BLOCK MERGING FOR QUADTREE-BASED VIDEO CODING

S. Oudin[1], P. Helle[1], J. Stegemann[1], C. Bartnik[1], B. Bross[1], D. Marpe[1], H. Schwarz[1], and T. Wiegand[1,2]

[1]Image Processing Department
Fraunhofer Institute for Telecommunications - Heinrich Hertz Institute
Einsteinufer 37, 10587 Berlin, Germany

[2]Image Communication Chair
Department of Telecommunication Systems
Technical University of Berlin
Einsteinufer 17, 10587 Berlin, Germany

ABSTRACT

Quadtree-based block partitioning together with motion-compensated prediction has proven to be an efficient approach in video compression. However, when dealing with spatially neighboring blocks in uniformly displaced regions, quadtree-based partitioning may lead to redundant sets of transmitted motion parameters. This paper proposes and describes a simple but efficient block merging algorithm that aims at removing those redundancies by using only a single parameter set for a whole motion-compensated region of contiguous blocks. Simulation results show that our proposed merging technique works more efficiently than the conceptually similar direct mode as, e.g., specified in H.264/AVC. Due its efficiency and simplicity, our proposed merging approach has been adopted into the first test model of the high efficiency video coding (HEVC) standardization project, as currently pursued by ITU-T VCEG and ISO/IEC MPEG.

Index Terms— motion compensation, quadtree partition, block merging, H.265, HEVC

1. INTRODUCTION

In image and video coding, pictures and their corresponding sets of sample arrays are usually decomposed into blocks such that each block is associated with a particular set of coding parameters. In H.264/AVC [1], e.g., each block is either spatially or temporally predicted and the resulting prediction residual is represented by using transform coding. Temporal or inter-picture prediction typically uses samples of one or two displaced blocks in previously decoded and reconstructed reference pictures for the samples of a given picture block. The motion vectors corresponding to these displacements are typically coded in a differential way by using an appropriately specified motion-vector predictor. Thus, without any further provision, one or two sets of motion parameters have to be signaled for each inter-predicted block as motion information with each set consisting of a motion-vector difference and an index specifying the corresponding reference picture.

In H.264/AVC, macroblocks or sub-macroblocks of bi-predictively coded areas can also be coded in so-called *direct mode* without the need of transmitting any additional motion information [1, 2]. The direct mode infers the motion information for prediction from previously decoded blocks. In addition, if no prediction residual signal is transmitted for a direct-coded macroblock, it is also referred to as *skipped*, and skipped macroblocks can be indicated very efficiently as for the *skip mode* in H.264/AVC.

In the present paper, we introduce a coding tool called *block merging* which is conceptually similar to the direct mode of H.264/AVC. Block merging has been designed as part of a novel quadtree-based video coding approach [3, 4]. It exploits the spatial redundancy of motion information in neighboring blocks belonging to potentially different branches and levels in the quadtree hierarchy and it operates in a way that is conceptually similar to what has already been introduced in previous related publications [5, 6, 7].

The presented block merging algorithm has been first introduced as a feature of the Fraunhofer HHI response [3] to the joint Call for Proposals (CfP) on high efficiency video coding (HEVC) technology that was issued by ITU-T VCEG and ISO/IEC MPEG [8]. The intention of the CfP was to identify video coding technology with substantially higher compression capabilities than the existing H.264/AVC standard.

The rest of the paper is organized as follows. The next section briefly introduces the HHI proposal for HEVC, also called *HHI video codec* in the sequel. Section 3 deals with quadtree based partitioning and its drawbacks. The proposed block merging algorithm is described in detail in Section 4. Conducted experiments and their results are presented in Sections 5 and 6. Section 7 concludes this paper.

2. OVERVIEW OF THE HHI VIDEO CODEC

The HHI video codec is based on the hybrid video coding approach using temporal and spatial prediction followed by transform coding of the residual and entropy coding [3, 4]. In this video codec, the size of the prediction blocks can be adaptively chosen by using a quadtree-based partitioning. The block size used for DCT-based coding of the residual signal is also derived by using quadtree-based partitioning of the corresponding prediction block. The concept of a macroblock

978-1-61284-350-6/11/$26.00 ©2011 IEEE

Amazon.com, Inc., Amazon.com Services LLC - Ex. 1060, Page 000001
IPR2024-01505 (Amazon.com, Inc., Amazon.com Services LLC v. Nokia Technologies Oy)

as the basic processing unit in standardized video coding [1] is generalized to what is called a *coding tree block* (CTB). Fig. 1(a) shows an example of a coding tree block for luma samples (solid bold outline) and how it is subdivided into prediction blocks (solid lines) and transforms blocks (dashed lines). In Fig. 1(b), the corresponding quadtree structure for CTB partitioning is shown. Each node of the so-called prediction quadtree, as shown in solid lines in the example of Fig. 1(b), which corresponds to a prediction block and its related residual signal, can be further split into transform blocks of smaller size than the given prediction block size. In the example of Fig. 1, this is illustrated by the dashed blocks and the corresponding dashed nested residual quadtrees.

Block merging, as further described below, enables the creation of regions composed of neighboring prediction blocks with identical motion information. This motion information is composed of one or two sets of a motion-vector difference and an index specifying the corresponding reference picture. However, the dimensions of the transform blocks for residual coding are still constrained by corresponding residual quadtree partition, independent of the merging operation. The direct mode which is used in our simulation as a reference is similar to the spatial one specified in H.264/AVC [1]. This spatial direct mode can be applied to inter-partitions in pictures using multi-hypothesis prediction. For these inter-partitions, a flag is signaled indicating whether the partition is coded in direct mode or not. Interleaved Motion Vector Predictor (IMVP), as introduced in [3, 4], interleaves the prediction and coding for the components of a motion vector. More precisely, it uses the vertical component in order to predict its horizontal component. In the HHI video codec, no explicit skip mode is used. However, for each leaf in the residual quadtree, there is a flag which signals explicitly the presence of non-vanishing residual coefficients in the corresponding block. For more details of the HHI video codec, the reader is referred to [3, 4].

3. BACKGROUND AND PROBLEM FORMULATION

The quadtree data structure provides a suitable framework in video compression for optimizing the tradeoff between model accuracy (e.g., given by distortion D) and model coding cost, typically measured in bit rate R. The existence of simple tree pruning algorithms optimizing a Lagrangian functional

$$D + \lambda * R,$$

allows the development of compression algorithms which are based on quadtrees and related representations. The HHI video codec is a quadtree-structured video coding approach with syntax elements carrying the subdivision information for blocks of different types. Such syntax elements may be related to subdivision of blocks for the purpose of prediction or transform coding, as already outlined in Section 2 above.

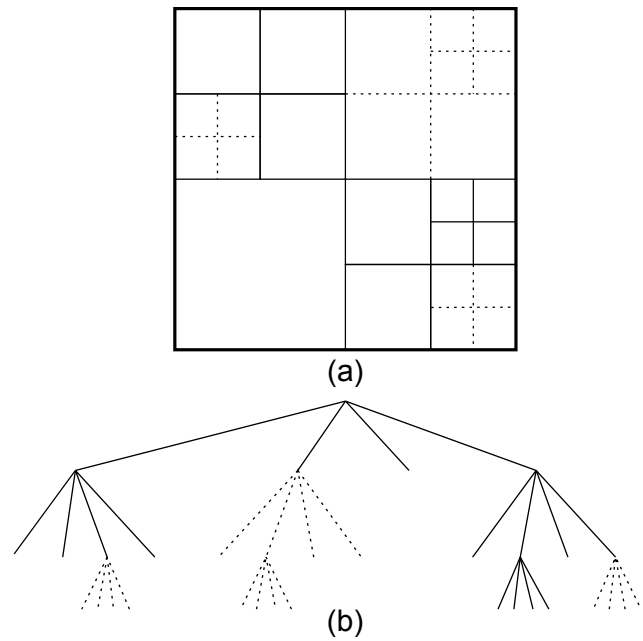

Fig. 1. **(a) Example of coding tree block (solid bold) into prediction blocks (solid) and transform blocks (dashed) of variable size. (b) Corresponding nested quadtree structure.**

However, the concept of quadtree-based image and video coding is intrinsically tied to certain drawbacks, as analyzed in [5]. For instance, the systematic decomposition of every block into four child blocks does not allow to jointly represent child blocks that belong to two different parent blocks. Also, if a given block is divided into four child blocks, all child blocks are typically coded separately, even if two or three of them may share the same coding parameters.

Thus, the sub-optimal rate-distortion properties of an initial quadtree-based subdivision can be substantially improved [5]. Block merging has been introduced in order to compensate the aforementioned drawbacks by enabling the merging of spatially neighboring blocks with identical coding parameters. As a consequence, block merging reduces the amount of redundant sets of transmitted coding parameters that are typically generated when using a quadtree-based block partitioning algorithm. In our proposed block merging algorithm, as further described below, specific syntax elements are introduced and coded together with the temporal prediction parameters for each leaf node of the prediction quadtree.

Merging is particularly well-suited for regions of homogeneous motion that can be sufficiently described by using a single motion parameter set. For example, a video signal often contains large spatially adjacent regions, each having homogeneous motion parameters that differ significantly among the regions. Inside each such region, the corresponding blocks can be merged, even if the maximum allowed block size for quadtree-based partitioning is reached.

Merging enables the creation of regions with arbitrary

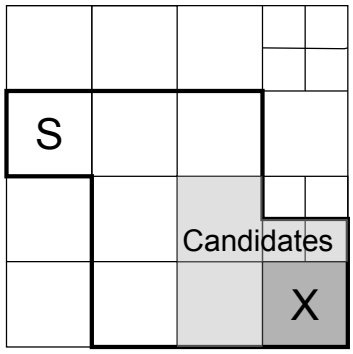

Fig. 2. **A quadtree segmentation with multiple blocks merged into one region (bold line). Block S contains the prediction information for the region. Blocks in the gray shaded region are the candidates for block X to be merged with.**

shapes. One illustrating example is shown in Fig. 2; other examples as outcome of our simulations are shown in Figs. 5–6. Ideally, the boundary of each region should coincide with the motion discontinuities in the given video signal. With quadtree-based block partitioning combined with merging, the picture can be subdivided into smaller and smaller blocks thereby approximating the motion boundary and minimizing the size of partitions containing the boundary. Subsequently, each created quadtree leaf block can be merged into a region on either side of the boundary.

4. THE PROPOSED BLOCK MERGING ALGORITHM

In order to illustrate the operation of our proposed block merging algorithm, Fig. 2 shows an example of a quadtree segmentation. The purpose of block merging, as already explained, is to compensate for the disadvantages of an initial quadtree-based subdivision by reducing the redundant sets of transmitted coding parameters. This is achieved by creating regions composed of neighboring prediction blocks with identical motion information. Each region must contain at least one block that is not merged, seeding the new motion information. In Fig. 2, block S (for seed) corresponds to such a particular block. Block X in Fig. 2 corresponds to the current block to be en-/decoded. All the blocks en-/decoded before the current block along the block scanning pattern form the set of causal blocks. Note that within each CTB, the subblocks are traversed in depth-first order. In any case, the block to be en-/decoded can be merged only with one of its causal neighboring blocks. In Fig. 2, these blocks are highlighted by the gray shaded region.

A list of candidate sets of motion information is created using the motion information collected from these neighboring blocks. If the block X is merged with one of these candidates, it means that block X copies and uses the same motion information of one of its neighbors. In order to identify this motion information in the list, an index is signaled for the block X. The particular design of the block merging algorithm used to obtain simulation results for this paper only uses the motion parameter sets of the left and top neighboring blocks as candidates. Thus, the candidate list contains at most two entries.

The new syntax elements merge_flag and merge_index are introduced for signalization. The binary valued merge_flag is signaled for each motion-compensated partition block prior to any prediction parameter. The list of merge candidates may be empty because, e.g., no neighboring block is predicted by using motion compensation. In this case, block X cannot be merged and merge_flag is not transmitted. Otherwise, if merge_flag is equal to 0, the current block X is not merged with any of its neighboring candidate blocks and motion parameters for X are signaled explicitly. If the motion information of two different merge candidates are identical, the list of candidate sets of motion information can be reduced. In this way, the list of candidates of motion information is only composed of motion information which are different and thus, its size is as small as possible. If there is only one candidate in the list, no merge_index needs to be transmitted. Otherwise, the value of merge_index which is transmitted identifies the position of the candidate in the reduced list and its associated motion information which will be used for en-/decoding the current block X.

There is a conceptual similarity between the direct mode (of H.264/AVC) and our proposed block merging approach. Direct mode also infers the motion information for motion-compensated prediction (MCP) from previously decoded blocks. Thus, both algorithms are using motion information from neighboring blocks in order to reduce the bit rate required for transmitting motion parameters. However, for direct mode the motion information is derived independently for each reference picture. This means that for each reference picture (two in the HHI video codec), a motion predictor needs to be calculated from neighboring blocks for the current block to be en-/decoded. To derive a motion predictor in direct mode, up to four motion vectors of neighboring blocks are used, namely the above, left, above-left and above-right neighbors. The median of these vector components is used to calculate the motion predictor of the current block. For direct mode in H.264/AVC, also a flag is signaled for MCP partitions indicating whether the partition is coded in direct mode or not [1].

Despite this similarity, the two algorithms differ in the way they handle neighboring motion information. While the direct mode infers motion parameters from adjacent blocks, merging creates regions where all the blocks share the complete motion information. The creation of such regions is performed using only simple operations, such as comparing and copying the complete motion information from a neighbor. In comparison with this, the calculation of motion predictors in direct mode requires more computational complexity. Motion

Class	Sequence	BD rate CS1 [%]
B1 (1920x1080)	Kimono	-38.30
	ParkScene	-24.28
B2 (1920x1080)	Cactus	-29.22
	Basketball Drive	-35.97
	BQTerrace	-41.92
C (832x480)	Basketball Drill	-31.88
	BQMall	-29.71
	PartyScene	-28.09
	RaceHorses	-29.67
D (416x240)	Basketball Pass	-21.97
	BQSquare	-43.96
	BlowingBubbles	-23.60
	RaceHorses	-19.64
Average		**-30.63**

Table 1. **Average BD rate savings (percentage) of the HHI codec relative to the H.264/AVC HP anchor at CS1 test conditions.**

predictors are calculated for all reference pictures individually while using up to four neighboring blocks, where additional operations are required to calculate the mean/median values, in case there is more than one neighboring motion vector available.

5. SIMULATION SETTINGS AND RESULTS

The codec used for performing our experiments is the HHI video codec as presented in section 2. The experiments were conducted by encoding all test sequences, as defined in the CfP [8]. Two constraint sets (CS) restricting the temporal coding structure were defined in the CfP as follows:

- CS1: hierarchical B picture coding structure [9] with a structural temporal delay not larger than 8 pictures,
- CS2: hierarchical P picture coding structure without picture reordering in the decoder.

Furthermore, the coding tree block has a fixed size of 64×64 pixels (for luma) and a maximum quadtree depth of 4, resulting in a minimum block size of 4×4. In order to obey CS1, inter-prediction with up to two hypotheses is employed within groups of 8 successive pictures in a hierarchical structure using 4 temporal layers. For CS2, inter prediction is restricted to one hypothesis from temporally preceding reference pictures. The intra period in CS1 is set depending on the number of frames per second for each sequence, whereas in CS2, only one intra picture at the beginning of each sequence has been utilized. Five quantization parameter (QP) values are chosen to obtain five RD points per sequence. The test sequences are separated into 4 classes according to their resolution and their content. For instance, the sequences of class E show typical videoconferencing scenes with static background. That is also why class E is only tested for CS2, the use case for low-delay applications. The other classes are characterized by different

Class	Sequences	BD rate CS1 [%]	BD rate CS2 [%]
B1 (1920x1080)	Kimono	-2.42	-2.37
	ParkScene	-2.78	-2.88
B2 (1920x1080)	Cactus	-2.65	-2.28
	Basketball Drive	-2.93	-2.56
	BQTerrace	-3.45	-2.62
C (832x480)	Basketball Drill	-2.79	-2.58
	BQMall	-4.80	-4.47
	PartyScene	-2.22	-3.46
	RaceHorses	-3.08	-3.00
D (416x240)	Basketball Pass	-2.82	-2.11
	BQSquare	-2.41	-2.14
	BlowingBubbles	-2.10	-1.98
	RaceHorses	-2.81	-2.51
Average		**-2.87**	**-2.70**
E (1280x720)	Vidyo1	n/a	-4.50
	Vidyo3	n/a	-5.79
	Vidyo4	n/a	-4.32
Total Average		**-2.87**	**-3.10**

Table 2. **Averaged BD rate savings (in percentage) of the HHI codec with block merging relative to the same codec with direct mode (instead of block merging).**

resolutions and a variety of content. With the exception of class E, all classes are tested for both CS1 and CS2. All presented bit rate savings are measured in terms of Bjøntegaard Delta (BD) rate [10].

First, the HHI proposal for HEVC (which incorporates block merging) has been evaluated in comparison to H.264/AVC high profile (HP) as an anchor [8]. Table 1 shows the bit rate savings of the HHI proposal relative to this anchor. The average gain for CS1 is around 30%.

The efficiency of block merging has been evaluated by comparing two versions of the HHI codec. The first version is given by the original HHI proposal for HEVC (including block merging), while the other version incorporates direct mode instead of block merging, as described in Section 2. The results of this comparison at CS1 and CS2 test conditions with the direct mode version as a reference are listed in Table 2. It can be observed that the gains of block merging in terms of average BD rate savings are around 3%, and that, to a large extent, those gains are independent of resolution or sequence type. However, there are several sequences where coding efficiency lies significantly above the average such as, e.g., all class E sequences, as shown in Table 2. These particularly favorable results for class E are mostly due to the static background of these sequences which can be easily represented by using a single merged region with zero motion vector at the corresponding block merging seed.

In Fig. 3, the RD curves of the sequence BQMall for CS1 are presented. The curves show the RD performance of three competing schemes: the HHI codec with block merging (the HHI proposal for HEVC), the HHI codec with direct mode,

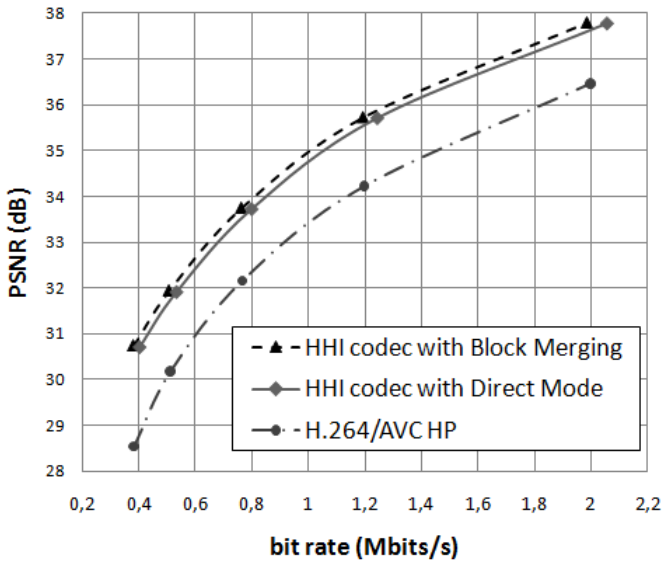

Fig. 3. **RD performance plot comparing direct mode and block merging in the HHI video codec as well as H.264/AVC HP by using the test sequence BQMall at CS1 test conditions.**

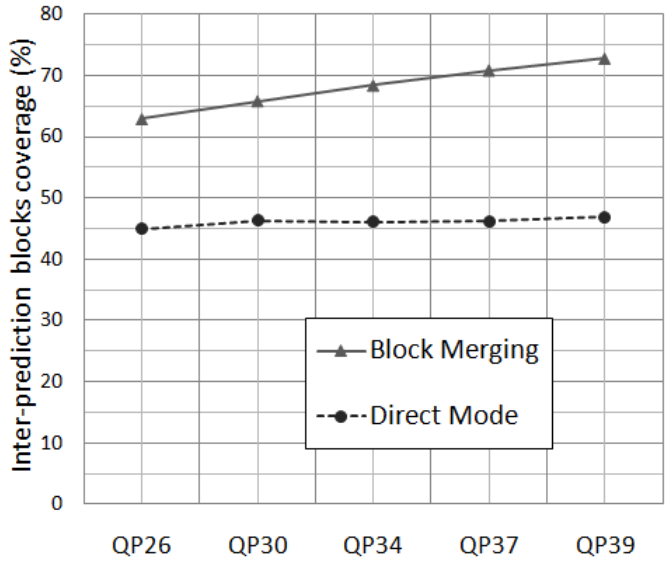

Fig. 4. **Percentage coverage of MCP blocks using the direct mode and MCP blocks using block merging, both depending on the QP by using the sequence BQMall at CS1 test conditions.**

and the H.264/AVC HP anchor scheme. For BQMall, the average BD rate saving gain of the HHI proposal relative to H.264/AVC HP is 29.7%. The average gain relative to the HHI codec with direct mode is 4.8% BD rate savings, where the individual gains for the different rate points are distributed more or less uniformly, which has also been observed for all other sequences.

Fig. 4 compares the percentage of picture samples covered by MCP blocks using direct mode with that using block merging. The curves were obtained by encoding the BQMall sequence at different QP values using both versions of the HHI codec. As can be seen from Fig. 4, the percentage of coverage for direct mode in comparison with that for block merging is much smaller. This behavior was consistently observed for the entire test set of sequences, which is an indication for block merging being selected more often due to its more favorable RD cost. Also, it can be observed that with larger QP values, the coverage increases for either method, although more distinctively for the case of block merging.

6. ILLUSTRATION OF THE RESULTS OBTAINED WITH BLOCK MERGING

In the following, details of encoded frames are further analyzed to demonstrate the motion modeling capabilities of block merging. In Figs. 5 and 6, white lines separate merged regions and grey lines indicate MCP leaf blocks of CTBs.

Fig. 5 shows a detail from a picture belonging to the BQMall sequence. In this part of the sequence, the background is static and only people in the foreground are moving. As can be seen from Fig. 5, the boundaries of the merged regions are clearly separating the moving persons from the static background.

Fig. 6 shows that block merging is effective also for more complex sequences. The corresponding Parkscene sequence was shot with a panning camera in front of a scene consisting of a park in the background and some tree branches in the foreground. The figure reveals distinct 'object planes' in the picture, all of which are moving differently such that each plane can be identified by a different set of block merging regions in Fig. 6. The tree branches correspond to the foreground, the colonnade belongs to a second plane, and the rest of the park can be attributed to the background.

As in the two examples examined above, natural video signals very often contain distinct moving objects, each with a different kind of motion. At the boundary of such objects, the field of motion parameters can be expected to exhibit very abrupt changes. These motion discontinuities can be modeled very efficiently by block merging, as it selects exactly one of the neighboring blocks to infer the motion information for the current block, leaving other candidates aside. In contrast to that, direct mode uses up to four spatially neighboring blocks to calculate an intermediate motion representation which, in the presence of distinct moving objects, may not be adequately representing the real motion in a given scene.

7. CONCLUSION AND FUTURE WORK

In this paper, block merging together with a quadtree-based block partitioning concept has been presented as an alternative to the conceptually similar direct mode for motion-compensated prediction. Simulation results show that the

Fig. 5. **Detail of sequence BQMall in Class C with a static background and moving persons in the foreground.**

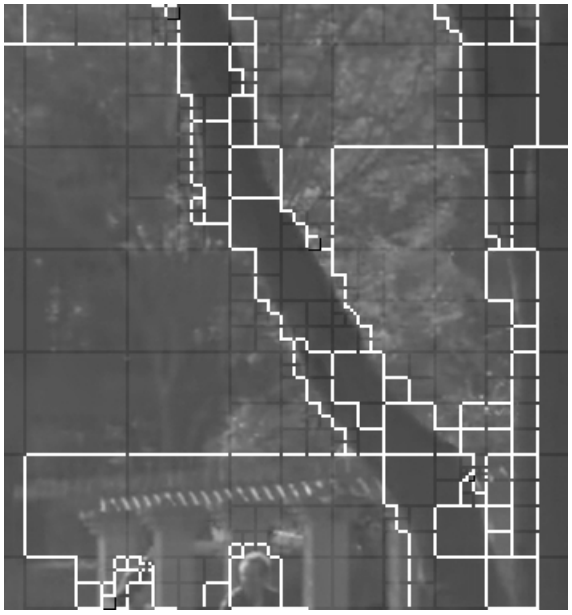

Fig. 6. **Detail of sequence Parkscene in Class B with tree branches in the foreground and colonnades at the lower picture boundary, captured by a panning camera.**

proposed merging algorithm better exploits spatiotemporal correlations in motion model parameters. Averaged BD rate savings of around 3% for a whole test set of video sequences have been achieved in comparison to the direct mode. Block merging not only yields better RD coding performance than the traditional direct mode, but also offers a computationally less demanding algorithm. These benefits were crucial for the proposed block merging algorithm to be adopted as part of the working draft of HEVC [11] which has been currently issued by the Joint Collaborative Team on Video Coding (JCT-VC). Further aspects of block merging within the HEVC test model are currently under investigation and further study [12].

8. REFERENCES

[1] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 560–576, Jul. 2003.

[2] A. M. Tourapis, F. Wu, and S. Li, "Direct Mode Coding for Bipredicitve Slices in the H.264 Standard, *IEEE Trans. Ciruits and Syst. Video Technol.*, vol. 15, no. 1, pp. 119–126, Jan. 2005.

[3] 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, Apr. 2010.

[4] D. Marpe, H. Schwarz, S. Boße, B. Bross, P. Helle, T. Hinz, H. Kirchhoffer, H. Lakshman, T. Nguyen, S. Oudin, M. Siekmann, K. Sühring, M. Winken, and T. Wiegand, "Video Compression Using Quadtrees, Leaf Merging and Novel Techniques for Motion Representation and Entropy Coding", *IEEE Trans. Ciruits and Syst. Video Technol.*, vol. 20, no. 12, Dec. 2010.

[5] R. Shukla, P. L. Dragotti, M. N. Do and M. Vetterli, "Rate-Distortion Optimized Tree-Structured Compression Algorithms for Piecewise Polynomial Images", *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 343–359, Mar. 2005.

[6] R. De Forni and D. S. Taubman, "On the benefits of leaf merging in quadtree motion models", *Proc. IEEE Int. Conf. Image Process.*, vol. 2., pp. 858–861, Sep. 2005.

[7] R. Mathew, D. S. Taubman, "Quad-Tree Motion Modeling with Leaf Merging", *IEEE Trans. Ciruits and Syst. Video Technol.*, vol. 20, no. 10, pp. 1331–1345, Oct. 2010.

[8] 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.

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

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

[11] T. Wiegand, W,-J. Han, B. Bross, J. R. Ohm, G. J. Sullivan, "WD2: Working Draft 2 of High-Efficiency Video Coding" JCT-VC Meeting, Daegu, KR, Doc. *JCTVC-D503*, Output Document, Jan. 2011.

[12] J. Jung, B. Bross, P. Chen, W,-J. Han, "Summary report of Core Experiment 9: MV Coding and Skip/Merge operations" JCT-VC Meeting, Geneva, Switzerland, Doc. *JCTVC-E029*, Input Document, Jan. 2011.