Video Compression Using Generalized Binary Partitioning and Advanced Techniques for Prediction and Transform Coding

J. Pfaff, H. Schwarz, D. Marpe, *Fellow, IEEE*, B. Bross, S. De-Luxán-Hernández, P. Helle, C. R. Helmrich, *Senior Member, IEEE*, T. Hinz, W. Q. Lim, J. Ma, T. Nguyen, J. Rasch, M. Schäfer, M. Siekmann, G. Venugopal, A. Wieckowski, M. Winken and T. Wiegand, *Fellow, IEEE*

***Abstract*—In this paper, we describe a video coding design that enables a higher coding efficiency than the HEVC standard. The proposed video codec follows the design of block-based hybrid video coding, but includes a number of advanced coding tools. A part of the incorporated advanced concepts was developed by the Joint Video Exploration Team, while others are newly proposed. The key aspects of these newly proposed tools are the following. A video frame is subdivided into rectangles of variable size using a binary partitioning with variable split ratios. Three new approaches for generating spatial intra prediction signals are supported: A line-wise application of conventional intra prediction modes, coupled with a mode-dependent processing order, a region-based template matching prediction method and intra prediction modes based on neural networks. For motion-compensated prediction, a multi-hypothesis mode with more than two motion hypotheses can be used. In transform coding, mode dependent combinations of primary and secondary transforms are applied. Moreover, scalar quantization is replaced by trellis-coded quantization and the entropy coding of the quantized transform coefficients is improved. The intra and inter prediction signals can be filtered using an edge-preserving diffusion filter or a non-linear DCT-based thresholding operation. The video codec includes an adaptive in-loop filter for which one of three classifiers can be chosen on a picture basis. We also incorporated an optional encoder control, which adjusts the quantization parameters based on a perceptually motivated distortion measure. In a random access scenario, our proposed video codec achieves luma BD-rate savings between 32.5% for HDR A and 39.6% for SDR A over the HEVC (HM software) anchor for different categories of test sequences.**

I. Introduction

THIS paper describes a video codec that goes beyond the compression capabilities of Advanced Video Coding (AVC) and High Efficiency Video Coding (HEVC). This video codec has been submitted as a proposal [1] in response to the joint call for proposals (CfP) on video compression technology [2].

Similar to the video coding standards H.264 | AVC [3], [4] and H.265 | HEVC [5], [6], the codec design follows the approach of block-based hybrid video coding. Each video picture is partitioned into blocks and the blocks are predicted by either intra-picture or inter-picture prediction. The prediction error signals are transformed, the resulting transform coefficients are quantized, and the quantized transform coefficients as well as partitioning and prediction parameters are entropy coded. However, for all of these basic building blocks, we included new coding tools that improve the compression performance.

After the finalization of HEVC, experts of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) formed the Joint Video Exploration Team (JVET) with the goal of exploring new technology for future video coding standards. Promising coding tools developed in this activity were integrated into a common software basis, known as the Joint Exploration Model (JEM) [7], [8]. Our proposed video codec includes a significant part of these approaches, but it additionally comprises a number of newly developed coding tools. The main focus of the present paper lies on a description of the proposed new coding technologies. However, throughout the paper, we will also always briefly summarize which JEM tools are integrated. For details on the JEM tools, the reader is referred to [7], [8].

This paper is organized as follows. In Section II, we give a brief overview of all new coding tools. In Section III, we outline the block partitioning scheme. Advanced concepts for intra- and inter-picture prediction are described in Section IV. Section V presents advanced filtering techniques for improving prediction signals. In Section VI, our approach for transform coding of prediction residuals is described. In Section VII, an improved adaptive in-loop filter is outlined and, in Section VIII, we highlight our perceptually motivated encoder control. Finally, experimental results for the proposed codec and individual tools are presented in Section IX.

II. Overview of the main tools

In this section, an overview of the main new coding tools that we integrated into our proposal for the CfP is given:

1) Partitioning with Generalized Binary Splits: Each coding block can be split horizontally or vertically at different locations. Up to five locations are possible at each side. The resulting blocks always have side lengths that are an integer multiple of four.

2) Line-Based Intra Coding: Intra-coded blocks can be split either horizontally or vertically into 1-D lines. The intra-picture prediction and the transform coding of the prediction

All authors are with Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, Germany

H. S. is also with Institute of Computer Science, Free University of Berlin, Germany

T. W. is also with Department of Telecommunication Systems, Technical University of Berlin, Germany

residual are performed on each line separately, but the same intra prediction mode is used for all lines of a block.

3) Intra Region-Based Template Matching: The intra prediction signal for a block is formed by a superposition of three already reconstructed signals on blocks in the same picture. The displacement vectors locating these blocks are not transmitted but are derived through a template matching search algorithm for which only a region index needs to be signaled.

4) Intra Prediction Based on Neural Networks: Intra prediction modes were trained based on a large set of training sequences. These trained modes are used as additional options for generating an intra prediction signal.

5) Multi-Hypothesis Inter Prediction: In inter prediction, it is possible to generate a prediction signal as a superposition of more than two motion-compensated prediction signals. The additional motion information required is either explicitly transmitted or inferred in the merge mode.

6) Signal Adaptive Diffusion Filter: A filtering is applied to both intra- and inter-prediction signals. In order to preserve relevant edges, the filter coefficients can be computed from the initial prediction signal itself and thus be spatially varying.

7) Prediction Refinement via DCT Thresholding: An inital prediction signal is extended by its adjacent reconstructed samples and the extended signal is transformed via a discrete cosine transform (DCT). Transform coefficients beneath a fixed threshold are set to zero. Transforming back yields the refined prediction signal.

8) Adaptive Transform Selection: The intra prediction residual is transformed using one out of five transform candidates. The set of transform candidates depends on the intra prediction mode used. Here, non-separable transforms are allowed which are restricted secondary transforms for large blocks.

9) Trellis-Coded Quantization: The conventional scalar quantization in transform coding of prediction residuals is replaced with trellis-coded quantization, which yields a higher packing density in the high-dimensional signal space.

10) Entropy Coding of Quantized Transform Coefficients: The absolute values of transform coefficient levels are transmitted in a single pass. Thus, it is possible to use neighboring already decoded absolute values for an improved context modeling of a current absolute value. Furthermore, a context model selection that depends on the state of the trellis-coded quantizer's state machine is used for two context-coded bins.

11) Multiple Feature based Adaptive Loop Filter: The adaptive in-loop filter of JEM is extended by two additional classifiers, a rank-based and a sample-value based classifier. The classifier used is transmitted in the bitstream.

12) Perceptually Optimized Encoder Control: In the encoder control, a weighted variant of the sum of squared errors with local signal-adaptive weights can optionally be used as an error measure. If this error measure is used, the encoder decisions are more aligned to perceptual quality metrics while the only change needed in comparison to a conventional encoder control is a local adaptation of the Lagrangian multiplier.

III. Partitioning with Generalized Binary Splits

Modern video codecs usually operate in a block based way. In the HEVC standard and in the JEM, a video frame is initially divided into so-called coding tree units (CTUs). The CTUs cover squares of $N_{\max} \times N_{\max}$ luma samples and are the starting point for a flexible partitioning into smaller blocks. In HEVC, each CTU is partitioned into coding units (CUs) of square shape using a quadtree. On each CU, either intra- or inter-picture prediction is applied, where for the latter, a further rectangular subdivision is possible. For transform coding of prediction residuals, each CU can again be subdivided by a second quadtree. In the JEM, a quadtree plus binary tree (QTBT) splitting [9], [10] is used to partition a CTU into rectangles on which both prediction (intra- or inter-picture prediction) and transform coding of prediction residuals are carried out. Here, each binary split divides a rectangular block horizontally or vertically into two blocks of equal size.

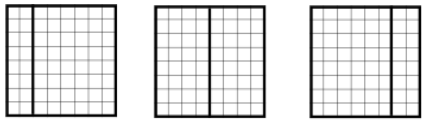


Fig. 1. Examples of vertical $1/4$, $1/2$ and $3/4$ splits.

Our partitioning scheme, called generalized binary splitting (GBS), is an extension of these methods that has a larger flexibility [11]. It also partitions a frame into CTUs which can be split recursively. For the splitting, only binary splits are used. However, in contrast to QTBT, binary splits into blocks of unequal size are supported. More precisely, a given rectangle of width W and height H can be split horizontally into two blocks of width W, where the first block has height αH and the second block has height $(1-\alpha)H$. Here, the split ratio α is chosen out of the set

$$\left\{1/2,\ 1/4,\ 3/4,\ 1/3,\ 2/3,\ 3/8,\ 5/8,\ 1/5,\ 2/5,\ 3/5,\ 4/5\right\}. \qquad (1)$$

Vertical splits are supported analogously. Fig. 1 shows examples of vertical splits with split ratios of $1/4$, $1/2$ and $3/4$.

Not all split ratios are always possible. The availability of a specific split ratio is predetermined by the block shape and the split direction. Here, two fundamental split constraints that reduce the number of possible splits are important: The granularity of the splitting scheme and the prohibition of redundancies. First, for the granularity, we have the constraint that the modified size, i.e., the size after performing a split, has to be a multiple of four. For example, given the set of split ratios (1) and assuming the side to be split has size 32, the available split ratios are

$$\left\{1/2,\ 1/4,\ 3/4,\ 3/8,\ 5/8\right\}, \qquad (2)$$

whereas if the size was 20, the set of available split ratios would be

$$\left\{1/5,\ 2/5,\ 3/5,\ 4/5\right\}. \qquad (3)$$

Second, the redundancy constraints guarantee that the final block partitioning can arise through only one sequence of consecutive splits. In order to illustrate why such constraints are required, we remark that, for example, splitting a block using a one-quarter split followed by a parallel two-third split on the larger subblock yields the same partitioning as if a one-third split would have followed a parallel three-quarter split on

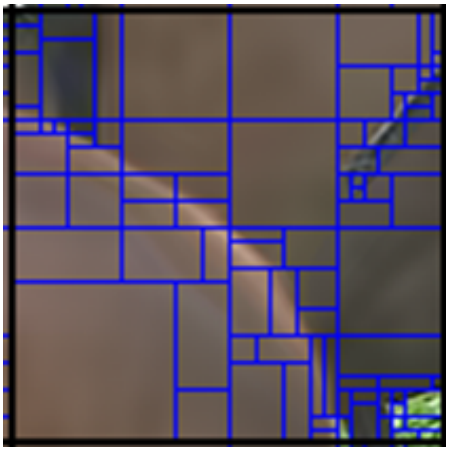


Fig. 2. Examples of a partitioning of a CTU that can be generated by the GBS.

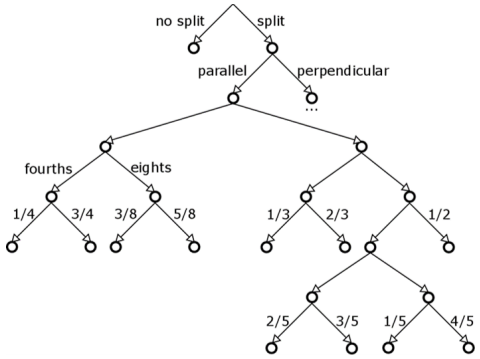


Fig. 3. Illustration of the split syntax for the generalized binary partitioning.

the larger subblock. Another important aspect in the selection of the used split ratios is the trade-off between signaling overhead, search space extension and additional achievable gain. We found that the described selection of split ratios provides a good balance between these aspects.

In general, our splitting scheme yields up to 10 available splits that are possible for each block. As in QTBT, the rectangular blocks that result from the partitioning are used for both prediction and transform coding. In our submission, the CTU size was set to 128. Compared to the partitioning of both HEVC and JEM, the number of partition options for a CTU is significantly increased. Figure 2 shows an example of a partitioning of a CTU that can be generated by the GBS. The split is coded as illustrated in Fig. 3. First, a split flag indicating if a block is further split is transmitted. If the split flag is equal to 1, the split direction and the split ratio are coded. The split direction is signaled as either parallel or perpendicular to the last split. For the first split, i.e., the CTU-level split, perpendicular is defined as a vertical split and parallel as a horizontal split. The split ratio is coded using the binarization shown in Fig. 3. If a binary decision can only take one value due to the restrictions on the block size or the redundancy constraints, the corresponding flag is not coded, but inferred at the decoder side.

The GBS scheme proposed in our CfP response did not include a quad-split. While this ensures a more consistent design, the inclusion of a quad-split into a future version of the partitioner provided additional coding gains [12]. In addition, a clear separation of the splits into coarse and fine partitioning, the first one being solely represented by quad-splits, makes the encoder control significantly easier.

Due to the large number of partitioning options, the selection of splits at an encoder is a challenging tasks. This is a general problem common to all partitioning schemes that provide such a flexible split topology. In fact, large portions of the encoder control for GBS have been successfully ported to the reference software for the upcoming video coding standard VVC [13], although the partitioning scheme of the latter is based on the multi-type-tree approach [14]. For more details on these encoder speedups, the reader is referred to [15].

In the design of the partitioner, different split ratios can be enabled or disabled in the high-level syntax. Specifically, the $1/4$ and $3/4$ (fourths) splits, as well as the $3/8$ and $5/8$ (eights) splits can be disabled. The $x/3$ and $x/5$ splits are complementary to the fourths and eights splits and thus do not need their own high-level switches. In this way, different operation points can be selected encompassing the search complexity and the signaling overhead. Compared to a configuration with only $1/2$ splits, the unrestricted configuration could reach up to 4% more bit-rate rate reduction for about an $10\times$ encoder run time increase. By varying the available split restrictions, a flexible selection of operation points for different use cases is possible.

IV. Intra- and inter-picture prediction

In this section, we discuss our methods for intra- and inter-picture prediction. For intra-picture prediction, we supported all associated tools of the JEM, which are 67 intra prediction modes, 4-tap filter for intra-sample prediction, intra boundary filtering, and intra planar PDPC. Moreover, we used multi-reference-line intra prediction similar as in [16]. In addition to these tools, we supported a line-based intra-prediction mode, region-based template matching and intra prediction modes based on neural networks.

For inter-prediction, all related tools of JEM were supported. These tools are comprised by sub-PU level motion vector derivation, locally adaptive motion vector resolution, 1/16-th luma sample accurate motion vectors, overlapped block motion compensation, local illumination compensation, affine motion-compensated prediction, pattern matched motion vector derivation, decoder-side motion vector refinement and bi-directional optical flow. Additionally, we supported a multi-hypothesis inter-picture prediction mode.

A. Line-Based Intra Coding

The line-based intra coding mode partitions a luma block into 1-D lines. The prediction as well as the transform coding of the prediction residual are then carried out for each line individually where the intra prediction mode is the same for all lines. Here, the reconstructed samples of the previously processed line comprise a part of the input for the intra-prediction on a current line. The motivation for this approach is that due to the loss of correlation between samples with increasing distance, intra prediction across large blocks may lead to residual signals with high levels of energy concentrated

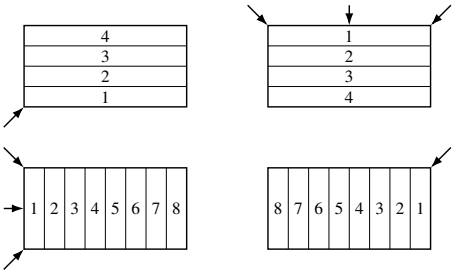


Fig. 4. Line-based intra prediction with horizontal (top) and vertical (bottom) splitting for the example of an 8×4 block. The numbers refer to the processing order and the arrows indicate examples of intra prediction directions.

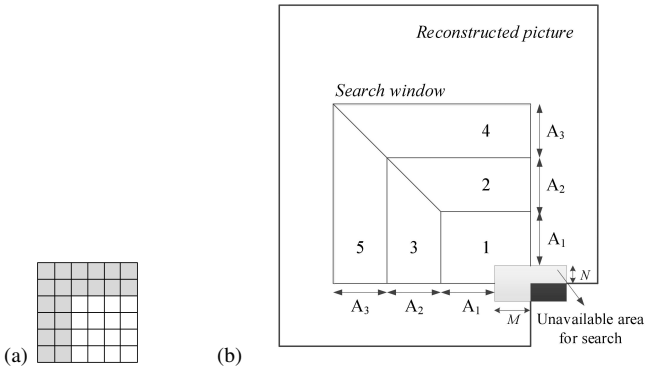


Fig. 5. Intra region-based template matching: (a) Template (grey) around a current block (white); (b) definition of search regions.

in the most distant regions of the block relative to the neighboring reference samples. We refer to [17], [18], [19], [20] as examples for previous work on line based intra coding.

The main new aspects of our approach are that we combine line-based intra coding with all intra prediction modes supported in the JEM as well as with all possible block shapes that arise in our partitioning scheme. Moreover, we introduce different processing orders in which the 1-D blocks are coded. For further details, we refer to [21], [22].

The line-based mode can be applied for luma intra-predicted blocks of all sizes. It divides a $W \times H$ block into W columns or H rows. In order to increase the prediction quality, for each split type two different processing orders are defined. In the normal processing order, one proceeds from left to right for vertical splits and from top to bottom for horizontal splits. In the reversed processing order, one proceeds from right to left for vertical splits and from bottom to top for horizontal splits. For each directional intra prediction mode, the processing order that follows the direction of the intra mode is supported (see Fig. 4). For example, if the intra prediction mode predicts in the diagonal direction coming from the top right, then for a vertical split, the reversed processing order is chosen while for the diagonal direction coming from the top left, the normal processing order is chosen.

All 1-D partitions use a 1-D DCT-II for the transform coding of the prediction residual. The only exception is the case of the planar mode, where a one-dimensional DST-VII is employed. Furthermore, the quantized transform coefficients are coded in the same way as for regular blocks with the following exceptions: First, the context of each coded block flag is the value of the coded block flag of the previously coded line. Second, the last position syntax element requires only one coordinate to be sent to the decoder. Third, the 4×4 coefficient groups degenerate into 1×4 or 4×1 lines. Finally, a vertical line employs a vertical scan and a horizontal line a horizontal one. The line based intra coding mode gives a particular high compression benefit in the case of screen content [21].

B. Intra Region-Based Template Matching

Intra region-based template matching (IRTM) generates a prediction signal for a current block by copying already reconstructed blocks inside the same picture. The location of these blocks is described by integral displacement vectors. Such an approach generally gives a particular high compression benefit for the case of screen content coding. A central part of the method presented here is that the displacement vectors are not explicitly signaled in the bitstream. Instead, they are derived by finding the best match between a template T consisting of reconstructed samples adjacent to a current block and the displaced template, [23], [24], [25]. Since the template matching search can result in an enormously large computational complexity, the search is typically restricted to a window [23], [24], [25]. The key idea of our approach is that it avoids searching a large picture area due to the sub-partitioning of the search window compared to the conventional template matching algorithms. The region to be searched is indicated by an index that is coded in the bitstream. Also, generalizing [24], in our approach, the prediction is comprised by a linear combination of three different reconstructed blocks.

In more detail, our template T_c consists of the reconstructed samples on two lines left and above the block, see Figure 5a. Moreover, as outlined in Figure 5b, five search regions are specified. The sizes of these search regions are parametrized by numbers A_1, A_2, A_3 that depend on the frame width [26]. In our CfP submission, these sizes were set to $8, 24, 144$ for HD and UHD sequences. When generating the prediction signal of the current block, the reconstructed samples in the five search regions are already available to the decoder. Thus, if v is an integral displacement vector pointing to a search region indexed by i, one can form the signal $T_{i,v}$ consisting of all reconstructed samples on the region that arises by displacing the sample positions of the templated T_c by v.

For each of the five search regions indexed by i, with $1 \leq i \leq 5$, let $v_{i,1},\ v_{i,2},\ v_{i,3}$ be the consecutive minima of

$$\mathrm{SSD}(T, T_v) \tag{4}$$

over all displacement vectors v pointing into the search region. Here, SSD denotes the sum of squared differences. The minimum in (4) is taken over all displacement vectors v pointing into the search region and a template T_v is allowed to cross multiple search regions. Moreover, a predefined search algorithm is to be used.

For a displacement vector v pointing into a search region, let $pred_v$ denote the reconstructed samples on the block for which the two lines of reconstructed samples left and above are formed by the template T_v, see Figure 5a. Then,

if displacement vectors $v_{i,1}$, $v_{i,2}$, $v_{i,3}$ for a specific search region are found as described, the overall prediction signal $pred_{i,final}$ corresponding to the i-th search region is given as

$$pred_{i,final} := (2pred_{v_{i,1}} + pred_{v_{i,2}} + pred_{v_{i,3}})/4. \quad (5)$$

If IRTM is used, then the index i of the search region is signaled in the bitstream. Given that index, the prediction $pred_{i,final}$ is generated as in (5).

It is important to note that the search algorithm to solve (4) is part of the specification of the method, since it has to be carried out by encoder and decoder simultaneously. A detailed description of the search algorithm that we used as well as more details on our method can be found in [26] and [27].

As show in [28], for natural content, RTM has more coding gain than the intra block copy (IBC) tool of HEVC Screen Content Coding [29]. However, for screen content, IBC is more efficient than RTM. Due to the template matching search at the decoder side, RTM is more complex at the decoder than IBC.

C. Intra Prediction Based on Neural Networks

In conventional video codecs like HEVC and also in the JEM, the intra prediction signal is generated either by angular prediction or by the DC and planar modes. For further improving the quality of intra-picture prediction, we tried to design more general intra prediction modes as the outcome of a training experiment based on a large set of training data. The concept of these modes is illustrated in Fig. 6. For each rectangular block with M rows and N columns, M and N being integer powers of two between 4 and 32, we supported n prediction modes that were realized by a neural network. The number n is equal to 35 for $\max(M, N) < 32$ and it is equal to 11, otherwise. Here, fewer modes were used for large blocks since the number of weights that need to be stored for each mode increases with the block size. The prediction modes perform the following key steps. Input for the prediction are the $d := 2(M + N + 2)$ reconstructed samples r on the two lines left and above the block as well as the 2×2 corner on the top-left. From these reconstructed samples, a set of features is extracted that can be used for all modes. These features are then used to select an affine linear combination of predefined image patterns as the prediction signal. The features are generated by applying a matrix-vector multiplication, an offset addition and a non-linear activation function three times.

The aforementioned predictors are thus represented by a fully connected network with three hidden layers which are shared by all predictors. The dimension of the hidden layers is equal to the input dimension d for $\max(M, N) \leq 32$ and to $d/2$, otherwise. For each hidden layer, the exponential linear unit [30] is used as an activation function.

In order to signal which of the given n modes is to be applied, a second neural network is used whose input is the same vector of reconstructed samples r as above and whose output is a conditional probability distribution p over the modes, given the reconstructed samples r. Then, an index i is sent in the bitstream indicating that the i-th most probable mode is to be selected. Here, the binarization of i is such that

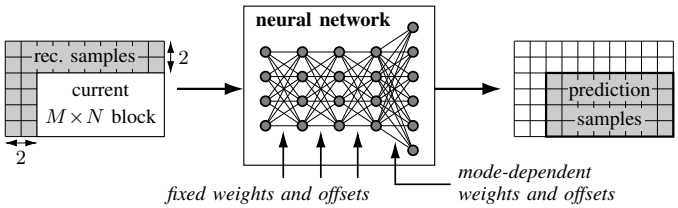


Fig. 6. Intra prediction with neural networks.

small values of i require less bins than large values of i. At the reconstruction stage, the probability mass function p is to be computed which allows to identify the correct mode. For the parsing of the index i itself, p is not needed and thus our signallling approach does not create a parsing dependency.

As already mentioned, the set of all parameters Θ needed to generate the prediction signals as above, i.e., all matrix and bias entries occurring in the prediction and the probability networks, were determined by experiments that used a large set of training data. These training data were disjoint from the sequences used in the CfP and in the JVET common test conditions.

The training algorithm used was based on the minimization of a loss function that attempts to capture two aspects of the overall codings system that surrounds our predictors. The first one is the transform coding of prediction residuals, where zero coefficients play an important role. The second aspect is the partitioning of pictures into blocks and the selection of the specific intra mode for each of them.

For the first aspect, assume that a block of original samples s is predicted by $pred$. Then denote by $c = W(s - pred)$ the transformed prediction residual, where W is the DCT-II 2D basis. If c_i is the i-th coefficient of c, we define

$$l(c) = \sum_i \left(\alpha|c_i| + \beta g(\gamma(|c_i| - 1))\right). \quad (6)$$

Here, g is the logistic function $g(x) = 1/(1 + e^{-x})$ and α, β, γ, and δ are constants that were experimentally determined. The function l quickly decreases for small coefficients, while there is only a minor slope for large coefficients. Thus it shares an important property with the amount of bits spent for coding quantized transform coefficients in typical video codecs where there is an extra benefit for the coding of zero transform coefficients, see, for example, [31].

For the second aspect, the overall loss function takes as input a signal s on a block B_{max} of maximal size 32×32 as well as the parameters Θ of our neural networks. Then, for each subblock B of B_{max}, the best mode k and its costs according to (6) are determined. The signaling costs of this mode are modelled by $-\log_2(p(k))$, where p is the probability mass function that is computed by our second neural network. Then they are added to the costs (6) to give the overall costs on B. For each partitioning of B_{max} into subblocks, the costs of the subblocks are added to a loss corresponding to that partitioning. The overall loss of s and Θ is defined as the minimal loss over all partitionings.

The parameters Θ were determined by attempting to minimize the accumulation of the loss function over a large

set of training data s. Here, we used a stochastic gradient descent approach. In this setting, for a given example only the optimal modes corresponding to the optimal partitioning obtain a gradient update. This algorithm was preceded by an initialization algorithm for the weights Θ.

The neural network based prediction modes were added as complementary to the intra prediction modes of JEM. For the test sequences specified in the CfP, the neural network prediction modes were used for approximately 50% of all intra blocks. For further details on our intra prediction with neural networks, the reader is referred to [32], [33] and [34].

In the paper [35], for every square block occurring in HEVC, two intra prediction modes were trained. In the training process, the clustering is carried out by putting examples that were coded in DC or planar mode into the first cluster and examples that were coded in an angular mode into the second cluster. Here, a fixed HEVC intra encoder is used. The gains reported in [35] were similar to the gains that can be achieved by our intra prediction modes, see [33], [34]. However, the complexity of the prediction modes of [35] is significantly higher than that of our modes.

The main novelties in our approach can be summarized as follows. First, we use a different training that does not use the mean-squared prediction error but the aformentioned more elaborate loss function and which directly invokes the clustering into a variety of block shapes and modes. For the latter clustering, it is also important to model the signaling costs during training and thus to find a way of signaling the modes, which we both tried to do with a second neural network as described above.

As a further new development, in their subsequent work [32], [34], the authors designed the predictors such that they predict into the frequency domain of the DCT where each predictor predicts only certain transform coefficients (independent of the input). All other frequency components are always inferred to be zero. This design significantly reduces the complexity of the prediction modes, in particular for large blocks, where more than three quarters of all DCT-coefficients are predicted to be always zero. Thus, in the last layer of the network, which contributes most to the complexity of the intra prediction modes, more than three quarters of all multiplications can be saved which reduces the encoder and decoder runtime overhead caused by the method.

D. Multi-Hypothesis Inter Prediction

Multi-hypothesis inter prediction refers to the generation of an inter-picture prediction signal by linearly superimposing more than one motion-compensated signals, called hypotheses. Theoretical investigations [36], [37] as well as practical implementations [38] have shown that this approach can improve the performance of inter-picture prediction. In both the HEVC standard and the JEM, the maximal number of hypotheses allowed is restricted to two. In that context, inter prediction with two hypotheses is called bi-prediction, while inter prediction with a single hypothesis is called uni-prediction.

Thus, if $p_{uni/bi}$ is the inter-prediction signal that arises by the conventional uni- or bi-prediction, in the case of multi-hypothesis inter prediction, an additional motion compensated prediction signal h_3 is used such that the overall inter-prediction signal p_3 is given as

$$p_3 := (1-\alpha)p_{uni/bi} + \alpha h_3. \tag{7}$$

Here, α is a predefined weighting factor given as $\alpha = 1/4$ or $\alpha = -1/8$. The above process can be generalized to the use of an arbitrary number of n hypotheses with $n > 3$. For that purpose, one inductively defines

$$p_{i+1} := (1-\alpha_{i+1})p_i + \alpha_{i+1}h_{i+1},$$

until $i = n-1$, which results in the prediction signal p_n. By (7), also a weighted bi-prediction mode similar to [39] is supported by the present method.

The multi-hypothesis inter prediction mode was integrated as follows. Additional hypotheses can be added to the inter prediction signal $p_{uni/bi}$ in all cases except for the case where the skip mode is used. In particular, the hypotheses can also be added in the case of merge mode that is not skip. In the case of merge mode, if a merging candidate has more than two hypotheses, not only the uni- or bi-prediction parameters, but also the additional prediction parameters of the selected merging candidate are used for the current block.

If the inter-prediction parameters for i hypotheses have been signaled and if more than i hypotheses are allowed, a flag is sent in the bitstream that determines whether an additional hypothesis is to be used. If this is the case, the weighting factor α for the additional hypothesis is additionally transmitted. The signaling of the motion vectors corresponding to additional hypotheses is very similar to the case of uni- or bi-prediction. The only exception is that for each additional hypothesis, a single reference picture list is used. This list is constructed by interleaving the reference picture lists 0 and 1. For more details on multi-hypotheses prediction, we refer to [40].

V. Enhancement of Prediction Signals

In this section, we describe two methods for improving the quality of intra- and inter-picture prediction signals.

A. Signal Adaptive Diffusion Filter

The idea of the signal adaptive diffusion filters is to increase the prediction quality by smoothing the prediction signals in such a way that noise is removed but edges are kept. If $pred$ is a given prediction signal, such an approach can be modeled by considering a scale space of filtered versions $F_t(pred)$, $t \geq 0$, with inital condition $F_0(pred) = pred$ that should become smoother the larger t is.

Going back to the work of [41], one way to generate such a model is to let $pred$ solve a discretization of the equation

$$\frac{\partial}{\partial t}F_t(pred)(x,y) = \mathrm{div}(c(x,y)\nabla F_t(pred)(x,y)), \tag{8}$$

where div is the divergence operator. If c is constant, then the differential operator occuring on the right hand side of (8) is just the Laplace operator. In this case, equation (8) describes a uniform diffusion, i.e., a smoothing that is uniform in all spatial directions. While such a filtering may attenuate noise present in the inital prediction signal, it can also degrade

important image content like edges. As a consequence, for such cases one tries to define the function c in (8) such that uniform smoothing is limited to image regions having similar sample values and is not carried out accross edges. In order to detect such edges, one invokes smoothed versions of the gradient of $pred$. In particular, the function c depends on the prediction signal $pred$ itself [41].

As suggested in [42], in order to incorporate the direction of edges, it is beneficial to let the function c take values in the 2×2 matrices rather than being scalar valued as in [41]. Taking up ideas of [42], we defined such a function c as follows. First, let $J_\rho(pred)$ denote the convolution of the diffusion tensor $J = \nabla pred \cdot (\nabla pred)^t$ with a Gaussian kernel K_ρ. Then, we put

$$c(x, y, pred) := \exp(-J_\rho(pred)/\mu). \tag{9}$$

Here, exp denotes the exponential function on matrices and μ is some fixed constant. At each sample position, the 2×2 matrix $J_\rho(pred)$ is diagonalizable. The major eigenvector corresponding to the larger eigenvalue points into the direction of the gradient characterizing the edge. Since $\exp(-\lambda/\mu)$, as a function of the eigenvalues λ, is monotonously decreasing, diffusion along the major eigenvector is attenuated.

We replace the continuous parameter t in (8) by a discrete time parameter n that belongs to a set of two predefined parameters $\{n_1, n_2\}$. Then, for each such n, the discretization of (8) is computed by n times applying a convolution of the initial prediction signal $pred$ with a 3×3 filter h. This filter varies for every sample position and is computed in advance out of the inital prediction signal $pred$.

As an alternative option, we also allowed uniform diffusion. In that case, for n belong to a parameter set $\{n_1', n_2'\}$, uniform diffusion is realized by n convolutions with a fixed 3×3-filter that is independent of the prediction signal and the sample position.

It is signaled in the bitstream if diffusion is to be applied on a given block. For inter-blocks, diffusion is not supported for the skip-mode. If diffusion is to be applied, it is additionally signaled whether non-uniform or uniform diffusion is to be used and which value for the parameter n has to be taken. The compression benefit of the diffusion filter highly depends on the resolution. For low resolutions, it gives significantly less coding gains than for high resolutions. For more details about the content of the present section, we refer to [43], [44].

B. Prediction Refinement using DCT thresholding

We designed a thresholding method by which we tried to align a given prediction signal with reconstructed samples in some neighborhood and thereby to improve the prediction quality. Our key idea is to do this by exploiting sparsity properties in the DCT domain which are typical for natural images and which are particularly present on large blocks. In contrast to the diffusion filter described in the previous section, this approach does not work as a denoising tool but rather as a texture synthesis tool to improve the prediction.

As illustrated in Fig. 7, we start with a prediction signal p that can arise either by motion-compensated or spatial intra prediction. Then, for given extension sizes K and L, we define

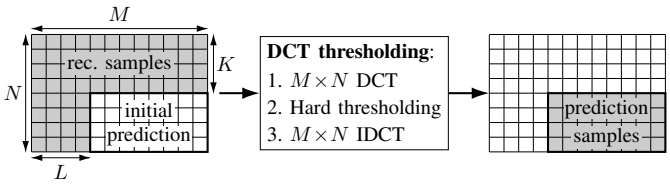


Fig. 7. Prediction signal filtering by DCT thresholding.

an enlarged prediction y by extending p with the reconstructed samples in the K lines above and L lines left. Given the extended prediction y, we map into the frequency domain via the orthogonal discrete cosine transform W and get the transformed block $Y = Wy$. Next, for a threshold value $\tau > 0$ the thresholded signal $\widetilde{Y}$ is defined by setting to zero all frequency components of Y whose absolute value is smaller than τ. Finally, if W^T denotes the inverse DCT, we compute the modified extended prediction signal $\tilde{y}$ as $\tilde{y} = W^T\widetilde{Y}$. The existing prediction p is then replaced by the restriction of $\tilde{y}$ to the given block.

Our extended prediction signal y always contains the initial prediction signal p in its bottom right corner. This contrasts the situation in sparse inpainting algorithms where the samples to fill are located on a random subset, see [45]. Also, our method does not describe a denoising in the manner of [46], [47]. Rather, the procedure aims to force the prediction signal to have sparse transform coefficients in accordance with its neighborhood. The coding gain vastly decreases when the reconstructed boundary is omitted from the scheme. Finally, in the case of an inter predicted signal p, the outcome of the thresholding still is subject to the reconstructed neighborhood.

The thesholding is applied solely to the luma signal. A flag signaled in the bitstream indicates whether it has to be applied or not. For each block size, four different pairs of parameters K, L as well as eight different thresholds are possible. If the thresholding is to be applied on a given block, these parameters are signaled for the corresponding block.

We refer to [48], which contains modifications of our method in comparison to our CfP submission that significantly increase both the coding gain and the encoder runtime.

VI. Transform Coding of Prediction Residuals

In this section, we describe our approach to transform coding. Prediction residuals are transformed using block adaptive transforms. The resulting transform coefficients are quantized using trellis-coded quantization. Finally, the obtained quantization indexes, which are also referred to as transform coefficient levels, are entropy coded. The entropy coding includes advanced concepts and adaptations for utilizing properties of the trellis-coded quantizer.

A. Adaptive Transform Selection

A central method of modern video codecs like AVC or HEVC is to transform prediction residuals in order to achieve energy compaction. In HEVC, only one transform is supported for each block: The two-dimensional DST-VII is used on intra-blocks of size 4×4, while the two-dimensional DCT-II is used in all other cases. Both of these transforms are separable.

On the other hand, in the JEM, the number of possible transforms is largely extended, in particular for intra blocks. More precisely, each intra block can be transformed using one out of five separable primary transforms that are combinations of DCTs and DSTs. These combinations depend on the intra mode [49]. Furthermore, one out of three non-separable secondary transforms may be additionally applied. These secondary transforms are restricted to transform the square of the at most 8×8 lowest frequencies of a given separable primary transform [50]. For each intra-prediction mode, a first set of three non-separable secondary transforms is used for blocks of size 4×4, 4×8 or 8×4 and a second set of three non-separable secondary transforms is used for all other blocks. As a consequence, 20 transforms are possible for each intra block in the JEM.

In our submission, we restricted the number of transforms that are possible on a given intra block to five. The set of supported transform candidates depends on both the intra prediction mode and the block size.

The five transform candidates are defined as follows. For each block size $M \times N$, with M and N being powers of two and $4 \leq \min(M, N)$ and $\max(M, N) \leq 32$, three non-separable transforms are specified. These transforms are used as primary transforms if $\max(M, N) \leq 8$. In all other cases, they are used as secondary transforms acting on the $\min(M, 8) \times \min(N, 8)$ lowest frequencies of a separable primary transform. In the first case, the three non-separable primary transforms are combined with two out of the five separable primary transforms of the JEM. In the second case, to each of our three non-separable secondary transforms, one or two primary transforms of the JEM are assigned while the remaining transforms are comprised by at most two of the primary transforms of the JEM.

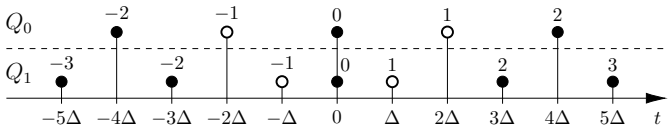
For blocks with a non-power-of-two side length, the non-separable transforms of the next-largest block that has power-of-two side lengths are reused. Note that these transforms are always restricted secondary transforms and thus they can be used in both cases. A candidate list of five transforms is then supported as before.

Our non-separable transforms were derived as KLTs using a large set of training data consisting of both video sequences and still images. This set did not include any test sequence from the CfP. In a second step, the specific five transforms for each intra prediction mode and block shape were derived by collecting rate-distortion costs of all possible transform candidates using a reference encoder and selecting the five best ones.

In the case of inter blocks, we used the five separable primary transforms exactly as the JEM. For more details about the content of the present section, we refer to [51].

B. Trellis-Coded Quantization of Transform Coefficients

In modern video coding standards such as AVC and HEVC, the prediction residues are coded using transform coding with scalar uniform reconstruction quantizers (URQs). For further improving coding efficiency, we propose to replace the URQs with a low-complexity variant of vector quantization, which is known as trellis-coded quantization (TCQ) [52]. Even though TCQ effectively represents a constrained vector quantizer, it has several commonalities with URQs and can be straightforwardly combined with state-of-the-art entropy coding techniques. From a decoder perspective, TCQ specifies two scalar quantizers and a procedure for switching between these quantizers based on preceding quantization indexes.



current state s_k	quantizer used for t_k	next state s_{k+1} for	
		even q_k	odd q_k
0	Q_0	0	2
1	Q_0	2	0
2	Q_1	1	3
3	Q_1	3	1

Fig. 8. Design of the trellis-coded quantizer: (top) Reconstruction levels and quantization indexes (labels above the circles) of the two scalar quantizers Q_0 and Q_1; (bottom) State transition for quantizer selection.

The two scalar quantizers Q_0 and Q_1 of the TCQ design chosen are illustrated in the top diagram of Fig. 8. Similar to URQs, the reconstruction levels of both quantizers represent integral multiples of a quantization step size Δ. Note that both quantizers include the reconstruction level of zero, which has been shown to improve the low rate compression performance of TCQ [53]. But in contrast to other low-rate designs [54], we chose symmetric quantizers, which are better suited for the applied entropy coding. The reconstruction levels chosen by an encoder are indicated by quantization indexes q (labels in Fig. 8), which are transmitted in the bitstream (see Sec. VI-C).

The transform coefficients have to be reconstructed in a predefined order, which is chosen to be equal to the coding order of quantization indexes. The quantizer selection is specified by a state transition process with 4 states, which is given by the table in Fig. 8. The state s_0 for the first coefficient t_0 of a block is set equal to $s_0 = 0$. Then, the state s_{k+1} for a coefficient t_{k+1} is uniquely determined by the preceding state s_k and the parity of the preceding quantization index q_k.

At the decoder side, the transform coefficients of a block can be reconstructed by first deriving integral numbers z_k according to

$$z_k = \begin{cases} 2\,q_k & : \quad s_k < 2 \\ 2\,q_k - \mathrm{sgn}(q_k) & : \quad s_k \geq 2, \end{cases} \tag{10}$$

where the states s_k are determined as described above. The reconstructed transform coefficients t'_k are then obtained by multiplying the numbers z_k with the quantization step size Δ.

For selecting quantization indexes q_k in an encoder, the potential transitions between the quantizers Q_0 and Q_1 can be elegantly represented by a trellis [52] with 4 states per coefficient. In our encoder, first, the two quantization indexes q_k with minimum distortion $D(q_k)$ are selected for each coefficient t_k and each quantizer. Then, all connections between two trellis nodes are assigned with the corresponding Lagrangian costs $D(q_k) + \lambda R(q_k)$, where $R(q_k)$ represents an estimate

$\lvert q_k \rvert$	0	1	2	3	4	5	6	7	8	9	$\cdots$
sig	0	1	1	1	1	1	1	1	1	1	$\cdots$
gt1	–	0	1	1	1	1	1	1	1	1	$\cdots$
gt2	–	–	0	1	1	1	1	1	1	1	$\cdots$
gt3	–	–	–	0	1	1	1	1	1	1	$\cdots$
gt4	–	–	–	–	0	1	1	1	1	1	$\cdots$
rem	–	–	–	–	–	0	1	2	3	4	$\cdots$

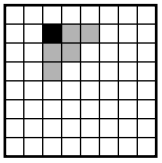


Fig. 9. Transform coefficient coding: (left) Binarization of absolute values; (right) Local template (gray) around current scan position (black).

of the number of bits required for coding q_k. Finally, the sequence of quantization indexes q_k is determined by finding the minimum cost path through the trellis using the Viterbi algorithm [55]. The encoding algorithm is more complex, but still comparable to state-of-the-art rate-distortion optimized quantization (RDOQ) approaches [56], [57]. For further details on the decision algorithm, the reader is referred to [58].

C. Transform Coefficient Coding

For entropy coding the quantization indexes q_k for transform coefficients, we use an approach that is similar to the HEVC transform coefficient coding [31], but includes additional improvements as well as adjustments for TCQ. The subdivision into 4×4 coefficient groups, the scanning order, the coding of coded block flags (for the transform block and the coefficient groups), and the coding of the position of the last significant coefficient is exactly the same as in HEVC. Changes relative to HEVC are introduced at a coefficient group level and are described in the following.

1) Binarization: The absolute values $|q_k|$ of the quantization indexes are binarized as illustrated in Fig. 9. The binary decisions (also referred to as bins) *sig*, *gt1*, *gt2*, *gt3*, *gt4* are coded in the regular mode of the arithmetic coding engine, which uses adaptive probability models. The non-binary syntax element *rem* is binarized using the same parametric codes as in HEVC and the resulting bins are coded in the bypass mode of the arithmetic coding engine. The signs (for absolute values greater than zero) are also coded in the bypass mode.

2) Coding Order: In contrast to HEVC, all bins specifying the absolute values are transmitted in a single pass over the scan positions of a coefficient group. This has the following two advantages for the context modeling (see below): (a) The context selection can be improved by evaluating completely reconstructed absolute values in a local neighborhood; (b) The knowledge of the quantizer used for a current transform coefficient (which depends on the parities of the preceding quantization indexes) can additionally be exploited for improving the context modeling in connection with TCQ. The signs are coded in a second pass over the scan positions.

3) Context Modeling: In order to utilize conditional statistics for an efficient coding, the adaptive probability models (also called contexts) for the regular coded bins are chosen among a set of available models. One difference to HEVC is that the context selection depends on already transmitted absolute values in a local neighborhood [59], [60]. Let *sumAbs* and *numSig* represent the sum of absolute values and the number of non-zero values, respectively, in the local template illustrated in Fig. 9. The context for the *sig* bin depends on the diagonal position $d = x + y$ (3 classes) and the value $\min(\textit{sumAbs}, 5)$. The context for the bins *gt1*, *gt2*, *gt3*, and *gt4* is chosen depending on the diagonal position d (4 classes) and the value $\min(\textit{sumAbs} - \textit{numSig}, 4)$. An additional context is used for the last significant position in a transform block. Since the distances between zero and the first non-zero reconstruction levels are different for the two quantizers Q_0 and Q_1, the binary probabilities also depend on the quantizer used. This fact is exploited by using two different sets of context models (one for Q_0 and another for Q_1) for the bins *sig* and *gt1*.

4) Rice Parameter Selection: Similarly as in HEVC, the code that is used for binarizing the remainder *rem* is specified by a so-called Rice parameter. In our approach, the Rice parameter is chosen depending on the sum of absolute values *sumAbs* in the local template (via a look-up table).

VII. Multiple Feature based Adaptive Loop Filter

The idea of adaptive loop filters is to apply Wiener Filters [61] to the reconstructed frame. The performance of this approach greatly increases if one does not only use one filter for the whole frame but uses a clustering of the reconstructed frame into disjoint filter classes $\mathcal{C}_1, \ldots, \mathcal{C}_L$ such that on each class $\mathcal{C}_i$ a different filter F_i is applied, [62], [63].

In more detail, at the encoder side, the coefficients of the filter F_i are computed by minimizing the mean-squared error between the original and the reconstructed samples that belong to the class $\mathcal{C}_i$, [61], [64]. If this is beneficial in a rate-distortion sense, the filter taps of F_i are sent in the bitstream, [64]. Then, at the decoder, the filter F_i is applied to all reconstructed samples in class $\mathcal{C}_i$.

The most important problem for the latter approach to work is to find a suitable classification algorithm that leads to the classification into the classes $\mathcal{C}_1, \ldots, \mathcal{C}_L$. In the JEM, a Laplacian-based classifier was used [63]. Our classifyer is motivated by the following classifier. We let $L = 2$ and set

$$\begin{aligned} \mathcal{C}_0^{opt} &:= \{(x,y)\colon s(x,y) \le \hat{s}(x,y)\}, \\ \mathcal{C}_1^{opt} &:= \{(x,y)\colon s(x,y) > \hat{s}(x,y)\}, \end{aligned} \tag{11}$$

where $s(x,y)$ and $\hat{s}(x,y)$ denote the original and reconstructed samples at sample position (x,y). We think of (11) as a kind of ideal classifier that we try to approximate avoiding the use of original samples. For that purpose, we use a rank-based classifier as follows. We let

$$rk(x,y) := \#\{k,l\colon \hat{s}(x{+}k,y{+}l) > \hat{s}(x,y)\colon |k| \le 1\colon |l| \le 1\}$$

and define $\mathcal{C}_i^{rk}$ as the class of all samples that have rank i. In this way, we obtain 9 clusters $\mathcal{C}_0^{rk}, \ldots, \mathcal{C}_8^{rk}$. Heuristically, the larger the rank, the more likely it is that (x,y) belongs to class $\mathcal{C}_1^{opt}$ from (11). Dividing the dynamic range into three intervals of equal size, we refine each rank-based cluster $\mathcal{C}_i^{rk}$ into three classes according to the sample value. In this way, we have defined a clustering into 27 disjoint clusters.

In fact, dividing the dynamic range into 27 intervals of equal size, we found out that for some cases it is sufficient to cluster the samples according to the sample value. We call the latter classifier sample-based classifier.

We added the aforementioned two classifiers as alternative options to the Laplacian-based classifier used in the JEM. Thus, for each slice exactly one of these three classifiers can be used. Which one is to be used is tested at the encoder and is signaled to the decoder. The signaling of the filter taps that are to be used on the clusters was the same as in the JEM. More details about our classifieres can be found in [65].

VIII. Perceptually Optimized Encoder Control

In conventional encoder control algorithms, the deviation between an original picture s and the reconstructed picture $\hat{s}$ is measured using the sum of squared errors $D_{SSE}(s,\hat{s})$ or its normalized logarithm, the PSNR. However, it is well known that the PSNR in general does not correlate well with subjective judgment of image quality [66]. To mitigate this phenomenon, we partition a picture into blocks B_k on which we weight D_{SSE} by factors $w_k(s)$ that specify the subjective error sensitivity of the local content. Our overall distortion measure D_{WSEE}, the weighted sum of squared errors, is then defined as

$$D_{WSSE}(s,\hat{s}) := \sum_k w_k(s) \cdot D_{SSE,k}(s,\hat{s}). \tag{12}$$

Here, $D_{SSE,k}$ is the sum of squared errors on B_k.

We use the error measure (12) for the encoder control as follows. For each feasible rate budget, the encoder tries to minimize the distortion. By using the approach of Lagrangian multipliers, this is equivalent to the minimization of

$$J(\lambda) := D_{WSSE} + \lambda \cdot R \tag{13}$$

for each $\lambda > 0$ in a suitable interval. If, for simplification, one makes the assumption that the blocks B_k can be treated independently for the optimization of (13), then on each block B_k the encoder has to minimize

$$D_{SSE,k}(s,\hat{s}) + \lambda_k \cdot R_k, \qquad \lambda_k := \frac{\lambda}{w_k}. \tag{14}$$

Here, R_k is the rate on the block B_k. Thus, if a fixed operation point of the rate distortion curve for the error measure (12) is realized by a fixed Lagrangian multiplier λ via (13), then on each block B_k our encoder control uses the traditional mean squared error as a distortion measure but is steered by locally varying, signal adaptive Lagrangian multipliers λ_k via equation (14).

In particular, the optimal quantization step size, which is part of the encoder decision to minimize (14), changes for each block B_k. More precisely, as has been shown in [67], assuming a high-rate approximation of the rate-distortion curve and a uniform quantization error, the optimal quantization step size Δ_k to minimize (14) is approximately proportional to the square root of λ_k. This has also been verified experimentally [67], [68]. Thus, if $QP(\lambda)$ is the quantization parameter corresponding to λ, defined as in HEVC or JEM, then on each block B_k we need to work with the modified quantization parameter

$$QP_k(\lambda) := QP(\lambda) - \lfloor 3 \cdot \log_2(w_k) \rceil. \tag{15}$$

Here, $\lfloor \cdot \rceil$ indicates rounding.

We finally describe a simple approach for choosing the weighting factors. Let h be the high-pass filtered version of s that is computed using a 9-tap Laplacian filter. Then, if B is an image block with $|B|$ samples, we define

$$\alpha(s,B) := \min\left(\alpha_{\min}, \left(\sum_{(x,y)\in B} \frac{|h[x,y]|}{|B|}\right)^2\right),$$

where $\alpha_{\min}$ is a fixed constant that models the lower visual sensitivity limit, and put

$$w(s,B) := \left(\frac{\alpha(B)}{\alpha(s,B)}\right)^{0.5}. \tag{16}$$

Here, $\alpha(B)$ is a normalization constant that only depends on the image bit-depth and resolution and, like the exponent 0.5, is experimentally determined. For further details, we refer to [69], [70].

The meaning of the factors from (16) is that the less activity is present in the image content on B, the larger $w(s,B)$ becomes. This is to be seen in accordance with the well-known fact that, due to reduced perceptual masking capabilities, local image regions dominated by low frequency content are subjectively more sensitive to reconstruction errors than those also containing high frequency content. A subjective evaluation of our QP adaptation method, which is published in [69], supports this observation.

IX. Experimental Results

In the CfP [2], three different test categories comprised by different types of content were defined: The standard dynamic range (SDR) category, consisting of UHD (class A) and HD (class B) content, the high dynamic range (HDR) category and the 360° category. Moreover, two sets of coding conditions were defined. A random access case, denoted as constraint set 1 (CS1), and a low delay case, denoted as constraint set 2 (CS2). For the random access case, the structural delay was limited to 16 frames and the random access intervals were required to be 1.1s or less. For the low delay case, no picture reordering was allowed and no random access capabilities were required. Our submission was tested against two anchors, the first one being the HM 16.16 anchor and the second one being the JEM anchor.

Table I shows the objective results of our proposal against the HM 16.16 anchor and the JEM anchor for the random access scenario CS1. Table II shows test results of our submission against the HM 16.16 anchor and the JEM anchor for the low delay case CS2. Here, according to the CfP conditions, only results for the class SDR B are reported. For the results presented in Table I and Table II, four different bit rate points specified in the CfP [2] were used. During the 10-th JVET meeting held in San Diego in April 2018, subjective testing results for each individual response to the CfP were reported. Here, it turned out that also in a subjective evaluation, our submitted proposal yields a significant benefit over the current HEVC standard.

In Table III, we delineate the individual gains of the tools presented above in a random access configuration. Here, as

TABLE I
LUMA BD-RATE SAVINGS AND AVERAGE ENCODING/DECODING TIMES OF THE PROPOSED CODEC IN COMPARISON TO THE HM AND JEM ANCHORS FOR THE RANDOM ACCESS SCENARIO (CS1).

class	sequence	vs. HM anchor	vs. JEM anchor
SDR A	FoodMarket4	-37.4%	-6.3%
	CatRobot1	-43.7%	-7.4%
	DaylightRoad2	-45.0%	-8.0%
	ParkRunning3	-34.1%	-4.4%
	Campfire	-37.8%	-5.5%
	Avg. BD-rate savings	**-39.6%**	**-6.3%**
	Avg. encoding time	1540%	188%
	Avg. decoding time	778%	99%
SDR B	BQTerrace	-39.3%	-13.0%
	RitualDance	-32.2%	-6.8%
	MarketPlace	-33.4%	-6.2%
	BasketballDrive	-37.3%	-9.2%
	Cactus	-40.4%	-8.7%
	Avg. BD-rate savings	**-36.5%**	**-8.8%**
	Avg. encoding time	1869 %	221%
	Avg. decoding time	781%	104%
HDR A	DayStreet	-38.3%	-6.6%
	PeopleInShoppingCenter	-32.4%	-5.9%
	SunsetBeach	-26.8%	-6.2%
	Avg. BD-rate savings	**-32.5%**	**-6.2%**
	Avg. encoding time	775%	147%
	Avg. decoding time	748%	88%
HDR B	Market3	-29.3%	-7.9%
	ShowGirl2	-34.1%	-11.5%
	Hurdles	-37.4%	-9.7%
	Starting	-34.4%	-8.2%
	Cosmos1	-28.6%	-7.7%
	Avg. BD-rate savings	**-32.8%**	**-9.0%**
	Avg. encoding time	1726%	241%
	Avg. decoding time	757%	103%
360°	Balboa	-44.3%	-14.0%
	Chairlift	-48.4%	-25.7%
	KiteFlite	-27.5%	-12.1%
	Harbor	-31.4%	-13.9%
	Trolley	-26.6%	-12.8%
	Avg. BD-rate savings	**-35.7%**	**-15.7%**
	Avg. encoding time	1373%	241%
	Avg. decoding time	793%	126%

TABLE II
LUMA BD-RATE SAVINGS AND AVERAGE ENCODING/DECODING TIMES OF THE PROPOSED CODEC IN COMPARISON TO THE HM AND JEM ANCHORS FOR THE LOW DELAY SCENARIO (CS2).

class	sequence	vs. HM anchor	vs. JEM anchor
SDR B	BQTerrace	-32.9%	-9.5%
	RitualDance	-26.1%	-5.7%
	MarketPlace	-25.3%	-5.8%
	BasketballDrive	-30.3%	-6.5%
	Cactus	-34.1%	-8.6%
	Avg. BD-rate savings	**-29.7%**	**-7.2%**
	Avg. encoding time	2035%	240%
	Avg. decoding time	679%	111%

reference configuration, we used an encoder setting of our submission in which all presented and all JEM coding tools are disabled, but a modified block partitioning is used. This partitioning is the QT+BTS partitioning described in [12] in a configuration that only yields block sizes for which both the block width and height represent integer powers of two. This setting ensures that only block sizes also available with QTBT are used and it provides an increased coding efficiency relative to QTBT. In comparison to HEVC, QT+BTS in the configuration used provides luma BD rates of -8.1% while QTBT only provides -5.9%. For these simulations, we used fixed QP values of 22, 27, 32, 37. As test sequences, the CfP sequences as well as all JVET test sequences, including class F, are taken. If in Table III, a proposed tool replaces a JEM tool, the gains of the corresponding JEM tool are also given for comparison. Finally, in the last row of Table III, we report results over HM for a configuration of our codec in which all tools delineated in this table are switched on but all JEM coding tools are disabled. Here, the tools that replace a JEM tool and occur in Table III are enabled.

TABLE III
CODING EFFICIENCY AND COMPLEXITY ANALYSIS OF THE PROPOSED NEW CODING TOOLS, INCLUDING THE MODIFIED JEM TOOLS, MEASURED AS LUMA BD RATES AND AVERAGE ENCODING/DECODING TIMES FOR THE RANDOM ACCESS SCENARIO. FOR THE FIRST AND LAST ROW, THE ANCHOR IS HM. OTHERWISE, THE ANCHOR IS HM+QT+BTS.

coding tool	luma BD rate	encoding Time	decoding Time
QT+BTS partitioning over HM	-8.1%	133%	113%
QTBT (JEM) over HM	-5.9%	82%	108%
Line-based intra coding	-1.2%	117%	100%
Intra region-based template matching	-1.1%	118%	101%
Intra prediction with neural networks	-1.8%	168%	124%
Multi-hypothesis inter prediction	-1.1%	120%	101%
Diffusion filter	-0.7%	111%	102%
DCT thresholding	-0.5%	140%	103%
Adaptive transform selection	-4.0%	164%	106%
EMT and NSST (JEM)	-3.6%	185%	105%
Trellis-coded quantization and coefficient coding	-2.8%	111%	100%
Coefficient coding in JEM	-0.9%	103%	101%
Multiple feature based adaptive loop filter	-4.4%	129%	168%
GALF (JEM)	-4.1%	116%	171%
Proposed codec without JEM tools (over HM)	-21.0%	960%	209%

All objective results report luma Bjøntegaard delta (BD) rates according to [71], [72]. For the $360°$ content, we used the equi-angular cubemap (EAC) projection format, see [73] for details. No specific $360°$ or HDR coding tools were used and also no pre- or post-processing was applied in these categories, except for the projection format for $360°$. Moreover, in all reported results, the encoder control based on a subjective distortion measure as described in Section VIII was disabled. The reason is that the performance metric of [71], [72] is based on the unweighted sum of squared errors and is thus not aligned to the error measure presented in Section VIII.

X. CONCLUSION

In this paper, we presented new coding tools that were part of our response to the call for proposals. These tools are a new partitioning scheme based on generalized binary splits as well as new methods for prediction and transform coding. When used well together, they provide significant compression gains over state-of-the-art video coding technologies.

REFERENCES

[1] M. Albrecht *et al.*, "Description of SDR, HDR and 360° video coding technology proposal by Fraunhofer HHI," *Joint Video Experts Team, doc. JVET-J0014, San Diego*, Apr. 2018.

[2] A. Segall, V. Baroncini, J. Boyce, J. Chen, and T. Suzuki, "Joint Call for Proposals on Video Compression with Capability beyond HEVC," *doc. JVET-H1002, Macau*, Oct. 2017.

[3] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Rec. H.264 and ISO/IEC 14496-10, vers. 1, 2003.

[4] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits and Systems for Video Technol.*, vol. 13, no. 7, pp. 560–576, July 2003.

[5] ITU-T and ISO/IEC, "High Efficiency Video Coding," ITU-T Rec. H.265 and ISO/IEC 23008-2, vers. 1, 2013.

[6] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits and Systems for Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[7] J. Chen, E. Alshina, G. J. Sullivan, J.-R. Ohm, and J. Boyce, "Algorithm description of Joint Exploration Test Model 7 (JEM7)," *doc. JVET-G1001, Torino*, Aug. 2017.

[8] J. Chen, M. Karczewicz, Y.-W. Huang, K. Choi, J.-R. Ohm, and G. J. Sullivan, "The Joint Exploration Model (JEM) for Video Compression with Capability beyond HEVC," to appear in this volume.

[9] J. An, Y.-W. Chen, K. Zhang, H. Huang, Y.-W. Huang, and S. Lei, "Block partitioning structure for next generation video coding," ITU-T SG16/Q6, doc. COM16-C966, Sep. 2015.

[10] J. An, H. Huang, K. Zhang, Y.-W. Huang, and S. Lei, "Quadtree plus binary tree structure integration with JEM tools," *doc. JVET-B0023, San Diego*, Feb. 2016.

[11] A. Wieckowski, J. Ma, V. George, H. Schwarz, D. Marpe, and T. Wiegand, "Generalized binary splits: A versatile partitioning scheme for block-based hybrid video coding," in *submitted to IEEE Picture Coding Symp. (PCS), Ningbo*, Nov. 2019.

[12] J. Ma, A. Wieckowski, V. George, T. Hinz, S. De-Luxán Hernández, H. Kirchhoffer, R. Skupin, H. Schwarz, T. Schierl, D. Marpe, and T. Wiegand, "Quadtree plus binary tree with shifting (including software)," *doc. JVET-J0035, San Diego*, Apr. 2018.

[13] B. Bross, "Versatile Video Coding (Draft 1)," *doc. JVET-J1001, San Diego*, May 2018.

[14] X. Li, H.-C. Chuang, J. Chen, M. Karczewicz, L. Zhang, X. Zhao, and A. Said, "Multi-Type-Tree," *doc. JVET-D0117, Chengdu*, Oct. 2016.

[15] A. Wieckowski, J. Ma, H. Schwarz, D. Marpe, and T. Wiegand, "Fast partitioning decision strategies for the upcoming Versatile Video Coding (VVC) standard," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Taipei*, Sep. 2019, in press.

[16] J. Li, B. Li, J. Xu, and R. Xiong, "Intra Prediction Using Multiple Reference Lines for Video Coding," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Apr. 2017, pp. 221–230.

[17] Y. Sohn and W.-J. Han, "One Dimensional Transform for H.264 Based Intra Coding," in *Proc. IEEE Picture Coding Symp. (PCS), Lisbon*, Nov. 2007.

[18] G. Laroche, J. Jung, and W. Pesquet, "Intra prediction with 1D macroblock partitioning for image and video coding," in *Proc. SPIE Visual Commun. and Image Process.*, vol. 7257, Jan. 2009.

[19] C. Lai, J. Jiang, and P. Zhang, "One dimensional prediction and transform for intra coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Brussels*, Sep. 2011, pp. 3485–3488.

[20] X. Cao, C. Lai, Y. Wang, L. Liu, J. Zheng, and Y. He, "Short Distance Intra Coding Scheme for High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 790–801, Feb. 2013.

[21] S. De-Luxán-Hernández, H. Schwarz, D. Marpe, and T. Wiegand, "Line-Based Intra Prediction for Next-Generation Video Coding," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Athens*, Oct. 2018, pp. 221–225.

[22] ——, "Fast Line-Based Intra Prediction for Video Coding," in *Proc. IEEE Int. Symp. Multimedia (ISM), Taichung*, Dec. 2018, pp. 135–138.

[23] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra Prediction by Template Matching," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Atlanta*, Oct. 2006, pp. 1693–1696.

[24] ——, "Intra Prediction by Averaged Template Matching Predictors," in *Proc. IEEE 4th Consumer Commun. and Networking Conf.*, Jan. 2007, pp. 405–409.

[25] S. Cherigui, C. Guillemot, D. Thoreau, P. Guillotel, and P. Perez, "Hybrid template and block matching algorithm for image intra prediction," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Process. (ICASSP), Kyoto*, Mar. 2012, pp. 781–784.

[26] G. Venugopal, P. Merkle, D. Marpe, and T. Wiegand, "Fast template matching for intra prediction," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Beijing*, Sep. 2017, pp. 1692–1696.

[27] ——, "Intra Region-based Template Matching," *doc. JVET-J0039, San Diego*, Apr. 2018.

[28] X. Xu, K. Müller, and L. Wang, "CE8: Summary Report on Current Picture Referencing," *doc. JVET-L0028, Macau*, Oct. 2018.

[29] X. Xu, S. Liu, T. Chuang, Y. Huang, S. Lei, K. Rapaka, C. Pang, V. Seregin, Y. Wang, and M. Karczewicz, "Intra Block Copy in HEVC Screen Content Coding Extensions," *IEEE J. Emerging and Selec. Topics in Circuits and Systems*, vol. 6, no. 4, pp. 409–419, Dec. 2016.

[30] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," *arXiv:1511.07289*, 2015.

[31] J. Sole, R. Joshi, N. Nguyen, T. Ji, M. Karczewicz, G. Clare, F. Henry, and A. Duenas, "Transform Coefficient Coding in HEVC," *IEEE Trans. Circuits and Systems for Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.

[32] J. Pfaff, P. Helle, D. Maniry, S. Kaltenstadler, B. Stallenberger, P. Merkle, M. Siekmann, H. Schwarz, D. Marpe, and T. Wiegand, "Intra Prediction Modes based on Neural Networks," *doc. JVET-J0037, San Diego*, Apr. 2018.

[33] J. Pfaff, P. Helle, D. Maniry, S. Kaltenstadler, W. Samek, H. Schwarz, D. Marpe, and T. Wiegand, "Neural network based intra prediction for video coding," in *Proc. SPIE Applic. of Digital Image Process. XLI*, vol. 10752, Sep. 2018.

[34] P. Helle, J. Pfaff, M. Schäfer, R. Rischke, H. Schwarz, D. Marpe, and T. Wiegand, "Intra Picture Prediction for Video Coding with Neural Networks," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Mar. 2019.

[35] 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, July 2018.

[36] B. Girod, "The Efficiency of Motion-Compensating Prediction for Hybrid Coding of Video Sequences," *IEEE J. Selec. Areas in Commun.*, vol. 5, no. 7, pp. 1140–1154, Aug. 1987.

[37] ——, "Efficiency Analysis of Multihypothesis Motion-Compensated Prediction for Video Coding," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 173–183, Feb. 2000.

[38] M. Flierl, T. Wiegand, and B. Girod, "A locally optimal design algorithm for block-based multi-hypothesis motion-compensated prediction," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Mar. 1998, pp. 239–248.

[39] C.-C. Chen, X. Xiu, Y. He, and Y. Ye, "Generalized bi-prediction for inter coding," *doc. JVET-C0047, Geneva*, May 2016.

[40] M. Winken, C. Bartnik, H. Schwarz, D. Marpe, and T. Wiegand, "Multi-Hypothesis Inter Prediction," *doc. JVET-J0041, San Diego*, Apr. 2018.

[41] P. Perona and J. Malik, "Scale-Space and Edge Detection Using Anisotropic Diffusion," *IEEE Trans. Pattern Analyis and Machine Intell.*, vol. 12, no. 7, pp. 629–639, July 1990.

[42] J. Weickert, *Anisotropic Diffusion in Image Processing*. Stuttgart: B. G. Teubner, 1998.

[43] J. Rasch, J. Pfaff, M. Schäfer, H. Schwarz, M. Winken, M. Siekmann, D. Marpe, and T. Wiegand, "A Signal Adaptive Diffusion Filter for Video Coding," in *Proc. IEEE Picture Coding Symp. (PCS), San Francisco*, June 2018, pp. 131–133.

[44] J. Pfaff, J. Rasch, M. Schäfer, H. Schwarz, A. Henkel, M. Winken, M. Siekmann, D. Marpe, and T. Wiegand, "Signal Adaptive Diffusion Filters for Video Coding," *doc. JVET-J0038, San Diego*, Apr. 2018.

[45] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Basel: Birkhäuser, 2013.

[46] D. L. Donoho, "De-Noising by Soft-Thresholding," *IEEE Trans. Inform. Theory*, vol. 41, no. 3, pp. 613–627, May 1995.

[47] M. Jansen, *Noise Reduction by Wavelet Thresholding*. New York: Springer, 2001.

[48] M. Schäfer, J. Pfaff, J. Rasch, T. Hinz, H. Schwarz, T. Nguyen, G. Tech, D. Marpe, and T. Wiegand, "Improved Prediction via Thresholding Transform Coefficients," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Athens*, Oct. 2018, pp. 2546–2549.

[49] X. Zhao, J. Chen, M. Karczewicz, L. Zhang, X. Li, and W.-J. Chien, "Enhanced Multiple Transform for Video Coding," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Mar. 2016, pp. 73–82.

[50] X. Zhao, J. Chen, A. Said, V. Seregin, H. Egilmez, and M. Karczewicz, "NSST: Non-separable secondary transforms for next generation video coding," in *Proc. IEEE Picture Coding Symp. (PCS), Nuremberg*, Dec. 2016, pp. 1–5.
[51] M. Siekmann, C. Bartnik, B. Stallenberger, J. Pfaff, H. Schwarz, D. Marpe, and T. Wiegand, "Set of Transforms," *doc. JVET-J0040, San Diego*, Apr. 2018.
[52] M. W. Marcellin and T. R. Fischer, "Trellis Coded Quantization of Memoryless and Gauss-Markov Sources," *IEEE Trans. Communic.*, vol. 38, no. 1, pp. 82–93, Jan. 1990.
[53] J. H. Kasner, M. W. Marcellin, and B. R. Hunt, "Universal Trellis Coded Quantization," *IEEE Trans. Image Process.*, vol. 8, no. 12, pp. 1677–1687, Dec. 1999.
[54] R. L. Joshi, V. J. Crump, and T. R. Fischer, "Image Subband Coding Using Arithmetic Coded Trellis Coded Quantization," *IEEE Trans. Circuits and Systems for Video Technol.*, vol. 5, no. 6, pp. 515–523, Dec. 1995.
[55] G. D. Forney, Jr., "The Viterbi Algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.
[56] K. Ramchandran and M. Vetterli, "Rate-Distortion Optimal Fast Thresholding with Complete JPEG/MPEG Decoder Compatibility," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 700–704, Sep. 1994.
[57] M. Karczewicz, Y. Ye, and I. Chong, "Rate Distortion Optimized Quantization," ITU-T SG16/Q6 (VCEG), doc. VCEG-AH21, Jan. 2008.
[58] H. Schwarz, T. Nguyen, D. Marpe, and T. Wiegand, "Hybrid Video Coding with Trellis-Coded Quantization," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Mar. 2019.
[59] T. Nguyen, H. Schwarz, H. Kirchhoffer, D. Marpe, and T. Wiegand, "Improved context modeling for coding quantized transform coefficients in video compression," in *Proc. IEEE Picture Coding Symp. (PCS), Nagoya*, Dec. 2010, pp. 378–381.
[60] J. Chen, W.-J. Chien, M. Karczewicz, X. Li, H. Liu, A. Said, L. Zhang, and X. Zhao, "Further improvements to HMKTA-1.0," ITU-T SG16/Q6 (VCEG), doc. VCEG-AZ07, June 2015.
[61] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. New York: John Wiley and Sons, 1949.
[62] C.-Y. Chen, C.-Y. Tsai, Y.-W. Huang, T. Yamakage, I. Chong, C. Fu, T. Itoh, T. Watanabe, T. Chujoh, M. Karczewicz, and S. Lei, "The adaptive loop filtering techniques in the HEVC standard," in *Proc. SPIE Applic. of Digital Image Process. XXXV*, vol. 8499, Oct. 2012.
[63] M. Karczewicz, L. Zhang, W.-J. Chien, and X. Li, "Improvements on adaptive loop filter," *doc. JVET-B0060, San Diego*, Feb. 2016.
[64] S. Wittmann and T. Wedi, "Transmission of Post-Filter Hints for Video Coding Schemes," in *Proc. IEEE Int. Conf. Image Process. (ICIP), San Antonio*, vol. 1, Sep. 2007, pp. 81–84.
[65] J. Erfurt, W. Q. Lim, H. Schwarz, D. Marpe, and T. Wiegand, "Multiple Feature-based Classifications Adaptive Loop Filter," in *Proc. IEEE Picture Coding Symp. (PCS), San Francisco*, June 2018, pp. 91–95.
[66] B. Girod, "Psychovisual aspects of image processing: What's wrong with mean squared error?" in *Proc. 7th Workshop on Multidimensional Signal Process.*, Sep. 1991, pp. P.2–P.2.
[67] G. J. Sullivan and T. Wiegand, "Rate-Distortion Optimization for Video Compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.
[68] T. Wiegand and B. Girod, "Lagrange multiplier selection in hybrid video coder control," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Thessaloniki*, vol. 3, Oct. 2001, pp. 542–545.
[69] C. R. Helmrich, S. Bosse, M. Siekmann, H. Schwarz, D. Marpe, and T. Wiegand, "Perceptually Optimized Bit Allocation and Associated Distortion Measure for Block-Based Image or Video Coding," in *Proc. IEEE Data Compression Conf. (DCC), Snowbird*, Mar. 2019.
[70] J. Erfurt, C. R. Helmrich, S. Bosse, H. Schwarz, D. Marpe, and T. Wiegand, "A Study of the Perceptually Weighted Peak Signal-to-Noise Ratio (WPSNR) for Image Compression," in *Proc. IEEE Int. Conf. Image Process. (ICIP), Taipei*, Sep. 2019, in press.
[71] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16/Q6 (VCEG), doc. VCEG-M33, Mar. 2001.
[72] ——, "Improvement of BD-PSNR Model," ITU-T SG16/Q6 (VCEG), doc. VCEG-AI11, July 2008.
[73] Y. Ye, E. Alshina, and J. Boyce, "Algorithm descriptions of projection format conversion and video quality metrics in 360°," *doc. JVET-G1003, Torino*, July 2017.