Video Compression Using Nested Quadtree Structures, Leaf Merging and Improved Techniques for Motion Representation and Entropy Coding

Detlev Marpe, *Senior Member, IEEE,* Heiko Schwarz, Sebastian Bosse, Benjamin Bross, Philipp Helle, Tobias Hinz, Heiner Kirchhoffer, Haricharan Lakshman, Tung Nguyen, Simon Oudin, Mischa Siekmann, Karsten Sühring, Martin Winken, and Thomas Wiegand, *Senior Member, IEEE*

***Abstract*—A video coding architecture is described that is based on nested and pre-configurable quadtree structures for flexible and signal-adaptive picture partitioning. The primary goal of this partitioning concept is to provide a high degree of adaptability for both temporal and spatial prediction as well as for the purpose of space-frequency representation of prediction residuals. At the same time, a leaf merging mechanism is included in order to prevent excessive partitioning of a picture into prediction blocks and to reduce the amount of bits for signaling the prediction signal. For fractional-sample motion-compensated prediction, a fixed-point implementation of the Maximal-Order-Minimum-Support (MOMS) algorithm is presented that uses a combination of IIR and FIR filtering. Entropy coding utilizes the concept of probability interval partitioning entropy (PIPE) codes that offers new ways for parallelization and enhanced throughput. The presented video coding scheme was submitted to a joint Call for Proposals of ITU-T Visual Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG) and was ranked among the five best performing proposals, both in terms of subjective and objective quality.**

***Index Terms*—High efficiency video coding, HEVC.**

I. Introduction

THIS paper describes a video compression scheme that intends to address both the aspects of coding efficiency and implementation cost in a well-balanced relationship. Its design can be considered as a generalization of concepts that already form the basis of the existing state-of-the-art H.264/AVC standard [1], [2], [3], [4]. While the individual building blocks of the proposed algorithm are kept as simple as possible, the flexibility of block partitioning for prediction and transform coding is substantially increased relative to prior standardized designs.

The proposed scheme was submitted as a proposal [5] in response to the joint Call for Proposals (CfP) on video compression technology [6]. It was ranked among the five best performing proposals [7], from which design elements were selected to specify a first Test Model under Consideration (TMuC) [8] in the course of the recently initiated standardization project of High Efficiency Video Coding (HEVC) [9], [10].

The paper is organized as follows. The next section highlights the main features of the proposed video compression scheme. Section III explains the fundamental structural elements for picture partitioning. The methods of motion-compensated prediction are described in Section IV. Section V deals with spatial intra prediction and Section VI explains the concept of variable block-size spatial transforms and quantization. Internal bit-depth expansion and in-loop filtering are described in Section VII and VIII, respectively. The new entropy coding concept is presented in Section IX. Section X describes the encoder control and Section XI presents the experimental results.

II. Overview of the Video Coding Scheme

The presented video coding scheme is based on the conventional hybrid approach of using spatial and temporal prediction, followed by transform coding of the residual and entropy coding of quantized transform coefficients and other coding parameters. The main innovative and distinctive features are given as follows:

- **Wide-range variable block-size prediction**: The size of prediction blocks can be adaptively chosen by using a quadtree-based partitioning. Maximum ($N_{\max}$) and minimum ($N_{\min}$) admissible block edge length can be specified as a side information. The results in Section XI are obtained with $N_{\max} = 64$ and $N_{\min} = 4$.
- **Nested wide-range variable block-size residual coding**: The block size used for DCT-based residual coding is adapted to the characteristics of the residual signal by using a nested quadtree-based partitioning of the corresponding prediction block.
- **Merging of prediction blocks**: In order to reduce the side information required for signaling the prediction parameters, neighboring blocks can be merged into one region that is assigned only a single set of prediction parameters.
- **Fractional-sample MOMS interpolation**: Interpolation of fractional-sample positions for motion-compensated prediction is based on a fixed-point implementation

D. Marpe, H. Schwarz, S. Bosse, B. Bross, P. Helle, T. Hinz, H. Kirchhoffer, H. Lakshman, T. Nguyen, S. Oudin, M. Siekmann, K. Sühring, and M. Winken are with the Image & Video Coding group, Fraunhofer Institute for Telecommunications — Heinrich Hertz Institute (Fraunhofer HHI), 10587 Berlin, Germany (e-mail: name.surname@hhi.fraunhofer.de).

T. Wiegand is jointly affiliated with the Image Processing Department, Fraunhofer HHI, and the Image Communication Chair, Technical University of Berlin, 10587 Berlin, Germany (e-mail: thomas.wiegand@hhi.fraunhofer.de).

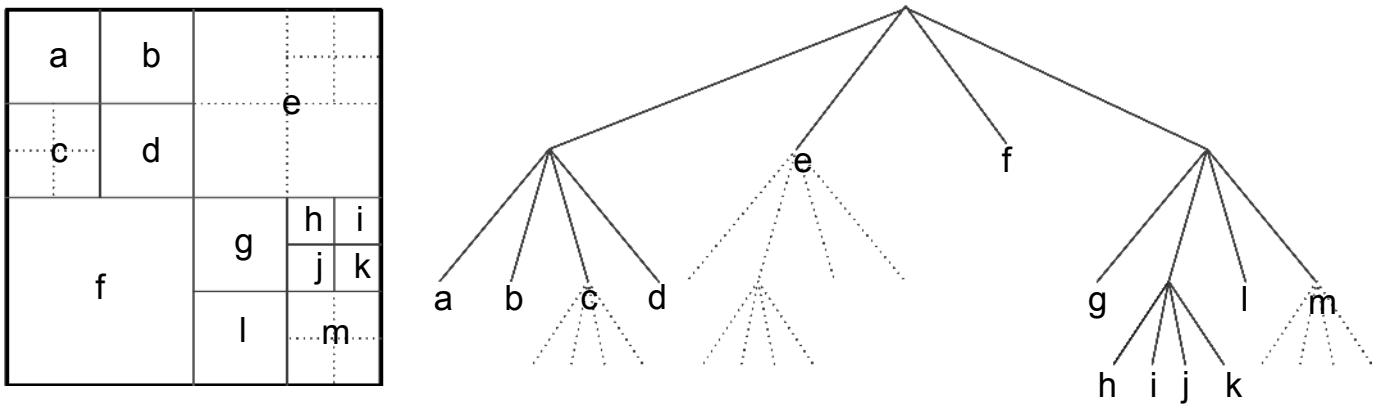

Fig. 1. Example of a nested quadtree structure (right part) for dividing a given coding tree block (left part; in black) into prediction blocks (solid gray lines) and transform blocks (dashed gray lines) of variable size. The order of parsing the prediction blocks follows their labeling in alphabetical order.

of the Maximal-Order-Minimum-Support (MOMS) algorithm using an IIR/FIR filter.

- **Adaptive in-loop filter**: In addition to the deblocking filter, a separable 2D Wiener filter is applied within the coding loop. The filter is adaptively applied to selected regions indicated by the use of quadtree-based partitioning.
- **PIPE coding**: The novel probability interval partitioning entropy (PIPE) coding scheme provides the coding efficiency and probability modeling capability of arithmetic coding at the complexity level of Huffman coding.

III. Picture Partitioning for Prediction and Residual Coding

The concept of a macroblock as the basic processing unit in standardized video coding is generalized to what we call a *coding tree block* (CTB). A CTB covers a square block of $N_{\max} \times N_{\max}$ luma samples and two corresponding blocks of chroma samples. To each CTB an associated quadtree structure is attached that indicates how the blocks are further subdivided for the purpose of prediction and residual coding. Dividing each picture into CTBs and further recursively subdividing each CTB into square blocks of variable size allows to partition a given picture of a video signal in such a way that both the block sizes and the block coding parameters such as prediction or residual coding modes will be adapted to the specific characteristics of the signal at hand. Note that all three typically used signal components (luma and two chroma) share the same structure for picture partitioning.

Figure 1 (left) shows an example of a coding tree block (in black; luma samples only) and how it is subdivided into prediction blocks (solid lines) and transform blocks (dashed lines). On the right-hand side of the same figure, the corresponding nested quadtree structure for CTB partitioning is shown. In this example, the quadtree specifying the prediction blocks (solid lines) has four levels, with the root at level 0 corresponding to the full CTB size (maximum prediction block size), and with level 3 corresponding to a block size, i.e., edge length of one eighth of the CTB edge length. Generally, subblocks at level i always have a block edge length of $2^{-i} \cdot N_{\max}$ with $N_{\max}$ given as a power of two and denoting the edge length of the square block of luma samples associated with the CTB. Both the CTB edge length $N_{\max}$ and the maximum number of levels, or equivalently the maximum depth D of the so-called *prediction quadtree*, as shown by solid gray lines in the example of Figure 1 (right), are specified as side information in the bitstream. Note that with the choice of D, the minimum possible prediction block size in terms of edge length is constrained to $N_{\min} = 2^{-D} \cdot N_{\max}$. Consequently, the maximum and minimum possible prediction block size can be freely chosen on a sequence level, depending on the application, the video material, the resolution, etc.

The samples of each prediction block, covering both luma and chroma components, are either *intra-picture predicted*, i.e., predicted by using decoded and reconstructed samples of neighboring blocks of the same picture, or they are predicted by using decoded and reconstructed samples from previously decoded pictures. The latter case is commonly referred to as *inter-picture prediction* or *motion-compensated prediction* (MCP). In both the intra-picture and inter-picture prediction case, the corresponding *residual* of block samples, which is obtained as the difference between the original input samples and the predicted samples, is further processed by DCT-based coding with a variable block size. For that, each leaf node of the prediction quadtree, which corresponds to a prediction block and its related residual signal, can be further split recursively into transform blocks of smaller size than the corresponding prediction block size. This recursive partitioning of a given prediction block into transform blocks is represented by the so-called *residual quadtree* (RQT). Figure 1 illustrates an example, where transform blocks and their corresponding RQTs are shown in dashed lines. Note that the transform block size that corresponds to the root node of a given RQT is identical to the size of the related prediction block, or equivalently, the leaf of the prediction quadtree, to which the RQT is associated.

For the purpose of mode decision or transmission of data associated with each block, all CTBs of a given slice or picture are traversed in raster scan order (left-to-right, top-down), and within each CTB, the subblocks are traversed in depth-first order. In Figure 1, the prediction blocks are traversed in alphabetical order. The transform blocks are traversed once a leaf associated with a prediction block is reached. Using depth-first traversal has the benefit that both

the left neighboring block(s) and the top neighboring block(s) are always encoded/transmitted before the current block. Thus, the data already transmitted for these blocks can be used to facilitate rate-constrained encoding of the current block such as, e.g., for the purpose of motion vector prediction, merging of prediction blocks, or context modeling in entropy coding.

IV. Motion-Compensated Prediction

As in most hybrid video coding designs, in the presented scheme a translational motion model is used for MCP. Thus, each MCP block is associated with one or two sets of motion parameters, where each set of motion parameters consists of a picture *reference index* and a translational *motion vector*. The prediction signal related to each set of motion parameters is obtained by displacing an area of a previously decoded reference picture selected by the reference index with the displacement being specified by the motion vector. When a MCP block is associated with two sets of motion parameters, i.e., in the bi-predictive case, the prediction signal is obtained as a superposition of the MCP signals that are generated using the individual sets of motion parameters. Both components of a motion vector are represented with the same fractional-sample accuracy. The minimum admissible motion-vector accuracy in our video coding scheme can be set to units of 2^{-n} the distance between luma samples, with the corresponding parameter $n \geq 0$ being signaled at the slice level. For the generation of the results in Section XI, the motion vector accuracy was kept fixed at quarter-sample precision.

A. Fractional-Sample Interpolation Using MOMS

Generating the prediction signal for motion vectors not pointing to an integer-sample position requires the use of a fractional-sample interpolation method. In order to achieve a higher quality of the prediction signal, numerous improvements to the H.264/AVC fractional-sample interpolation scheme [1] have been proposed, including adaptive interpolation filters as well as filters based on higher fractional-sample accuracy [11], switched interpolation filters for each fractional-sample position [12], or filters with larger support such as 8-tap or 12-tap filters that better approximate the ideal sinc interpolator.

Generalized interpolation using MOMS: In contrast to these aforementioned methods, our fractional-sample interpolation scheme is based on a new conceptual approach in approximation theory, so-called *generalized interpolation* [13]. According to this concept, families of *maximal-order-minimal-support* (MOMS) basis functions have been introduced that are asymptotically optimal in the sense of having smallest possible support for a given $\mathbf{L}^2$ approximation order. This outstanding behavior of MOMS functions, however, comes at the expense of the interpolating property and thus requires an additional prefiltering step for deriving the expansion coefficients.

Choice of MOMS basis functions: MOMS basis functions are constructed as integer shifts of a piecewise polynomial kernel function $\varphi(x)$ of degree L with support of size $L+1$, which is the smallest achievable support for *any* generalized interpolation function with approximation order of $L+1$ [13]. For our application case of fractional-sample interpolation, we have considered two members of the family of so-called O-MOMS (optimal MOMS) with interpolation kernels of degree $L = 3$ (cubic) and $L = 5$ (quintic).

Implementation aspects of cubic and quintic O-MOMS: The prefiltering step can be efficiently realized by separably applying a discrete 1D infinite impulse response (IIR) filter along rows and columns of the reconstructed picture. In the case of cubic or quintic O-MOMS, this IIR filter can be factorized into one or two sets of first-order causal and anti-causal recursive filters, respectively. Subsequent to this prefiltering stage, the actual interpolation is performed as a separable application of a 1D FIR filter with 4 taps in the cubic and 6 taps in the quintic case. Thus, in terms of required multiplication operations per fractional sample, an implementation of the cubic O-MOMS scheme including both prefiltering and interpolation is equivalent to a conventional 8-tap FIR interpolation scheme, at least when neglecting any symmetry of the filters involved. All filtering operations can be implemented using 16-bit fixed-point arithmetic without significant loss in coding efficiency. When compared to a 16-bit high-precision implementation of the H.264/AVC 6-tap interpolation filter as, e.g., being considered in [12], average bit rate savings of around 4% with maximum gains of up to 15% for individual test sequences have been achieved.

For more information on the specific aspect of fractional-sample interpolation in our video coding scheme, the reader is referred to [14].

B. Interleaved Motion-Vector Prediction

In order to reduce the bit rate required for transmitting the motion vectors, we have employed a novel concept in which the prediction and coding of the components of a motion vector is interleaved. According to this concept, in a first step, the vertical motion vector component is predicted using conventional median prediction (as in [1]), and the corresponding prediction residual, i.e., the difference between the actual vertical component and its prediction, is coded. Then, only those motion vectors of neighboring blocks for which the absolute difference between their vertical component and the vertical component for the current motion vector is minimized are used for the prediction of the horizontal motion-vector component. Interleaving the prediction of the motion-vector components and the actual coding of related residuals in such a way leads to an overall increased coding efficiency.

C. Merging of Motion-Compensated Predicted Blocks

Our concept of quadtree-based picture partitioning, as presented in Sec. III, is a flexible and computationally efficient instrument for adapting prediction and residual coding to the nonstationary statistical properties of the video signal.

However, in general, quadtree-based block partitioning may result in an over-segmentation due to the fact that, without any further provision, at each interior node of a quadtree, four subblocks are generated while merging of blocks is possible only by pruning complete branches consisting of at least four child nodes in the parent-child relationship within a quadtree. This suboptimal behavior has already been addressed in [15], [16] by introducing strategies for joining, i.e., merging

Fig. 2. Left: Schematic representation of a current block X, its set of merging candidates {A, B}, and its causal neighborhood (gray shaded). Middle: Cropped part (512×512) of a frame of the 1080p test sequence "Park Scene". Right: Illustration of MCP blocks (black lines), merged regions of MCP blocks (white lines), and intra-coded blocks (striped pattern) for the same cropped "Park Scene" content. With a chosen CTB size of 64×64, an area covering 64 CTBs is shown. Note that "Park Scene" contains a scene captured using a laterally from left to right moving camera and thus, the trees in the foreground, as shown in the middle image, appear to be moving to the left while the background park scene between the trees appears to be (slightly) moving to the right.

spatially neighboring blocks with the same coding parameters, even if they belong to different parent nodes in a tree-based hierarchy. In the spirit of these approaches, we have extended our quadtree-based picture partitioning by a *block merging process*. In our proposed coding approach, however, only MCP blocks are treated as candidates for merging and, in addition, merging is allowed not only for neighboring MCP blocks within a single prediction quadtree, i.e., within one single CTB but also across CTB boundaries. Merging of MCP blocks can also be considered as a generalization of the well-known concept of "skip" and "direct" coding modes in H.264/AVC, where the coding parameters of a given block are inferred from those of neighboring blocks.

Block merging in our proposal considers the two directly neighboring blocks of the top-left sample position in a given block as possible merging candidates. In Figure 2 (left), an illustrating example of such a situation is depicted, where the current block to be encoded is denoted by "X" and the neighboring candidate blocks for merging at the top and to left of "X" are denoted by "A" and "B", respectively. Note that blocks A and B are part of the causal neighborhood of block X such that the coding parameters for these blocks are already available. As already noted, merging can only be performed among MCP blocks and therefore, the set of *available merging candidates* can be a proper subset of {A, B} (which may also be true for blocks at picture/slice boundaries). When the set of available merging candidates is not empty, i.e., when it contains at least one MCP block, then it is signaled by the use of the so-called *merge flag* whether the current block X is to be merged with one block out of this set by inheriting its motion parameters. If the merge flag indicates a merging operation and the set of available merging candidates contains exactly two blocks with different motion parameters, another flag indicates whether merging is performed with block A or B.

Figure 2 (right) shows the result of quadtree-based block partitioning and merging for a selected picture of the test sequence "Park Scene". It can be seen that the chosen prediction quadtrees and corresponding prediction blocks (in black) as well as the merged regions (in white) are quite well adapted to the particular motion characteristics of the scene, as further described in the caption of Figure 2.

V. Spatial Intra Prediction

In contrast to H.264/AVC, the set of available spatial intra-prediction coding methods in our scheme does not depend on the underlying block size or the specific component (luma or chroma). For all prediction block sizes, eight directional intra-prediction modes and one additional averaging (DC) mode are available. These modes are straightforward generalizations of the related intra-prediction modes for 4×4 luma blocks in H.264/AVC.

In addition, an adaptive smoothing operation using the third order binomial filter can be applied to the reference samples before calculating the prediction signal. The application of this so-called *adaptive smoothing of intra prediction signals* is determined by the encoder and signaled in the bitstream by a separate flag for each intra-coded prediction block.

VI. Variable Block-Size Spatial Transforms and Quantization

As already described in Sec. III, each prediction block can be further subdivided for the purpose of transform coding with the subdivision being determined by the corresponding RQT. Transform block sizes in the range of 4×4 to 64×64 for the luma component and correspondingly scaled block sizes for both chroma components are supported. Note, however, that within these limits the maximum admissible RQT depth is variable and can be either signaled on a sequence parameter level, or can be constrained further by the use of appropriate profile or level limits. The transform kernel for each supported transform block size is given by a separable integer approximation of the 2D DCT-II (type-II Discrete Cosine Transform) of the corresponding block size.

The main idea for supporting variable block-size transforms is to adapt the transform to the varying space-frequency characteristics of the residual signal. DCT basis functions of larger spatial support (i.e., larger block size) provide a better

frequency resolution than those having small spatial support, whereas the latter have a better spatial resolution than the former. Trading off both aspects is the specific task of the encoder control and will be described in more detail in Sec. X.

For the quantization of transform coefficients, we have used uniform-reconstruction quantizers (URQs) similar to those specified in H.264/AVC [17]. As in H.264/AVC, for each picture/slice, one of 52 possible quantizer step size scaling factors is selected by using a quantization parameter (QP), and for each increment of six in the value of QP, there is a doubling in quantization step size.

In the encoder, the actual quantization is performed by using a method of rate-distortion optimized quantization (RDOQ) which is similar to [18] and to the Joint Model (JM) implementation, version 15 (and above) [19].

VII. Internal Bit Depth Increase

The internal bit depth d_i for generating the prediction signal for both intra prediction and MCP as well as for generating the reconstructed residual signal are increased relative to the given bit depth d_o of luma and chroma samples of the original input video signal. For that, the input samples and the samples of the reference pictures for MCP are left-shifted by $d_s = d_i - d_o$, and the increased internal bit depth d_i is retained until the reconstructed pictures are fed into the in-loop filtering process. For that, the reconstructed signal samples are right-shifted by d_s. This implies in particular that the reference pictures as output of the in-loop filtering are stored with d_o bits.

An internal bit depth of $d_i = 14$ bits was chosen. Note that test video material of the CfP was provided with $d_o = 8$ bits input sample precision for both luma and chroma component.

VIII. In-Loop Filtering

Our proposed video coding scheme utilizes two types of cascaded in-loop filters: a *deblocking filter* and a subsequently applied *quadtree-based, separable 2D Wiener filter*. While the former is intended to deal with blocking artifacts in the reconstructed pictures, the latter mainly aims at reducing additional quantization noise in the output of the deblocking filter. Both types of filter are highly adaptive, and they are both applied within the coding loop with the output of the final filtering stage being stored in the reference picture buffer.

A. Deblocking Filter

The deblocking filter is a straightforward extension of the one specified in H.264/AVC. The filtering operations are applied to samples at block boundaries of the reconstructed signal in the same way as in H.264/AVC with the only modification being an extension of the filtering process to larger transform blocks. The derivation of filter strength as well as the transmission of filter parameters is performed exactly as in H.264/AVC.

B. Quadtree-Based Separable 2D Wiener Filter

Subsequent to the application of the deblocking filter, a separable 2D Wiener filter is applied to selected regions of its output. Regions to which the Wiener filter is applied are represented by individual quadtree structures. The application of nonseparable, quadtree-based 2D Wiener filters in the context of video coding has already been proposed in [20]. The quadtree-based Wiener filter as part of our proposed video coding approach, is designed as a *separable filter* with the advantage of providing a better trade-off in computational cost vs. rate-distortion (R-D) performance compared to nonseparable Wiener filters [21].

For the derivation of the filter coefficients c_n of the Wiener filter $\mathbf{c}$, the unique solution of the Wiener-Hopf equation $\mathbf{R}_{rr} \cdot \mathbf{c} = \mathbf{r}_{rs}$ is calculated, where $\mathbf{R}_{rr}$ denotes the estimated autocorrelation matrix of the reconstructed signal $r(x, y)$ and $\mathbf{r}_{rs}$ denotes the estimated cross-correlation vector between $r(x, y)$ and the original signal $s(x, y)$. Note, however, that due to our separable approach, this derivation process is applied twice. First, the vertical filter coefficients c_n^v are calculated and then, after applying the vertical Wiener filter, the horizontal filter coefficients c_n^h are derived, based on the output of the vertically filtered signal. The lengths of the vertical and horizontal filter are chosen from the set {3,5,7,9,11} by minimizing the Lagrangian R-D cost functional $D + \lambda R$ and taking into account the rate for transmission of the filter coefficients c_n^v and c_n^h, respectively.

Given an initial set of estimated filter coefficients, a quadtree-based block partitioning is derived by using a simple tree-pruning strategy based on the Lagrangian R-D cost functional. Then, given the R-D optimal partitioning, the filter coefficients are re-estimated by adapting them to the blocks which have been marked for filtering. The steps of filter redesign and re-partitioning can be iterated with the jointly R-D optimized result being finally applied and signaled to the decoder. Note that within our proposed filtering approach, the option of estimating and signaling two independent quadtree-based partitionings for vertical and horizontal filtering is supported [21].

IX. Entropy Coding

For entropy coding, a variation of CABAC [22] is employed. Binarization and context modeling are basically the same as in CABAC of H.264/AVC, except from a few modifications and additions as further explained below. However, the actual coding of binary decisions, so-called *bins*, is based on the novel concept of *probability interval partitioning entropy* (PIPE) coding that has been introduced in order to support parallelized implementations of entropy encoding and decoding as well as for decreasing the computational cost of entropy decoding [23], [24].

Other approaches to entropy coding with some degree of similarity to PIPE coding were presented in [25], [26], [27], [28]. As a major conceptual difference to these related papers, we are introducing and making use of the probability interval partitioning principle, which allows a decoupling of probability modeling and actual coding and thus, has some important implications in terms of applicability. For instance, without using that principle, the design and application of a large number of sets with individually tuned prefix code tables (see Sec. IX-A above) may be required. Typically, this number will be at least as large as the number of different

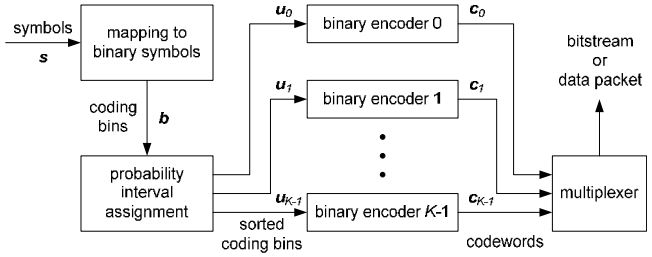

Fig. 3. Overview of the PIPE coding structure.

probability states that are used by the probability estimation process, which, e.g., in the case of CABAC is equal to 64 [22].

A. Probability Interval Partitioning Entropy Coding

In Figure 3, the basic PIPE coding concept is illustrated. When a syntax element or symbol does not already represent a binary syntax element, it is first binarized, i.e., it is mapped onto a sequence of bins. For each bin a context is selected. A context represents a (binary) probability model for a class of bins; it is characterized by the probability and the value of the less probable bin (LPB). As in H.264/AVC, the LPB probability is represented by one out of 64 states. At the beginning of the encoding/decoding of a slice, the probability models are initialized using fixed values (as in CABAC of H.264/AVC). Then, after encoding/decoding a bin with a particular model, the probability and LPB value of the model is updated. The probability model update, i.e., the probability estimation process is the same as in CABAC of H.264/AVC. The association of a bin with a context model is also similar as in CABAC of H.264/AVC. It depends on the syntax element, the bin number, and, for some bins, the values of neighboring syntax elements.

With the binarization and association of bins with context models being basically the same as in CABAC of H.264/AVC, the main difference is given by the step of transforming bin into bits and vice versa. Instead of directly applying a binary arithmetic coder to the bins as in CABAC of H.264/AVC, the estimated LPB probabilities are quantized, i.e., they are mapped onto a small number K of LPB probability intervals. For each of these K probability intervals a separate *bin encoder/decoder* (bin codec) is operated. In our implementation, we use $K = 12$ probability intervals and thus 12 separate bin codecs. Each bin codec operates at a fixed LPB probability, which can be considered as the representative probability for an LPB interval. The selection of a bin codec is implemented via a look-up table that associates each of the 64 state indices for the LPB probability with a unique bin codec. Hence, the 64 states that are used for estimating the LPB probability of a context model are mapped onto 12 probability intervals, for each of which a separate bin codec is operated. For bin encoders and decoders, two alternatives have been implemented as follows.

PIPE Coding Using Arithmetic Codes: In a first PIPE coding version, the K bin encoders and decoders represent binary arithmetic encoding and decoding engines, respectively, which are similar to the M coder used in CABAC [22]. The corresponding K arithmetic codewords are written to different partitions of the bitstream with the corresponding partitioning information being transmitted in the slice header. An obvious advantage of this approach is that this way, binary arithmetic decoding can be parallelized. For instance, when operating all of the K arithmetic decoding engines in parallel, the corresponding sequences of bins can be written into K separate bin buffers. The remaining entropy decoding process can then simply read the bins from the corresponding bin buffers without the need to wait until a bin is arithmetically decoded before proceeding with the next bin.

TABLE I
V2V CODE EXAMPLE FOR A PROBABILITY OF $p = 0.25$. NOTE THAT IN THE BIN SEQUENCE COLUMN, "0" REPRESENTS THE MORE PROBABLE BIN, WHILE "1" REPRESENTS THE LESS PROBABLE BIN.

bin sequence	probability for bin seq.	codeword
$'0'$	$0.25^1 = 0.2500$	$'10'$
$'11111'$	$0.75^5 = 0.2373$	$'11'$
$'110'$	$0.75^2 \cdot 0.25^1 = 0.1406$	$'001'$
$'101'$	$0.75^2 \cdot 0.25^1 = 0.1406$	$'010'$
$'1110'$	$0.75^3 \cdot 0.25^1 = 0.1055$	$'011'$
$'11110'$	$0.75^4 \cdot 0.25^1 = 0.0791$	$'0000'$
$'100'$	$0.75^1 \cdot 0.25^2 = 0.0469$	$'0001'$

PIPE Coding Using V2V Codes: A second version of PIPE coding uses prefix codes. For that, a variable number of bins is mapped onto variable-length codewords (also denoted as variable-to-variable (V2V) codes) and vice versa. As an example, Table I shows a V2V code that was designed for a representative LPB probability of $p = 0.25$ with the constraint of considering up to 8 leaf nodes, i.e., codeword entries in the corresponding V2V table. Note that, in general, it is possible to get closer to the entropy limit when the V2V table size is increased. The V2V code as shown in Table I has a redundancy of only 0.44% relative to the entropy limit for the corresponding probability. In order to minimize the redundancy of the overall design, the probability interval partitioning and the V2V codes can be jointly optimized [24].

The partial bitstreams that are generated by the two versions of bin encoders can be written to different partitions of a bitstream or they can be interleaved into a single bitstream.

Both PIPE coding versions have similar coding efficiency. For the generation of our CfP submitted bitstreams, the first version, i.e., the arithmetic coding based version of PIPE was used. Note, however, that lossless transcoding between the bitstreams of both PIPE versions was possible without exceeding the target bit rates given in the CfP [5].

B. Context Modeling Scheme for Larger Transform Blocks

CABAC transform coefficient coding was originally designed for 4×4 blocks and has been extended to 8×8 transform blocks in the specification of the H.264/AVC High Profiles. For transform block sizes greater than 8×8, we have extended the original CABAC context modeling scheme by taking into account specific observations we have made when analyzing the statistics of larger transform blocks [29]. The main elements of this extended context modeling scheme can be summarized as follows.

Context models for the syntax element indicating significant, i.e., nonzero transform coefficient levels are selected based on already coded values for neighboring transform

coefficients. Furthermore, the significance map is coded using a backward-adaptive scanning pattern. For coding absolute values of transform coefficient levels, transform blocks larger than 4×4 are partitioned into 4×4 subblocks and for each of these subblocks a set of context models is selected based on the absolute values of already transmitted 4×4 subblocks of the same transform block. The context modeling inside such a 4×4 subblock is the same as in the original CABAC. For more details, the reader is referred to [29].

X. Encoder Control

When configured like H.264/AVC, i.e., with a chosen CTB size of 16×16 and a maximum prediction quadtree depth $D = 2$, the encoder control of the presented video coding scheme is (at least in terms of computational complexity) comparable to that used in the JM or JSVM implementation [19], [30]. This is true, even though H.264/AVC, in contrast to our quadtree-based approach, has many restrictions about what combinations of block sizes and prediction modes are allowed for a 16×16 macroblock and how the residual may be subdivided for transform coding. Since this kind of ad-hoc limitations do not exist in the presented video coding approach, in principle, a large number of admissible combinations can be tested for obtaining further R-D improvements. Note that, e.g., the number of possible partitionings for prediction alone exceeds $2^{4^{D-1}}$ and thus, for a more realistic configuration with an CTB size of 64×64 and $D = 4$, more than 2^{64} partitionings need to be considered for selecting the optimal one. At least for this example, a brute force exhaustive search is clearly not feasible.

A. Application of a Fast Optimal Tree Pruning Algorithm

Fortunately, the problem of finding the optimal partition can be efficiently solved by application of a fast optimal tree search algorithm, also known as G-BFOS algorithm (generalized version of an algorithm introduced by Breiman, Friedman, Olshen, and Stone) [31], [32]. The G-BFOS algorithm can be briefly summarized as follows.

In a first step, a full tree is grown by starting from root and populating each node up to some pre-defined depth. Then, in a second, recursively performed step, a pruning decision is made at each internal node starting from the parent nodes of the leaves at the bottom and moving up to the root of the tree. Given a cost functional J (with certain properties as specified in [31]), pruning at a parent node is performed whenever the criterion $J(\text{parent node}) \leq \sum J(\text{child node})$ is fulfilled; otherwise, the sum of the costs of its child nodes, i.e., the value of the right-hand side of the inequality is assigned to the left-hand side $J(\text{parent node})$. The recursion terminates after the pruning decision at the root node is completed, and the resulting pruned tree is the optimal subtree in terms of minimum cost, which is obtained as $J(\text{root node})$. Note that the application of the G-BFOS algorithm requires only as much pruning decisions as given by the number of internal nodes of a tree, which, e.g., for the above mentioned prediction quadtree of maximum depth $D = 4$ amounts to 53 pruning decisions for selecting the optimal partition out of more than 2^{64} partitions.

Being equipped with the G-BFOS algorithm, our proposed encoder control process performs the task of finding the *best coded representation* of a given CTB in the sense of minimizing the Lagrangian R-D cost functional $J = D + \lambda R$ over all possible choices of prediction modes for all prediction block sizes *and* all corresponding transform block sizes. According to the nested structure of prediction quadtree and residual quadtrees of an CTB, this process requires a nested and intertwined application of the G-BFOS algorithm as follows.

B. Mode Decision Process

At the heart of our encoder control is the well-known mode decision process for deriving a prediction and residual coding mode for a given prediction block of fixed size [33]. In principle, this process is similar to that of the JM or JSVM, where out of a set P of competitive coded representations of the given block, the R-D optimal one is selected by minimizing the Lagrangian cost functional $D(p) + \lambda_{\text{QP}} \cdot R(p)$ over all $p \in P$. Note that the parameter λ_{QP}, as indicated by the subscript, was derived by using a fixed relationship between λ and the QP [34], [35]. The main distinction as compared to the JM/JSVM is given by the fact that for each prediction mode to be evaluated, the following residual quadtree pruning process is invoked.

C. Residual Quadtree Pruning Process

Embedded into the mode decision process, the residual quadtree pruning process derives the R-D optimal residual quadtree by application of the G-BFOS algorithm. Given a residual signal for a fixed prediction mode and corresponding prediction block size, the G-BFOS algorithm first generates a full tree of transform blocks with the maximum transform block size being determined by the given prediction block size and the minimum transform block size being given per sequence. Based on the Lagrangian R-D cost functional, the bottom-up pruning process, as described in Sec. X-A, then generates the optimal partitioning into transform blocks of potentially varying size. Note that the underlying assumption that the bit rate spent for encoding a particular transform block is independent of other transform blocks is not fully justified. Due to context modeling and probability estimation, neighboring blocks do have an influence on each other, though these effects are typically marginal.

It is also worth noting that this process of selecting optimal DCT basis functions of variable size is conceptually similar to the well-known *best basis* algorithm for generating optimal signal-adapted time/space-frequency tilings [36].

D. Prediction Quadtree Pruning Process

The all-embracing process of our encoder control is given by the prediction quadtree pruning process. Based on the outcome of the mode decision process for each admissible prediction block size, the R-D optimal quadtree-based subdivision of a CTB into prediction blocks is determined by applying the bottom-up G-BFOS pruning algorithm for the given CTB size and prediction quadtree depth D. Note that due to reasons already explained in Sec. III, the prediction quadtree is grown by traversing in depth-first order.

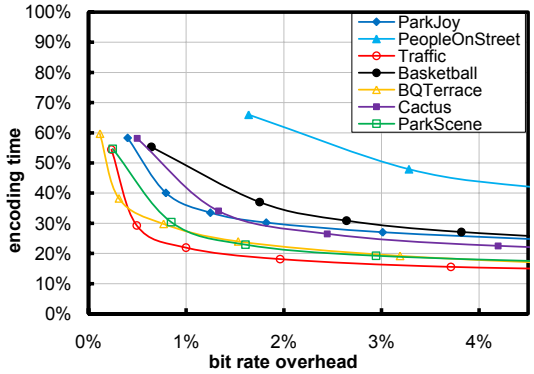

Fig. 4. Trade-off between encoding time and bit rate overhead in percentage relative to that of the R-D optimal result by applying the early termination strategy with different choices of thresholds for a set of 1080p test sequences.

E. Early Termination Strategy

In practice, it is often desirable to trade-off computational complexity and R-D performance in a flexible and configurable way. To this end, we have combined the G-BFOS-based top-to-bottom tree-growing stage with an early termination strategy.

During the depth-first tree-growing stage of the prediction quadtree pruning process, as described in Sec. X-D, the following *abort criterion* is applied to each prediction block, i.e., each node of the prediction quadtree. If the absolute values of all transform coefficients (before quantization) of the non-subdivided representation of the residual signal are below a threshold T_{sub} and are quantized to zero, no further subdivision of this node of the prediction quadtree is generated.

In addition to this abort criterion for the prediction quadtree growing process, a further early termination rule was implemented for the prediction mode decision process. According to that rule, the MCP mode is tested generally before testing the intra-prediction modes. Then, if the absolute values of all transform coefficients (before quantization) for MCP are below a threshold T_{mode} and are quantized to zero, no further testing of intra-prediction modes is performed.

With the choice of the two thresholds T_{sub} and T_{mode}, the trade-off between encoder complexity (in terms of run time) and R-D performance can be continuously adjusted. For the results of Section XI, we have controlled both thresholds by a simple linear relationship with the QP. This led to sub-optimal R-D results with an average increase in bit rate of around 2–3%, but with a notable decrease in encoder run time of 70–85%. An example for demonstrating this behavior when applied to a set of 1080p test sequences is depicted in Figure 4.

XI. Coding Conditions and Results

In the CfP [6], two sets of coding conditions with different constraints are defined. A random access case, denoted as constraint set 1 (CS 1) and a low delay case, denoted as constraint set 2 (CS 2). For CS 1, the structural delay is limited to 8 pictures with random access intervals not exceeding 1.1 sec. According to those constraints, we used for the generation of our submitted CS 1 bitstreams a hierarchical B picture coding structure [37] with 4 layers and a corresponding intra frame period. For CS 2, a structural delay is not allowed and random access capabilities are not required. Hence, we used hierarchical P frames without picture reordering for our submitted CS 2 bitstreams with only one intra picture at the beginning of each sequence. For both constraint sets, we configured our encoder to operate with a fixed CTB size of 64×64 (for luma) and a maximum prediction quadtree depth of $D = 4$.

For motion estimation and temporal layer dependent QP scaling (i.e., QP cascading), our encoder was configured in a way comparable to the JM encoder for generating the H.264/AVC conforming anchor bitstreams, as specified in [6].

TABLE II
Averaged BD rate savings (in percentage) relative to the H.264/AVC HP anchors at CS 1 and CS 2 test conditions.

Class	Sequence	BD Rate CS 1 [%]	BD Rate CS 2 [%]
A (2560x1600)	Traffic	-27.92	n/a
	People	-17.92	n/a
	Average	**-22.92**	n/a
B1 (1920x1080)	Kimono	-38.30	-38.11
	ParkScene	-24.28	-20.85
	Average	**-31.29**	**-29.84**
B2 (1920x1080)	Cactus	-29.22	-23.43
	BasketballDrive	-35.97	-34.42
	BQTerrace	-41.92	-31.45
	Average	**-35.70**	**-29.77**
C (832x480)	BasketballDrill	-31.88	-14.74
	BQMall	-29.71	-31.45
	PartyScene	-28.09	-16.55
	RaceHorses	-29.67	-22.39
	Average	**-29.84**	**-21.28**
D (416x240)	BasketballPass	-21.97	-14.38
	BQSquare	-43.96	-13.68
	BlowingBubbles	-23.60	-7.44
	RaceHorses	-19.64	-14.23
	Average	**-27.29**	**-12.43**
E (1280x720)	Vidyo 1	n/a	-28.49
	Vidyo 3	n/a	-22.58
	Vidyo 4	n/a	-28.65
	Average	n/a	**-26.57**
	Total Average	**-29.60**	**-22.68**

A. Objective Performance

Table II shows averaged bit rate savings for each of our CS 1 and CS 2 related bitstreams, as submitted to the CfP, relative to the H.264/AVC High Profile (HP) anchors [6]. For each of the test sequences, the averaged bit rate savings are obtained as mean of the Bjøntegaard delta (BD) bit rate values [38] for the upper four and lower four out of a total of five rate points. Overall, significant objective gains in terms of average 29.6% BD rate savings for CS 1 and 22.7% BD rate savings for CS 2 relative to the H.264/AVC HP anchors have been achieved. In general, the dominant part of these gains can be attributed to the structural design elements of this proposal, which are given by the quadtree-based block partitioning concept for improved prediction and residual coding as well as the block merging scheme for efficient region-based motion representation.

In a set of additional coding simulations, we have evaluated the intra-only coding performance of our proposed video scheme by using the two well-known grayscale test images “Lena” and “Barbara”. As a reference, we have used the JM configured to produce H.264/AVC HP bitstreams in intra-only coding mode and the Kakadu software (version 5.1) for JPEG2000, Part 1 [39]. The JM encoder was operated in a configuration similar to the one used for the H.264/AVC HP

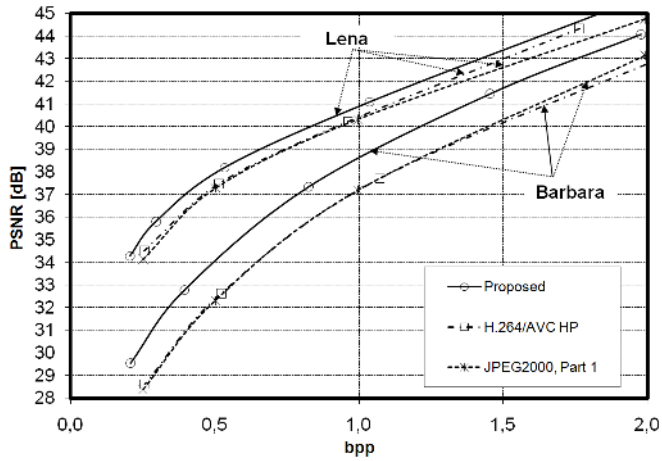

Fig. 5. R-D performance in terms of bits per pixel (bpp) vs. PSNR (in dB) for intra-only coding using the popular grayscale test images "Lena" and "Barbara", both with 512×512 pixels. The three upper R-D curves are related to "Lena" and the 3 lower curves belong to "Barbara". Comparison is made between the proposed video scheme, H.264/AVC HP, and JPEG2000.

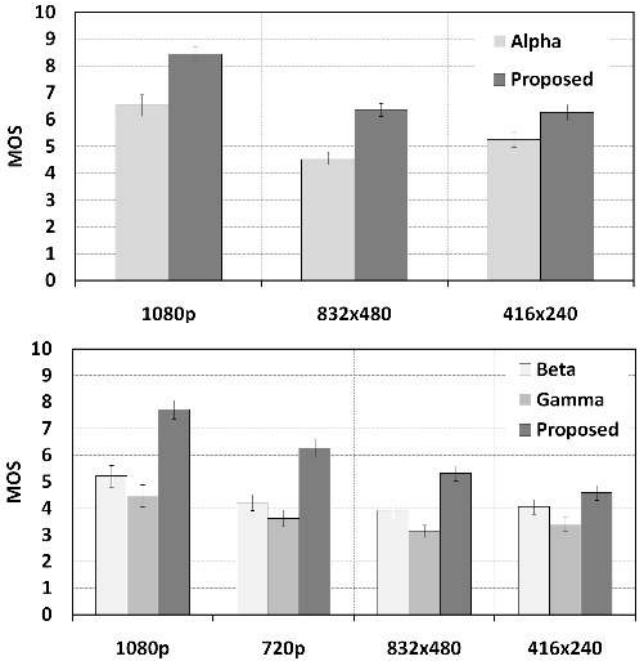

Fig. 6. Subjective test results in terms of average MOS values for different resolutions. Comparison of results related to CS 1 (top) and CS 2 (bottom).

anchors and the JPEG2000 Kakadu software was used with the default encoding options for maximizing the R-D performance. The proposed video coding scheme was configured to be operated in intra-only coding mode with the same CTB parameters as described above.

The results of these intra-only coding simulations are shown in the R-D diagram of Figure 5. While JM-based H.264/AVC HP and Kakadu-based JPEG2000 are performing on a comparable R-D level, the presented intra coding shows significant PSNR gains of 0.5–1.0 dB for "Lena" and 1.0–2.0 dB for "Barbara" with the tendency of larger gains at lower bit rates. Especially, the gains achieved for "Barbara" can be attributed to a large extend to the residual quadtree (RQT) and its capability to better adapt to the specific space-frequency characteristics of the given signal.

B. Subjective Performance

All 27 submitted proposals to the CfP were evaluated in a subjective test together with two H.264/AVC HP anchors, denoted as "Alpha" (satisfying CS 1) and "Beta" (satisfying CS 2), and an additional "Gamma" anchor satisfying CS 2 and conforming to H.264/AVC Constrained Baseline Profile. The corresponding results have been published in [40]. Figure 6 shows the results of those tests for the proposed video coding scheme in comparison to the particular anchors. The bars in both diagrams illustrate the average Mean Opinion Score (MOS) for each test class with separate diagrams for CS 1 and CS 2. On the top of each bar, the related 95% confidence intervals are shown.

It can be seen that the proposed coding scheme achieved significantly higher scores than the anchors, typically with higher gains at higher resolutions. Except for the low-resolution test class D, the average gain on the doubled MOS scale is in the range of 1.4–2.5, meaning an average increase in perceived quality of roughly one ordinary MOS value relative to the Alpha and Beta anchors. Overall, the proposed video coder was among the best rated proposals in the subjective tests [7], [40].

C. Computational Complexity

As a rough measure of computational complexity, the encoding and decoding time has been measured for both our implementation and the JM software using the same hardware platform. By averaging over all rate points of all CS 1 and CS 2 bitstreams, we obtained a factor of around 4 in encoding time relative to JM version 16.2 and roughly a factor of 3–4 in decoding time relative to JM version 17.0 .

XII. Conclusion

We have presented our proposed video compression design for the next-generation video coding standard. Its most notable features are given by a flexible partitioning for prediction and residual coding, a block merging process for efficient region-based motion modeling, a highly efficient fractional sample interpolation method, a computationally efficient adaptive in-loop filter, and a conceptually new approach to entropy coding.

Appendix
Downloadable Resources Related to this Paper

The JCT-VC document [5] describing the video coding technology related to this paper is publicly available and can be downloaded (together with all other JCT-VC documents) at http://wftp3.itu.int/av-arch/jctvc-site in the "2010_04_A_Dresden" folder. Note that the archive file "JCTVC-A116.zip" contains also software, configuration files, and further material of the CfP submission related to this paper. All cited VCEG and JVT documents are also publicly available and can be downloaded at http://wftp3.itu.int/av-arch in the "video-site" and "jvt-site" folder, respectively.

Acknowledgment

The authors would like to thank Heribert Brust, Philipp Merkle, Hunn F. Rhee, and Gerhard Tech for their valuable help in preparing the bitstreams of our CfP submission. The authors would also like to thank the reviewers for their insightful and constructive comments.

REFERENCES

[1] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," *ITU-T Rec. H.264* and *ISO/IEC 14496-10 (MPEG-4 AVC)*, Version 8, July 2007.

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

[3] J. Ostermann, J. Bormans, P. List, D. Marpe, N. Narroschke, F. Pereira, T. Stockhammer, T. Wedi, "Video Coding with H.264/AVC: Tools, Performance and Complexity," *IEEE Circuits and Systems Magazine*, vol. 4, no. 1, pp. 7-28, April 2004.

[4] D. Marpe, T. Wiegand, G. J. Sullivan, "The H.264/MPEG4 Advanced Video Coding Standard and its Applications," *IEEE Communications Magazine*, Standards Report column series, pp. 134–143, August 2006.

[5] M. Winken, S. Boße, B. Bross, P. Helle, T. Hinz, H. Kirchhoffer, H. Lakshman, D. Marpe, S. Oudin, M. Preiß, H. Schwarz, M. Siekmann, K. Sühring, T. Wiegand, "Description of video coding technology proposal by Fraunhofer HHI," *Doc. JCTVC-A116*, Dresden, Germany, April 2010.

[6] ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/16, "Joint Call for Proposals on Video Compression Technology," *WG11 Doc. N11113* and *ITU-T Q6/16 Doc. VCEG-AM91*, Kyoto, Jan. 2010.

[7] G. J. Sullivan, J.-R. Ohm, "Meeting Report of the First Meeting of the Joint Collaborative Team on Video Coding (JCT-VC), Dresden, DE, 15–23 April, 2010," *Doc. JCTVC-A200*, April 2010.

[8] JCT-VC, "Test Model under Consideration," *Doc. JCTVC-A205*, April 2010.

[9] J.-R. Ohm *et al.* "Special Section on the Joint Call for Proposals on High Efficiency Video Coding (HEVC) Standardization," *IEEE Transactions on Circuits and Systems for Video Technology*, this issue.

[10] G. J. Sullivan, J.-R. Ohm, "Recent Developments in Standardization of High Efficiency Video Coding (HEVC)," *Proc. SPIE*, vol. 7798, Aug. 2010.

[11] T. Wedi, "Adaptive Interpolation Filters and High-Resolution Displacements for Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 4, pp. 484–491, April 2006.

[12] M. Karczewicz, Y. Ye, P. Chen, "Switched Interpolation Filter with Offset," *ITU-T Q6/16 Doc. COM16-C463*, April 2008.

[13] T. Blu, P. Thévenaz, M. Unser, "MOMS: Maximal-Order Interpolation of Minimal Support," *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 1069–1080, July 2001.

[14] H. Lakshman, B. Bross, H. Schwarz, T. Wiegand, "Fractional-Sample Motion Compensation Using Generalized Interpolation," *Proc. Picture Coding Symposium* 2010 (PCS 2010), Dec. 2010.

[15] R. Shukla, P. L. Dragotti, M. N. Do, M. Vetterli, "Rate-Distortion Optimized Tree-Structured Compression Algorithms for Piecewise Polynomial Images," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 343–359, March 2005.

[16] R. De Forni, D. S. Taubman, "On The Benefits of Leaf Merging in Quad-Tree Motion Models," *Proc. IEEE Int. Conf. Image Proc.*, 2005.

[17] G. J. Sullivan, S. Sun, "On Dead-Zone Plus Uniform Treshold Scalar Quantization," *Proc. SPIE, vol. 5960, no. 1*, 2005.

[18] M. Karczewicz, Y. Ye, I. Chong, "Rate-Distortion Optimized Quantization," *ITU-T Q6/16 Doc. VCEG-AH21*, Jan. 2008.

[19] JVT of ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/16, Joint Model (JM) – H.264/AVC Reference Software, http://iphome.hhi.de/suehring/tml/.

[20] T. Chujoh, N. Wada, G. Yasuda, "Quadtree-Based Adaptive Loop Filter," *ITU-T Q6/16 Doc. COM16-C181*, Geneva, January 2009.

[21] M. Siekmann, S. Bosse, H. Schwarz, T. Wiegand, "Separable Wiener Filter Based Adaptive In-Loop Filter for Video Coding," *Proc. Picture Coding Symposium* 2010 (PCS 2010), Dec. 2010.

[22] D. Marpe, H. Schwarz, T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in H.264 / AVC Video Compression Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, July 2003.

[23] D. Marpe, H. Schwarz, T. Wiegand, "Novel Entropy Coding Concept," *Doc. JCTVC-A032*, Dresden, Germany, April 2010.

[24] D. Marpe, H. Schwarz, T. Wiegand, "Entropy Coding in Video Compression Using Probability Interval Partitioning," *Proc. Picture Coding Symposium* 2010 (PCS 2010), Dec. 2010.

[25] F. Ono, S. Kino, M. Yoshida, T. Kimura, "Bi-Level Image Coding With MELCODE - Comparison of Block Type Code and Arithmetic Type Code," *Proc. IEEE GLOBECOM '89*, vol. 1, pp. 255–260, 1989.

[26] M. Boliek, J. D. Allen, E. L. Schwartz, M. J. Gormish, "Very High Speed Entropy Coding," *Proc. IEEE International Conference on Image Processing*, 1994.

[27] H. S. Malvar, "Fast Adaptive Encoder for Bi-Level Images," *Proc. Data Compression Conference*, 2001.

[28] D. He, G. Korodi, G. Martin-Cocher, E.-h. Yang, X. Yu, J. Zan, "Video coding technology proposal by RIM," *Doc. JCTVC-A120*, Dresden, Germany, April 2010.

[29] T. Nguyen, H. Schwarz, H. Kirchhoffer, D. Marpe, T. Wiegand, "Improved Context Modeling for Coding Quantized Transform Coefficients in Video Compression," *Proc. Picture Coding Symposium* 2010 (PCS 2010), Dec. 2010.

[30] JVT of ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/16, Joint Scalable Video Model (JSVM) – SVC Reference Software, http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm.

[31] P. A. Chou, T. Lookabaugh, R. M. Gray, "Optimal Pruning with Applications to Tree-Structured Source Coding and Modeling," *IEEE Transactions on Information Theory*, vol. 35, no. 2, pp. 299–315, March 1989.

[32] G. J. Sullivan, R. L. Baker, "Efficient Quadtree Coding of Images and Video," *IEEE Transactions on Image Processing*, vol. IP-3, no. 3, pp. 327–331, May 1994.

[33] T. Wiegand, M. Lightstone, D. Mukherjee, T. G. Campbell, S. K. Mitra, "Rate-Distortion Optimized Mode Selection for Very Low Bit Rate Video Coding and the Emerging H.263 Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 2, pp. 182–190, April 1996.

[34] G. J. Sullivan, T. Wiegand, "Rate-Distortion Optimization for Video Compression," *IEEE Signal Processing Magazine*, pp. 74–90, Nov. 1998.

[35] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, G. J. Sullivan, "Rate-Constrained Coder Control and Comparison of Video Coding Standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 688–703, July 2003.

[36] V. M. Wickerhauser, *Adapted Wavelet Analysis from Theory to Software*, A. K. Peters, 1994.

[37] H. Schwarz, D. Marpe, T. Wiegand, "Hierarchical B pictures," *Joint Video Team, Doc. JVT-P014*, Poznan, Poland, July 2005.

[38] G. Bjøntegaard, "Calculation of Average PSNR Differences between RD Curves, *ITU-T Q6/16 Doc. VCEG-M33*, April 2001.

[39] ITU-T and ISO/IEC, "JPEG2000 Image Coding System: Core Coding System (JPEG2000 Part 1)," *ITU-T Rec. T.800* and *ISO/IEC 15444-1*, 2000.

[40] V. Baroncini, J.-R. Ohm, G. J. Sullivan, "Report of Subjective Test Results of Responses to the Joint Call for Proposals on Video Coding Technology for High Efficiency Video Coding (HEVC)," *Doc. JCTVC-A204*, April 2010.