=1, \frac{1}{2}, \frac{1}{4}\}} K_{ver}^s(x, y, \theta)' * K_{hor}^s(x, y, \theta), \quad (3)$$

where s represents the scale of the factorized kernel. Here, the representation of the adaptive kernel $W(x, y)$ is not realized by kernel-wise summation and is equivalently injected into the frame generation process. That is, the separate kernels are directly used to interpolate the target frame successively at different scales and are combined by the fusion of the synthesized frames of different scales.

Specifically, for each scale, the target pixel is synthesized by:

$$I_m^s(x, y) = \sum_{\theta=1}^c K_{ver}^s(x, y, \theta)' * K_{hor}^s(x, y, \theta) \otimes \hat{P}^s(x, y, \theta) + \tilde{I}_m^{s/2}(x, y), \quad (4)$$

where I_m^s is the target frame and s represents the corresponding scale of $1/4, 1/2$ or 1 . $\hat{P}^s(x, y, \theta)$ is the reference patch centered at the position (x, y) and will be down-sampled with bilinear interpolation for scales $1/4$ and $1/2$. $\tilde{I}_m^{s/2}(x, y)$ is obtained by doubly up-sampling the previously interpolated frame $I_m^{s/2}$ with bilinear interpolation and it will be set to 0 for $s = 1/4$.

Multi-Scale Quality Attentive Factorized Kernel Convolution. As mentioned above, under the RA configuration, two-sided reference frames will be of different quality since they are coded with different QPs. It is meaningful to pay more attention to the reference frame of higher quality. Consequently, the quality attentive mechanism is introduced to the factorized kernel convolution and a new quality attentive kernel $Q^s(x, y)$ of size $c \times 1 \times 1$ is added. The quality attentive modulation as shown in Fig. 1(e) is formulated as follows,

$$W(x, y, \theta) = \sum_{\{s=1, \frac{1}{2}, \frac{1}{4}\}} Q^s(x, y, \theta) * K_{ver}^s(x, y, \theta)' * K_{hor}^s(x, y, \theta). \quad (5)$$

From the view of frame interpolation, an illustration of the target frame synthesis that uses quality attentive factorized kernel convolution is shown as the QA-FKC component in Fig. 2. We generate normalized quantization parameter (QP) maps \hat{I}_l^Q and \hat{I}_r^Q of the two-sided reference frames as the additional input to make our network more aware of the quality differences between reference frames. Two-sided quality attentive kernels and factorized kernels are estimated for synthesizing the target frame.

The target pixel $I_m^s(x, y)$ is obtained by:

$$I_m^s(x, y) = \sum_{\theta=1}^c Q^s(x, y, \theta) * (K_{ver}^s(x, y, \theta)' * K_{hor}^s(x, y, \theta)) \otimes \hat{P}^s(x, y, \theta) + \tilde{I}_m^{s/2}(x, y). \quad (6)$$

D. Architecture of MQ-FKCNN

The architecture of MQ-FKCNN is shown in Fig. 2. The whole pipeline is illustrated in details as follows.

Bi-Directional Motion Feature Extraction. An encoder-decoder structure is employed to extract bidirectional motion feature. The progressive down-sampling and up-sampling operations effectively enlarge the receptive fields so large scale

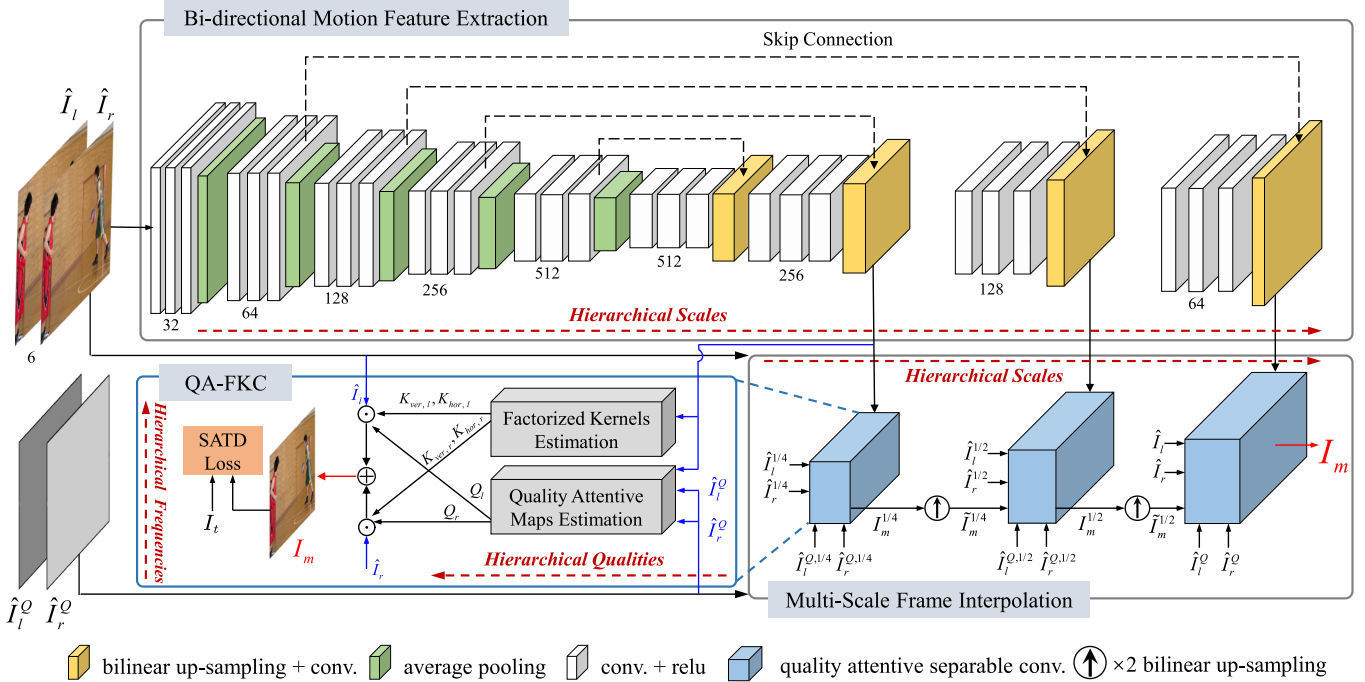


Fig. 2. Architecture of MQ-FKCNN. Numbers below the feature maps indicate channel numbers. 1/2 and 1/4 mean scales of the images. The feature extraction part predicts bi-directional motion information between input frames. In the multi-scale frame interpolation component, intermediate frames of different scales are interpolated by QA-FKC. QA-FKC denotes quality attentive factorized kernel convolution. Inputs of this block are all marked in blue and the outputted I_m is marked in red. The convolution is modulated with quality-related features to be aware of using more information from high quality frames. SATD loss measures the difference in both spatial and frequency domains. The whole network is regularized by the constraints in the spatial, frequency and quality domains.

motion can also be caught by MQ-FKCNN. Kernel sizes of all convolutional layers are 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. Following [16], bilinear interpolation is used for up-sampling at the decoder side since checkerboard artifacts can occur if other up-sampling layers like transposed convolution is selected [46]. Skip connections are used here to bypass low-level information from the encoder side to the decoder side.

Multi-Scale Frame Interpolation. With the extracted bi-directional motion feature, the multi-scale frame interpolation part generates target intermediate frames of different scales from small to large at the decoder side. At each scale s , the target intermediate frame is interpolated by quality attentive factorized kernel convolution.

Quality Attentive Factorized Kernel Convolution. Details of factorized kernels estimation have been described in Section III-C. At each scale s , two-sided factorized kernels $K_{v,l}^s, K_{h,l}^s, K_{v,r}^s, K_{h,r}^s$ and quality attentive maps Q_l^s, Q_r^s are estimated for interpolation. Feature maps of corresponding scales extracted in the feature extraction part are used as input. For a target frame of size $H \times W$, each of four factorized kernel maps of size $H \times W \times n$ will be inferred by four layers of convolutions. For scales of 1/4, 1/2 and 1, n is respectively set to 13, 25 and 51. Thereby, each pixel in the target frame I_m^s can find four corresponding $1 \times n$ factorized kernels at the same position of four factorized kernel maps.

TABLE I
SUMMARIZATION OF MODULES AND THE CORRESPONDING CONSTRAINTS

Module	Constrained Domain	Explanation
Bi-directional motion feature extraction	Spatial	Multi-scale encoder decoder
Multi-scale frame interpolation	Spatial, quality, frequency	It integrates all parts to get results
Factorized kernel estimation	Spatial	As shown in Fig. 1 (e)
Quality attentive maps estimation	Quality	Make the network aware of the quality differences
SATD loss	Spatial, frequency	The signal difference after a frequency transformation

As for the quality attentive maps estimation, normalized QP maps of reference frames are generated and used as the input together with the extracted bi-directional motion feature. The normalized QP maps are derived by dividing QPs of reference frames with the value 51. Then, an $H \times W \times 2$ quality attentive map is estimated by four layers of convolutions. The interpolated result I_m^s is obtained by quality attentive factorized kernel convolution on the reference frames as illustrated in Eq. (6).

After multi-scale frame interpolation, the multi-scale SATD loss function is used to measure the prediction error and guide optimization of network parameters, which is illustrated in Section III-E. The corresponding components in our network which implement the multi-domain hierarchical constraints are summarized in Table I.

Residue				Transformed				
1	1	1	1		16	0	0	0
1	1	1	1		0	0	0	0
1	1	1	1		0	0	0	0
1	1	1	1		0	0	0	0

(a) $\ell_1 = 16, \ell_S = 16$

Residue				Transformed				
1	-1	1	-1		4	0	0	4
-1	-1	1	1		-4	-8	0	4
1	1	-1	1		0	4	-4	8
1	1	1	-1		0	4	4	0

(b) $\ell_1 = 16, \ell_S = 48$

Fig. 3. Two example residue blocks with same ℓ_1 losses but different ℓ_S losses. Intuitively, SATD loss is superior in measuring the redundancy of the residual signal after transform.

E. Multi-Scale SATD Loss Function

In the training process, parameters of the network are optimized by back-propagating the gradient of the loss calculated between the interpolated frame I_m and ground truth I_t . In deep video frame interpolation methods, the ℓ_1 loss function is commonly adopted [16], [31], [32], [34] to train the model for the order of better objective performance:

$$\ell_1(I_m, I_t) = \|I_m - I_t\|_1. \quad (7)$$

However, ℓ_1 loss function cannot fully measure the modeling capacity from the view of video coding performance. It regards each pixel as an independent one and thus cannot measure the bits needed for coding the prediction residue, which is the other important factor that affects the final coding performance.

In the fractional motion estimation process of HEVC, SATD is adopted as a matching criterion for it can better indicate the rate distortion costs, which is also reported in previous works [47], [48]. By computing the SATD of a residue block with Hadamard transformation, parts of the spatial correlations are considered, which is more consistent with the step of transform coding. Thus, SATD can better reflect the rate distortion costs of the residual signals. Two example residue blocks and the corresponding 4×4 Hadamard transformed blocks are shown in Fig. 3. Though their ℓ_1 losses are the same, the residue block (a) will intuitively cost less in the successive coding process since there are higher spatial similarities among the residual signals in the block. Compared with ℓ_1 , SATD successfully reflects the difference of the coding cost.

Consequently, we adopt SATD as the loss function ℓ_S to apply constraints to MQ-FKCNN in the frequency domain for better coding performance. In conjunction with the hierarchical prediction architecture, multi-scale SATD loss function is further

calculated to constrain the prediction process from coarse to fine in the frequency domain.

We calculate the ℓ_S loss by 8×8 blocks. The 8×8 Hadamard transformation matrix \mathbf{H} is defined as follows:

$$\mathbf{H} = \begin{bmatrix} 1, & 1, & 1, & 1, & 1, & 1, & 1, & 1. \\ 1, & -1, & 1, & -1, & 1, & -1, & 1, & -1. \\ 1, & 1, & -1, & -1, & 1, & 1, & -1, & -1. \\ 1, & -1, & -1, & 1, & 1, & -1, & -1, & 1. \\ 1, & 1, & 1, & 1, & -1, & -1, & -1, & -1. \\ 1, & -1, & 1, & -1, & -1, & 1, & -1, & 1. \\ 1, & 1, & -1, & -1, & -1, & -1, & 1, & 1. \\ 1, & -1, & -1, & 1, & -1, & 1, & 1, & -1. \end{bmatrix} \quad (8)$$

By dividing the residue $I_t - I_m$ into T non-overlapping 8×8 residue blocks, we transform each residue block \mathbf{B}_j by:

$$\tilde{\mathbf{B}}_j = \mathbf{H} \times \mathbf{B}_j \times \mathbf{H}, \quad (9)$$

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 residual signals:

$$\ell_S(I_m, I_t) = \sum_{j=1}^T \sum_{x=1}^8 \sum_{y=1}^8 |\tilde{\mathbf{B}}_j(x, y)|. \quad (10)$$

The final multi-scale loss \mathcal{L} is calculated by:

$$\mathcal{L} = \alpha \ell_S(I_m^{1/4}, I_t^{1/4}) + \beta \ell_S(I_m^{1/2}, I_t^{1/2}) + \gamma \ell_S(I_m, I_t), \quad (11)$$

where α, β, γ are the weighting parameters. The down-scaled images $I_t^{1/4}$ and $I_t^{1/2}$ are derived from I_t with Bilinear interpolation.

IV. TRAINING AND INTEGRATION DETAILS OF MQ-FKCNN

A. Training Data Preparation

We use the Vimeo-90 K dataset [49] to generate training data. The dataset consists of 89,800 video clips with a fixed resolution of 448×256 resized from high-quality video frames. Video clips with three frames are sampled from the dataset to form training samples. We annotate three consecutive frames in each clip as I_l , I_t and I_r , where I_l and I_r are the two-sided reference frames and I_t is the ground truth. In the video coding scenario, two-sided reference frames are reconstructed frames which suffer from coding artifacts. The frame quality may be low especially for high QPs. In order to make the network work well in this condition, we code the reference frames I_l and I_r and use the reconstructed frames \hat{I}_l and \hat{I}_r as the input in training data generation. The reference frames are coded with HM-16.15 under the all intra configuration with a random QP value ranging from 0 to 51.

Besides, frames are coded under different QPs in RA configuration, which means two-sided reference frames usually have different quality. For the sake of further simulating the real application situation, we set QPs of two-sided reference frames to have a random difference of 0 to 10. QPs of the reference frames

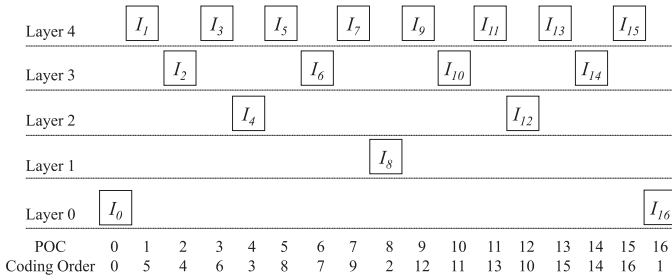


Fig. 4. Illustration of the hierarchical B coding structure in HM-16.15.

are also saved in the training set as the side information for training. With the quality attentive mechanism, our MQ-FKCNN can be more aware of the quality difference between reference frames and learn to interpolate higher quality frames.

Later on, we randomly extract blocks with a size of 150×150 pixels at the same positions from two-sided coded reference frames and the ground truth frame to form training samples. Furthermore, a deep learning-based optical flow estimation method SpyNet [11] is utilized here for candidate samples selection. We will not add samples in which the mean flow magnitudes between two-sided blocks are greater than 15 pixels to the training set.

We divide the dataset into two subsets to respectively form the training and validation dataset. After the selection, in total 234,192 samples will be generated from the dataset for training.

B. Training Procedure

In the training procedure, we refer to [16] for training data augmentation. Specifically, 128×128 patches are randomly cropped from the 150×150 blocks in the training dataset for training. The cropped patches are further augmented by randomly changing the order of two-sided reference blocks. Moreover, we also randomly flip all the input blocks horizontally or vertically for augmentation.

The network is implemented in PyTorch and AdaMax [50] is used as the optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The batch size is set to 16. Weighting parameters α , β , γ in Eq. (11) of the multi-scale SATD loss function are respectively set to 0.2, 0.3, 0.5. The learning rate is initially set to 0.001 and changed to 0.0001 after 30 epochs. We end the training procedure when 70 epochs reach.

C. Integration into HEVC

We implement and test our method on HM-16.15 under the RA configuration, where frames are coded in the hierarchical B coding structure. Frames are allocated to different group of pictures (GOP) and frames of different GOPs are coded successively. In HM-16.15, each GOP consists of 16 frames. The coding order of frames in the same GOP is not decided by their picture order count (POC) value but systematically redesigned. As shown in Fig. 4, frames are assigned to different temporal layers. The frames are coded successively according to their temporal layers. Frames in higher layers can utilize the reconstructed frames in lower layers for inter prediction. Moreover,

in addition to frames of the same GOP, coded frames in previous GOPs can also be adopted as the reference.

We choose to generate the PC-frame for frames whose temporal layers are greater than 1 in this paper. 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:

$$I_m = \begin{cases} f(\hat{I}_{t-4}, \hat{I}_{t+4}), \tau(I_t) = 2, \\ f(\hat{I}_{t-2}, \hat{I}_{t+2}), \tau(I_t) = 3, \\ f(\hat{I}_{t-1}, \hat{I}_{t+1}), \tau(I_t) = 4, \end{cases} \quad (12)$$

where I_m is the desired PC-frame and \hat{I}_{t+*} means the reconstructed reference frame. $f(\cdot)$ represents MQ-FKCNN which infers the PC-frame from two-sided coded reference frames.

In the coding process, two reference picture lists $List0$ and $List1$ will be maintained. For most frames, two forward frames in $List0$ and two backward frames in $List1$ are available as reference for inter prediction. For each reference frame, the reference frame index will be allocated to it which indicates its place in the reference picture list. Prediction units (PU) at the decoder side can find corresponding reference frames through decoded reference frame indexes. To add the interpolated PC-frame to reference picture lists, we choose an existing reference frame \hat{I}_f in reference lists which is farthest from the to-be-coded frame I_t and use its reference index to access I_m at the decoder side.

\hat{I}_f and I_m share the reference index of \hat{I}_f in inter prediction. Specifically, we implement a CU level RDO to decide which reference frame to be accessed by the shared reference index. Two passes of encoding that respectively use \hat{I}_f and I_m for inter prediction are performed at the decoder side. A flag is set based on the rate-distortion costs of the two passes to indicate which reference frame to be used. When the flag is set to true, I_m will be accessed if the shared reference index is chosen. Otherwise \hat{I}_f will be used. The flag is coded with one bit and integrated at CU level. All PUs in a CU share the same flag. Moreover, if all PUs in a CU do not choose the shared reference index after two passes of encoding, we will not code the flag since it is no need to indicate which frame the shared reference index points to if it is never visited.

V. EXPERIMENTAL RESULTS

A. Experimental Settings

The over all performance is tested in HM-16.15 under the common test conditions [51]. BD-rate is used to measure the coding performance. Apart from normal QPs {22, 27, 32, 37}, we additionally test the overall performance of our method under larger QPs {27, 32, 37, 42}. It should be noted that we only need to train one model for all QPs. Luma and chroma components share the same interpolation model. During testing, the chroma components will be first up-sampled and concatenated with the luma component to form a three-channel YUV image. The YUV image is then transformed to an RGB image to form the input. We also compare with a method proposed in [44], which similarly introduces deep frame interpolation to video coding but

TABLE II
BD-RATE REDUCTION OF THE PROPOSED METHOD COMPARED TO HEVC

Class	Sequence	BD-rate (normal QPs)			BD-rate (larger QPs)		
		Y	U	V	Y	U	V
Class A	Traffic	-4.5%	-3.5%	-2.9%	-5.4%	-4.8%	-3.6%
	PeopleOnStreet	-8.8%	-11.3%	-10.0%	-10.9%	-14.0%	-12.3%
	Nebuta	-2.0%	-9.1%	-3.2%	-2.2%	-13.9%	-5.2%
	SteamLocomotive	-1.4%	-1.3%	-1.7%	-1.0%	-1.8%	-1.6%
	Average	-4.2%	-6.3%	-4.4%	-4.9%	-8.6%	-5.7%
Class B	Kimono	-3.6%	-4.3%	-3.1%	-4.2%	-5.1%	-3.9%
	ParkScene	-3.5%	-4.1%	-3.1%	-4.0%	-4.5%	-3.5%
	Cactus	-5.5%	-6.5%	-5.6%	-5.2%	-7.7%	-6.9%
	BasketballDrive	-2.6%	-4.1%	-3.8%	-3.6%	-5.6%	-5.1%
	BQTerrace	-1.9%	-1.9%	-0.6%	-1.6%	-1.7%	-0.8%
	Average	-3.4%	-4.2%	-3.2%	-3.7%	-4.9%	-4.0%
Class C	BasketballDrill	-3.4%	-7.1%	-6.8%	-4.8%	-9.8%	-9.4%
	BQMall	-5.8%	-6.8%	-6.6%	-7.1%	-8.5%	-8.3%
	PartyScene	-6.3%	-8.6%	-7.3%	-6.2%	-9.9%	-7.6%
	RaceHorsesC	-1.5%	-3.0%	-3.4%	-2.4%	-4.8%	-5.1%
	Average	-4.2%	-6.4%	-6.0%	-5.1%	-8.2%	-7.6%
Class D	BasketballPass	-6.2%	-10.4%	-9.4%	-8.1%	-12.6%	-11.4%
	BQSquare	-8.6%	-4.4%	-6.6%	-6.5%	-3.4%	-5.5%
	BlowingBubbles	-5.8%	-7.4%	-6.7%	-5.6%	-7.9%	-6.8%
	RaceHorses	-3.8%	-6.8%	-6.5%	-5.3%	-9.1%	-8.4%
	Average	-6.1%	-7.3%	-7.3%	-6.4%	-8.2%	-8.0%
Class F	BasketballDrillText	-3.1%	-6.6%	-6.1%	-4.5%	-9.9%	-9.0%
	ChinaSpeed	-0.7%	-1.9%	-1.9%	-1.2%	-3.7%	-3.3%
	SlideEditing	0.1%	0.0%	-0.1%	0.1%	0.0%	-0.3%
	SlideShow	-0.8%	-1.8%	-2.1%	-0.7%	-3.5%	-3.9%
	Average	-1.1%	-2.6%	-2.5%	-1.6%	-4.3%	-4.1%
All Sequences	Overall	-3.8%	-5.3%	-4.6%	-4.3%	-6.8%	-5.8%

directly use the interpolated block as the reconstruction block. For simplicity, we call it DVRF.

B. Experimental Results and Analysis

1) *Overall Performance*: Table II shows the overall performance of our method for classes A, B, C, D and F. Our method has obtained on average 3.8%, 5.3% and 4.6% BD-rate savings respectively for the Y, U, V components under the normal QPs and 4.3%, 6.8% and 5.8% BD-rate savings under larger QPs. For the test sequence *PeopleOnStreet*, up to 10.9% BD-rate saving can be obtained for the luma component under larger QPs. For further verification, some example rate-distortion (R-D) curves are shown in Fig. 5.

It should be noted that, sequences in class F are screen content sequences, where most of the displacement is simple shift motion. Consequently, our method cannot bring much BD-rate decrease for sequences in class F and even lead to BD-rate increase for the sequence *SlideEditing*, which significantly affect the overall coding performance. On the other hand, our method successfully obtain significant gain over other sequences with complex motions, which effectively identify the performance of our method.

The computational complexity of the proposed method is provided in Table III. A GTX 1080TI GPU is used for network forwarding. Under the normal QPs, the overall encoding time complexity is 231% and decoding time complexity is 16346%. In our current implementation, we did not make any effort in

TABLE III
COMPUTATIONAL COMPLEXITY OF THE PROPOSED METHOD

Class	Complexity (normal QPs)		Complexity (Larger QPs)	
	Enc (%)	Dec(%)	Enc (%)	Dec(%)
Class A	184%	3797%	187%	4513%
Class B	180%	7002%	201%	8203%
Class C	232%	26820%	243%	31946%
Class D	402%	96649%	446%	116085%
Class F	227%	20934%	232%	23007%
Overall	231%	16346%	246%	19121%

reducing computational complexity. We could further make endeavors in improving the computational efficiency. First, the encoding complexity could be greatly reduced by removing the RDO process realized by CU-level two-pass coding. Second, we could apply some deep model quantization [52] techniques to simplify our network so as to reduce both encoding and decoding time complexity. Furthermore, our model can be implemented on chips with specially designed hardware acceleration techniques e.g. Single Instruction Multiple Data (SIMD), for lower time complexity in real applications.

Besides, the JVET ultra high definition (UHD) test sequences of size 3840×2160 [53] are also tested. Results are shown in Table IV. It should be noted that in the following experiments, all the sequences are tested under QPs {27, 32, 37, 42} and the number of encoded frames is set to be twice of the frame rate. Our method can still obtain considerable BD-rate reduction for

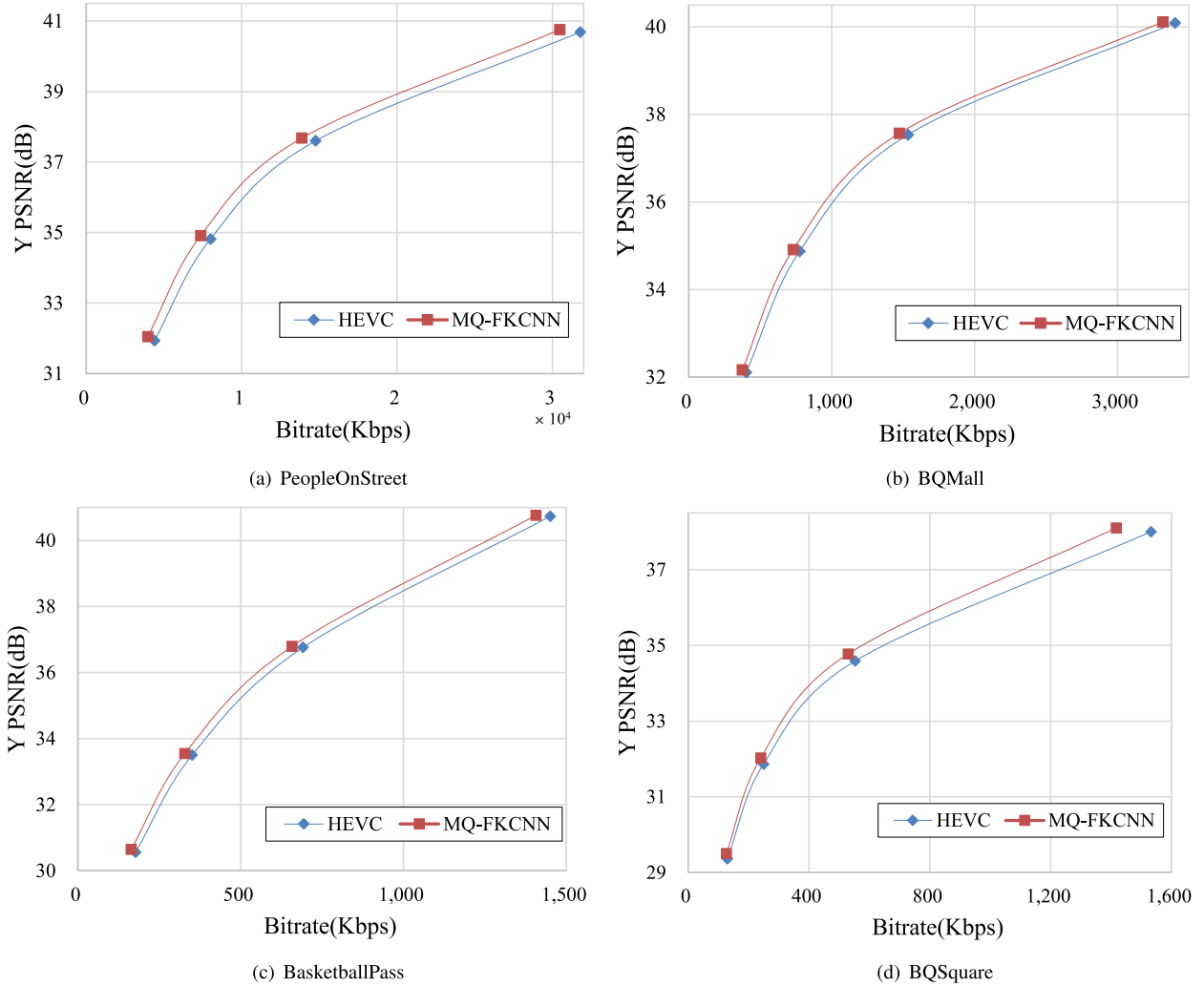


Fig. 5. Four example R-D curves of the sequences *PeopleOnStreet*, *BQMall*, *BasketballPass* and *BQSquare* for the luma component under RA configuration.

TABLE IV
BD-RATE REDUCTION OF THE PROPOSED METHOD ON UHD SEQUENCES

Class	Sequence	BD-rate		
		Y	U	V
Class A1	Traffic	-4.8%	-5.4%	-7.4%
	FoodMarket4	-2.2%	-4.2%	-3.2%
	Campfire	-0.8%	-2.1%	-3.8%
	Average	-2.6%	-3.9%	-4.8%
Class A2	CatRobot	-6.8%	-11.7%	-11.0%
	DaylightRoad2	-2.9%	-2.7%	-2.3%
	ParkRunning3	-1.6%	-1.5%	-1.5%
	Average	-3.8%	-5.3%	-5.0%
All Sequences	Overall	-3.2%	-4.6%	-4.9%

the UHD sequences where on average 3.17% and up to 6.77% BD-rate savings on the luma component can be observed.

2) *Comparison With the Existing Method*: We compare our MQ-FKCNN with DVRF [44], which introduces a deep frame interpolation method to video coding. DVRF is implemented on HM-16.6. For a fair comparison, we also implement our method on HM-16.6 and test our method under the same conditions as

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 deal with frames of layer 2 and layer 3 and directly replace the temporally farthest reference frame without CU level RDO. It should be noted that only results of sequences in classes B, C and D are posted by [44]. So we also test the same sequences for comparison.

As shown in Table V, though DVRF obtains gain over HEVC, they use a pre-trained model without any consideration on the video coding scenario, whose performance is limited. Moreover, directly utilizing generated blocks as the reconstructed blocks cannot fully exploit the benefits of frame interpolation and will bring prediction errors to the following coding process. Differently, by specially designing our model in the video coding scenario and integrating the generated PC-frame into inter prediction, our method obtains on average 2.5% more BD-rate saving for the luma component compared with DVRF.

3) *Comparison With VVC*: Furthermore, we integrate our method into VVC Test Model (VTM-3.0) without the CU level RDO. The BD-rate reduction brought by our method over VVC is shown in Table VI. Although a bunch of techniques such as

TABLE V
BD-RATE REDUCTION COMPARISON BETWEEN DVRF AND MQ-FKCNN

Class	Sequence	DVRF	Ours
Class B	Kimono	-1.7%	-4.7%
	BQTerrace	-0.2%	-0.3%
	BasketballDrive	-1.1%	-2.7%
	ParkScene	-2.6%	-5.3%
	Cactus	-4.6%	-6.1%
	Average	-2.0%	-3.8%
Class C	BasketballDrill	-3.2%	-5.6%
	BQMall	-6.0%	-10.1%
	PartyScene	-3.0%	-6.3%
	RaceHorsesC	-0.8%	-2.0%
	Average	-3.2%	-6.0%
Class D	BasketballPass	-5.4%	-9.9%
	BlowingBubbles	-4.1%	-6.0%
	BQSquare	-7.1%	-9.0%
	RaceHorses	-2.2%	-6.0%
	Average	-4.7%	-7.7%
All Sequences	Overall	-3.2%	-5.7%

TABLE VI
BD-RATE REDUCTION OF THE PROPOSED METHOD COMPARED TO VVC

Class	BD-rate (normal QPs)			BD-rate (larger QPs)		
	Y	U	V	Y	U	V
Class A1	-0.17%	-0.89%	-1.10%	-0.44%	-1.31%	-1.29%
Class A2	-0.59%	-2.71%	-2.31%	-1.09%	-3.16%	-2.71%
Class B	-0.31%	-1.79%	-1.69%	-0.93%	-2.57%	-2.46%
Class C	-2.16%	-3.70%	-3.64%	-3.23%	-5.59%	-5.36%
Class D	-3.08%	-4.62%	-5.03%	-3.63%	-5.58%	-5.82%
Class F	-0.44%	-1.42%	-1.23%	-0.99%	-2.96%	-3.18%
All	-0.83%	-2.30%	-2.21%	-1.48%	-3.24%	-3.05%

affine motion compensated prediction and BDOF have been proposed for improving inter prediction performance. Our method can still obtain BD-rate reduction over VVC. Under larger QPs, on average -1.48% BD-rate reduction can be obtained for the luma component.

4) *Verification of Multi-Domain Hierarchical Constraints:* The effectiveness of multi-domain hierarchical constraints is also verified. A network named FKCNN is first implemented without the quality attentive mechanism and multi-scale frame interpolation. Q-FKCNN is later trained by adding the quality attentive mechanism to FKCNN to verify the quality attentive mechanism. Both FKCNN and Q-FKCNN are trained with the same settings as MQ-FKCNN. The effectiveness of the hierarchical constraints can be proven by comparing between Q-FKCNN and MQ-FKCNN.

The comparison results of different networks on all the sequences are shown in Table VII. It can be seen that a considerable BD-rate reduction can be obtained by adding the quality attentive mechanism to FKCNN. We additionally visualize the fusion weighting maps of the two-sided synthesized results for further verification of our quality attentive mechanism. The weighting maps are generated by dividing the synthesized results from left and right reference frames with the interpolated frame, which indicate the proportion each reference frame takes in the final result. The visualization results are shown in Fig. 6. As we can see, reference frames of higher quality usually take

TABLE VII
BD-RATE REDUCTION COMPARISON FOR THE VERIFICATION OF MULTI-DOMAIN HIERARCHICAL CONSTRAINTS

Class	FKCNN	Q-FKCNN	MQ-FKCNN
Class A	-3.9%	-4.6%	-4.8%
Class B	-2.6%	-3.0%	-3.2%
Class C	-5.1%	-5.3%	-5.6%
Class D	-5.7%	-5.7%	-7.1%
Class F	-1.1%	-1.2%	-1.2%
Overall	-3.6%	-3.9%	-4.3%

a larger proportion in the final results. Moreover, the greater the quality difference is, the more proportion the higher quality one will obtain.

By comparing between Q-FKCNN and MQ-FKCNN, we can find that on average 0.4% BD-rate reduction can be obtained by employing the hierarchical constraints, which brings more multi-domain dependencies for reference and leads to more accurate prediction results.

5) *Verification of SATD Loss Function:* The superiority of ℓ_S loss function is also proven by experiments. We additionally use ℓ_1 and ℓ_2 loss functions to train the network under the same training configuration. Models trained with ℓ_1 , ℓ_2 and ℓ_S losses are respectively denoted as MQ-FKCNN- ℓ_1 , MQ-FKCNN- ℓ_2 and MQ-FKCNN- ℓ_S . Table VIII shows the BD-rate reduction obtained by models trained with different loss functions. By additionally constraining the interpolation in frequency domain, we can obtain on average 1.0% and 0.4% more BD-rate reduction for the luma component on all the sequences compared with models trained with ℓ_2 and ℓ_1 loss functions.

For the chroma components, BD-rate increase is unfortunately observed. Our explanation is that: 1) The smooth chroma signals are originally easier to be coded after transformation. Thus, it may bring more BD-rate savings for the chroma components if we pay more attention to their prediction accuracy. 2) However, since the luma component plays a much greater role in video compression, we believe that the BD-rate increase in chroma components could be ignored considering the BD-rate savings on the luma component brought by the SATD loss function.

6) *Rate Distortion Optimization and CU Partition Results Analysis:* For further verification of the proposed method, we analyze the RDO and CU partition results corresponding to the coding results shown in Table II. It is worth nothing that only frames whose temporal layers are greater than 1 are covered in our analysis. We first calculate the ratio of the CUs that choose PC-frames for inter prediction. RDO results of all the classes are shown in Table IX. It can be seen that PC-frames generated by our MQ-FKCNN are adopted by a considerable number of CUs for inter prediction.

Intuitively, more larger CUs will be used if we successfully alleviate the inconsistent pixel-wise displacement, since it is no need to further divide the CUs to handle the local differences caused by pixel-wise displacement. So we further analyze changes of the CU partition results before and after using the

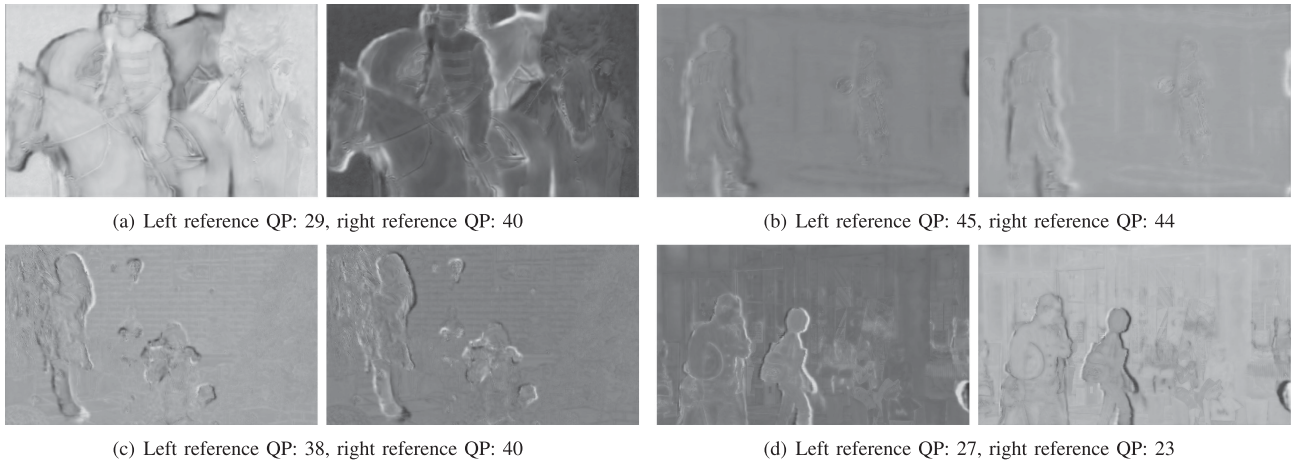


Fig. 6. Visualization examples of the weighting maps which indicate the proportion different reference frames take in the target frame interpolation. Brighter pixels mean higher weightings.

TABLE VIII
BD-RATE REDUCTION COMPARISON BETWEEN MODELS TRAINED WITH DIFFERENT LOSS FUNCTIONS

Class	MQ-FKCNN- ℓ_2			MQ-FKCNN- ℓ_1			MQ-FKCNN- ℓ_S		
	Y	U	V	Y	U	V	Y	U	V
Class A	-4.0%	-7.5%	-5.1%	-4.7%	-9.2%	-6.1%	-4.8%	-8.4%	-5.3%
Class B	-2.7%	-3.8%	-3.0%	-2.9%	-4.7%	-3.6%	-3.2%	-4.5%	-3.3%
Class C	-4.5%	-7.0%	-6.6%	-5.2%	-9.1%	-8.6%	-5.6%	-8.9%	-8.3%
Class D	-4.7%	-6.6%	-6.7%	-6.0%	-8.7%	-9.4%	-7.1%	-8.3%	-8.4%
Class F	-1.0%	-2.6%	-2.4%	-1.1%	-3.6%	-3.5%	-1.2%	-3.8%	-3.7%
All	-3.3%	-5.4%	-4.7%	-3.9%	-6.9%	-6.1%	-4.3%	-6.7%	-5.7%

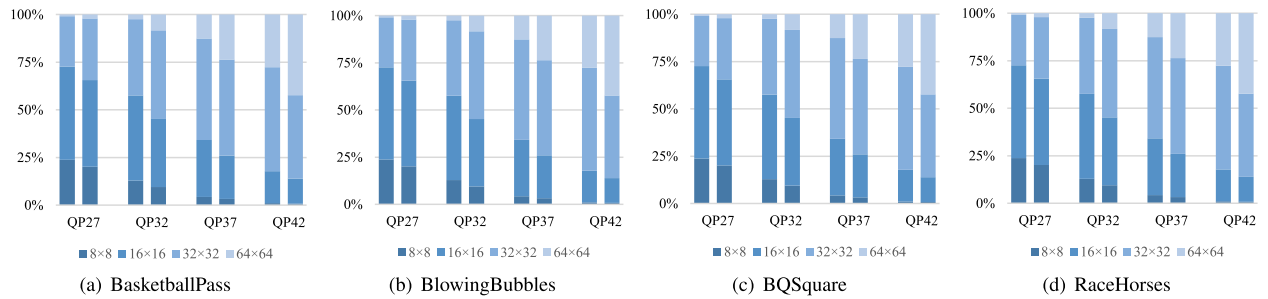


Fig. 7. Changes of the CU partition before and after using generated PC-frames for inter prediction. In each set, the left one shows ratios of different types of pixels coded by HM and the right one shows ratios of the pixels coded by our proposed method.

TABLE IX
RATIOS OF CUS THAT CHOOSE PC-FRAMES FOR INTER PREDICTION UNDER DIFFERENT QPS

Class	Choosing Ratio				
	22	27	32	37	42
Class A	40.0%	38.8%	37.9%	36.3%	35.1%
Class B	32.3%	28.2%	33.3%	36.4%	34.2%
Class C	26.0%	27.7%	33.0%	37.3%	37.8%
Class D	37.9%	40.8%	47.6%	49.8%	40.5%
Class F	8.4%	10.9%	14.6%	18.9%	20.1%
All Sequences	29.1%	29.2%	33.3%	35.8%	33.6%

generated PC-frames. We divide pixels into four types according to sizes of the CUs they belong to. Later, ratios of different types of pixels are calculated and shown in Fig. 7. It can be found that more larger CUs have been used for inter prediction after adding PC-frames to the reference lists.

7) *Results Under LD Configuration*: Furthermore, to test the generality of the proposed method, we also test our method under the low delay (LDB) configuration. We additionally train MQ-FKCNN for the LDB configuration on newly prepared training data. Video clips containing three consecutive frames are used to form the training samples. In each clip, the first two frames are used to form the input and the third frame is used as the target.

TABLE X
BD-RATE REDUCTION UNDER THE LDB CONFIGURATION

Class	Sequence	BD-rate		
		Y	U	V
Class B	Kimono	-2.4%	-8.1%	-3.0%
	ParkScene	-1.7%	-3.5%	-2.6%
	Cactus	-3.1%	-7.3%	-5.9%
	BasketballDrive	-0.2%	-0.8%	-0.7%
	BQTerrace	-0.2%	-1.1%	-1.9%
	Average	-1.5%	-4.2%	-2.8%
Class C	BasketballDrill	-2.1%	-9.4%	-6.9%
	BQMall	-6.5%	-11.1%	-10.7%
	PartyScene	-3.3%	-9.9%	-7.8%
	RaceHorsesC	-0.6%	-1.1%	-0.8%
	Average	-3.1%	-7.9%	-6.6%
Class D	BasketballPass	-4.0%	-8.5%	-6.7%
	BlowingBubbles	-3.1%	-7.2%	-9.4%
	BQSquare	-3.3%	-6.6%	-3.4%
	RaceHorses	-0.6%	-1.2%	-1.3%
	Average	-2.7%	-5.9%	-5.2%
Class E	FourPeople	-6.9%	-8.5%	-5.9%
	Johnny	-3.7%	-3.0%	-2.6%
	KristenAndSara	-4.9%	-7.3%	-5.0%
	Average	-5.2%	-6.3%	-4.5%
Class F	BasketballDrillText	-1.0%	-5.8%	-3.4%
	ChinaSpeed	-0.3%	-5.6%	-2.4%
	SlideEditing	0.2%	0.1%	0.1%
	SlideShow	-0.1%	-5.0%	-3.8%
	Average	-0.3%	-4.1%	-2.4%
All Sequences	Overall	-2.4%	-5.6%	-4.2%

Testing results on all the sequences are shown in Table X. On average 2.4% BD-rate reduction can be obtained for the luma component under the LDB configuration.

VI. CONCLUSION

In this paper, we propose a deep learning based frame interpolation method to improve the inter prediction performance of HEVC. We carefully analyze the difficulties of frame interpolation encountered in the video coding scenario and pertinently propose the MQ-FKCNN based frame interpolation regularized by multi-domain hierarchical constraints. The multi-scale quality attentive factorized kernel convolution is implemented to interpolate the target frame from small to large with quality attention. For the training of MQ-FKCNN, multi-scale SATD loss function is employed to guide the network optimization in both spatial and frequency domains, which further improves the coding performance. After adding the generated PC-frames under the hierarchical B coding structure, significant BD-rate reduction can be obtained. Extensive experiments identify the effectiveness of each component in our MQ-FKCNN and demonstrate the superiority of MQ-FKCNN to the previous method.

REFERENCES

- [1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
- [2] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
- [3] X. Jing and L.-P. Chau, "An efficient three-step search algorithm for block motion estimation," *IEEE Trans. Multimedia*, vol. 6, no. 3, pp. 435–438, Jun. 2004.
- [4] B. Bross, J. Chen, and S. Liu, "Versatile video coding (Draft 6)," *Document JVET-O2001*, Jul. 2019.
- [5] S. Lin *et al.*, "Affine transform prediction for next generation video coding," in *Document ITU-T SG16, Doc.COM16-C1016*, Oct. 2015.
- [6] A. Alshin, E. Alshina, and T. Lee, "Bi-directional optical flow for improving motion compensation," in *Proc. Picture Coding Symp.*, 2010, pp. 422–425.
- [7] A. Alshin and E. Alshina, "Bi-directional optical flow for future video codec," in *Proc. Data Compression Conf.*, 2016, pp. 83–90.
- [8] E. Alshina *et al.*, "Known tools performance investigation for next generation video coding," *ITU-T SG16/Q6 Doc. VCEG-AZ05*, Jun. 2015.
- [9] B. Li, J. Han, and Y. Xu, "Co-located reference frame interpolation using optical flow estimation for video compression," in *Proc. Data Compression Conf.*, 2018, pp. 13–22.
- [10] E. Ilg *et al.*, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 1647–1655.
- [11] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2720–2729.
- [12] P. Hu, G. Wang, and Y.-P. Tan, "Recurrent spatial pyramid cnn for optical flow estimation," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2814–2823, Oct. 2018.
- [13] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 8934–8943.
- [14] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, "Phase-based frame interpolation for video," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 1410–1418.
- [15] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2270–2279.
- [16] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 261–270.
- [17] Z. Liu *et al.*, "CU partition mode decision for HEVC hardwired intra encoder using convolution neural network," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5088–5103, Nov. 2016.
- [18] M. Xu *et al.*, "Reducing complexity of HEVC: A deep learning approach," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5044–5059, Oct. 2018.
- [19] J. Xu, M. Xu, Y. Wei, Z. Wang, and Z. Guan, "Fast H.264 to HEVC transcoding: A deep learning method," *IEEE Trans. Multimedia*, vol. 21, no. 7, pp. 1633–1645, Jul. 2019.
- [20] J. Li, B. Li, J. Xu, R. Xiong, and W. Gao, "Fully connected network-based intra prediction for image coding," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3236–3247, Jul. 2018.
- [21] Y. Hu, W. Yang, S. Xia, and J. Liu, "Optimized recurrent network for intra prediction in video coding," in *Proc. IEEE Vis. Commun. Image Process.*, 2018, pp. 1–4.
- [22] J. Pfaff *et al.*, "Intra prediction modes based on neural networks," in *Document JVET-J0037*, Apr. 2018.
- [23] N. Yan, D. Liu, H. Li, and F. Wu, "A convolutional neural network approach for half-pel interpolation in video coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2017, pp. 1–4.
- [24] Y. Wang, X. Fan, C. Jia, D. Zhao, and W. Gao, "Neural network based inter prediction for hevc," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2018, pp. 1–6.
- [25] J. Liu, S. Xia, W. Yang, M. Li, and D. Liu, "One-for-all: Grouped variation network-based fractional interpolation in video coding," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2140–2151, May 2019.
- [26] J. Kang, S. Kim, and K. M. Lee, "Multi-modal/multi-scale convolutional neural network based in-loop filter design for next generation video codec," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 26–30.
- [27] C. Jia *et al.*, "Content-aware convolutional neural network for in-loop filtering in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3343–3356, Jul. 2019.
- [28] A. Dosovitskiy *et al.*, "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 2758–2766.
- [29] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Models matter, so does training: An empirical study of CNNs for optical flow estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, p. 1, 2019.

- [30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The kitti vision benchmark suite," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 3354–3361.
- [31] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 4473–4481.
- [32] S. Niklaus and F. Liu, "Phase-based frame interpolation for video," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 1410–1418.
- [33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 770–778.
- [34] H. Jiang *et al.*, "Super slo-mo: High quality estimation of multiple intermediate frames for video interpolation," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 9000–9008.
- [35] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput. Assisted Intervention*, 2015, pp. 234–241.
- [36] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 184–199.
- [37] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 1646–1654.
- [38] W. Yang *et al.*, "Deep learning for single image super-resolution: A brief review," *IEEE Trans. Multimedia*, vol. 21, no. 12, pp. 3106–3121, Dec. 2019.
- [39] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.
- [40] C. Dong, Y. Deng, C. C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 576–584.
- [41] Y. Dai, D. Liu, and F. Wu, "A convolutional neural network approach for post-processing in HEVC intra coding," in *Proc. Int. Conf. Multimedia Model.*, 2017, pp. 28–39.
- [42] C. Jia *et al.*, "Content-aware convolutional neural network for in-loop filtering in high efficiency video coding," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3343–3356, Jul. 2019.
- [43] T. Laude and J. Ostermann, "Deep learning-based intra prediction mode decision for HEVC," in *Proc. Picture Coding Symp.*, 2016, pp. 1–5.
- [44] L. Zhao *et al.*, "Enhanced CTU-level inter prediction with deep frame rate up-conversion for high efficiency video coding," in *Proc. IEEE Int. Conf. Image Process.*, 2018, pp. 206–210.
- [45] T. Laude, F. Haub, and J. Ostermann, "HEVC inter coding using deep recurrent neural networks and artificial reference pictures," in *Proc. Picture Coding Symposium*, 2019, pp. 1–5.
- [46] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. [Online]. Available: <http://distill.pub/2016/deconv-checkerboard>.
- [47] I. Seidel, L. H. Cancellier, J. L. Gntzel, and L. Agostini, "Rate-constrained successive elimination of Hadamard-based SATDs," in *Proc. IEEE Int. Conf. Image Process.*, 2016, pp. 2395–2399.
- [48] J. Gu, M. Tang, and J. Wen, "SATD based fast intra prediction for HEVC," in *Proc. Data Compression Conf.*, 2017, pp. 442–442.
- [49] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *Int. J. Comput. Vision*, vol. 127, no. 8, pp. 1106–1125, 2019.
- [50] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–15.
- [51] F. Bossen, "Common test conditions and software reference configurations," *Document JCTVC-L1100*, Jan. 2013.
- [52] P. Gysel, J. Pimentel, M. Motamedi, and S. Ghiasi, "Ristretto: A framework for empirical study of resource-efficient inference in convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5784–5789, Nov. 2018.
- [53] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, "JVET common test conditions and software reference configurations for SDR video," *Document JVET-N1010*, Mar. 2019.



Jiaying Liu (S'08–M'10–SM'17) received the Ph.D. degree (Hons.) in computer science from Peking University, Beijing, China, 2010. She is currently an Associate Professor with the Wangxuan Institute of Computer Technology, Peking University. She has authored more than 100 technical articles in refereed journals and proceedings, and holds 42 granted patents. Her current research interests include multimedia signal processing, compression, and computer vision. Dr. Liu is a Senior Member of CCF/CSIG. She was a Visiting Scholar with the University of Southern California, Los Angeles, from 2007 to 2008. She was a Visiting Researcher with the Microsoft Research Asia in 2015 supported by the Star Track Young Faculty Award. She has served as a member of Multimedia Systems & Applications Technical Committee, Visual Signal Processing and Communications Technical Committee and Education and Outreach Technical Committee in IEEE Circuits and Systems Society, and a member of the Image, Video, and Multimedia Technical Committee in APSIPA. She was an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, and Elsevier JVCI. She has also served as the Technical Program Chair of IEEE VCIP-2019/ACM ICMR-2021, the Publicity Chair of IEEE ICME-2020/ICIP-2019/VCIP-2018, and the Area Chair of ECCV-2020/ICCV-2019. She was the APSIPA Distinguished Lecturer (2016–2017).



Sifeng Xia received the B.S. degree in computer science from Peking University, Beijing, China, in 2017. He is currently working toward the master's degree with the Wangxuan Institute of Computer Technology, Peking University. His current research interests include deep learning-based image processing and video coding.



Wenhan Yang (S'17–M'18) received the B.S. degree and Ph.D. degree (Hons.) in computer science from Peking University, Beijing, China, in 2012 and 2018, respectively. He is currently a Postdoctoral Research Fellow with the Department of Computer Science, City University of Hong Kong, Hong Kong. He was a Visiting Scholar with the National University of Singapore, from 2015 to 2016. His current research interests include deep-learning based image processing, bad weather restoration, and related applications and theories.