Context-Adaptive Binary Arithmetic Coding with Precise Probability Estimation and Complexity Scalability for High-Efficiency Video Coding*

Damian Karwowski [a], **Marek Domański** [a]
[a] Poznan University of Technology, 60-965 Poznań, Poland

Abstract. An improved Context-based Adaptive Binary Arithmetic Coding (CABAC) is presented in the paper. The idea for the improvement is to use a more accurate mechanism for estimation of symbol probabilities in the standard CABAC algorithm. The authors' proposal of such a mechanism is based on the Context-Tree Weighting (CTW) technique. In the framework of HEVC video encoder the improved CABAC allows 1.2%-4.5% bitrate saving compared to the original CABAC algorithm. The application of the proposed algorithm marginally affects the complexity of HEVC video encoder but the complexity of video decoder increases by 32%-38%. In order to decrease the complexity of video decoding, a new tool has been proposed for the improved CABAC that enables scaling of the decoder complexity. Experiments show that this tool gives 5%-7.5% reduction of the decoding time while still maintaining high efficiency of data compression.

Keywords: video compression, AVC, HEVC, entropy coding, CABAC, improved CABAC.

Address all correspondence to: Damian Karwowski, Poznan University of Technology, Faculty of Electronics and Telecommunications, Chair of Multimedia Telecommunications and Microelectronics, Polanka 3, Poznań, Poland, 60-965; Tel: +48 665 38 44; E-mail: dkarwow@multimedia.edu.pl

1. Introduction

More than half of the whole communication traffic is related to video, and the role of video communication is growing rapidly. Efficient representation of digital video plays an increasing

*
Version of the paper submitted, and after editorial works accepted for publication in SPIE Journal of Electronic Imaging, 2016.

Citation:
Damian Karwowski and Marek Domański, "Context-adaptive binary arithmetic coding with precise probability estimation and complexity scalability for high-efficiency video coding", J. Electron. Imaging. 25(1), 013010 (Jan 20, 2016).

DOI abstract link:
http://dx.doi.org/10.1117/1.JEI.25.1.013010

role in modern multimedia communications. Contemporary video encoding is performed in two phases, in which respective video bitstreams are produced[1-8]. In the more complex first phase, the decisions concerning syntax elements and their values are taken. Then, in the second phase, actual bitstreams are produced that constitute binary representations of the syntax elements. The second phase of video encoding is related to entropy coding of syntax elements. In this paper, only this phase of video compression is considered.

In older video compression technologies[1], entropy coding was performed using relatively simple techniques of Huffman coding[9]. These techniques were non-adaptive, with no (or very limited) ability for adaptation to the changing statistics of the video signal and with moderate compression performance. It was a motivation to work out even more efficient entropy coding techniques with more complex data statistics modelling. Two classes of techniques were developed in this context: adaptive variable-length coding (adaptive VLC)[10] and adaptive arithmetic coding[11,12]. Both classes of techniques were successfully applied in video encoders of the more recent generations (H.263, MPEG-4 AVC/H.264, HEVC)[2, 5, 6, 7, 8, 13].

Intensive research on both classes of methods that was performed in the framework of hybrid video coding technology revealed the superior efficiency of arithmetic coding based techniques in comparison to VLC-based solutions[14]. The state-of-the-art entropy coding technique used in video encoders is Context-based Adaptive Binary Arithmetic Coding (CABAC)[14,15,16]. As compared to other entropy encoders used in video compression, it uses binary arithmetic coding together with sophisticated mechanisms of data statistics modelling. Originally, the CABAC algorithm found application in the Advanced Video Coding (AVC) standard (ISO/IEC MPEG-4 AVC and ITU-T Rec. H.264)[5]. A slightly modified version of the CABAC technique is also used in the newest High Efficiency Video Coding (HEVC) technology[6,8,15,16].

Extensive research on the CABAC algorithm showed its very good compression performance (as compared to adaptive VLC-based solutions) in the framework of both the AVC and HEVC technologies[14,17]. The goal of the present paper is to explore the possibilities to increase further the efficiency of the CABAC algorithm. Some improved algorithms have been already developed in recent years. Most of the improvements were aimed at the version of CABAC algorithm used in the framework of the AVC encoder. In these works, efforts were put to further improve the techniques of data statistics estimation in CABAC. In particular, attempts at improving the CABAC were focused on: the application of a more complex context pattern in CABAC[18-21], using more sophisticated schemes of coding transformed residual data and motion data[22,23], and applying even more accurate techniques of conditional probability estimation of input data[21,24-29]. The proposed improvements were described in the literature in detail, and led to a 0.1%-10% reduction of the CABAC bitstream within the AVC encoder. Much less work has been carried out in the context of version of CABAC used in the HEVC technology. Most of the known solutions concern the simplifications of the entropy codec and they do not increase the efficiency of data compression[e.g. 30-34]. The exceptions here are two similar proposals for CABAC improvement that increase the accuracy of symbols probability estimation in the codec. These are the two-parameter probability update model[35,36] and the counter- based probability model update for CABAC[37]. The two proposals are characterized by similar coding efficiency and allow the reduction of bitrate by 0.5% - 0.8%.

The authors' previous research works carried out in this area were mainly focused on improving the CABAC of AVC through the use of a larger context pattern and a more precise mechanism of probability estimation using the Context-Tree Weighting (CTW) method[24-26]. The proposed improved version of the CABAC algorithm was thoroughly tested. The obtained results

showed the possibility of reducing the bitrate by 1.5%-8% if the proposed solutions were applied in the AVC encoder.

However, the new HEVC video encoder is significantly different from the AVC developed about 10 years earlier[6,8]. New compression tools in the HEVC, as well as high number of the coding modes resulted in significant changes in the set of syntax elements that are further encoded using the CABAC entropy encoder. Both the statistics of the encoded data and the efficiency of video encoding are different for the AVC and the HEVC versions of the CABAC.[38,39]. For these reasons, the improvements achieved for the CABAC of AVC and the conclusions drawn from the research cannot be easily adopted to the new version of the CABAC applied in the HEVC. Therefore, an important question arises regarding the possibilities of further improving the efficiency of the CABAC currently used as a part of the state-of-the-art HEVC video compression technology[6]. This paper addresses this abovementioned open research problem.

Currently the research has already started toward development of a future video coding technology that is expected to be finished until 2021-2022[40]. The usual methodology for such research is to use the existing video coding technology, i.e. the HEVC, both as a starting point for the developments and as the reference. The main objective of our work is to develop an improved version of the CABAC technique that may be useful in the development of the next generation of video codecs that will follow the contemporary HEVC technology.

The paper focuses on the possibilities of improving the CABAC of the HEVC by using an even more precise scheme of conditional probability estimation. An additional goal is to explore the relationship between compression performance improvement and the increase of complexity of the HEVC video codec. This question has already been partially answered by one of the authors in[41], where a preliminary version of the improved CABAC (also based on the CTW technique)

was briefly presented along with very limited experimental data. Nevertheless, an extended version of the improved CABAC is presented in this paper, together with a large set of new experimental results. The drawback of the proposal from[41] was high complexity of the context modeller. In this paper, an approach is proposed to achieve complexity scalability. Therefore, in this paper, the authors propose a more practical approach that results in compression efficiency improvement at the cost of limited complexity increase. In order to study thoroughly the properties of the proposed approach, we report the results for the CABAC with different numbers of statistical models. Therefore, the paper reports the experimental results obtained in the context of two versions of the CABAC technique:

1) the older CABAC version used during development and standardization of the HEVC technology – in our experiments we use the older CABAC version that is implemented in Version 3.2 (HM3.2) of HEVC reference software[51],

2) the newer CABAC version that is adopted into the ISO MPEG-H and ITU H.265 standards – in our experiments we use the implementation from Version 16.6 (HM16.6) of HEVC reference software[52].

The two versions of the CABAC use different numbers of statistical models as it will be explained later in this paper.

2. CABAC Overview and Idea for the Algorithm Improvement

CABAC performs binary arithmetic coding of syntax elements. The application of a binary arithmetic encoder core (the so-called M-encoder) creates the requirement to map all m-ary (non-binary) valued syntax elements into a string of binary symbols (a binary symbol is called a bin and may be of value 0 or 1). This is realized in CABAC at the first stage of coding (see Fig. 1.).

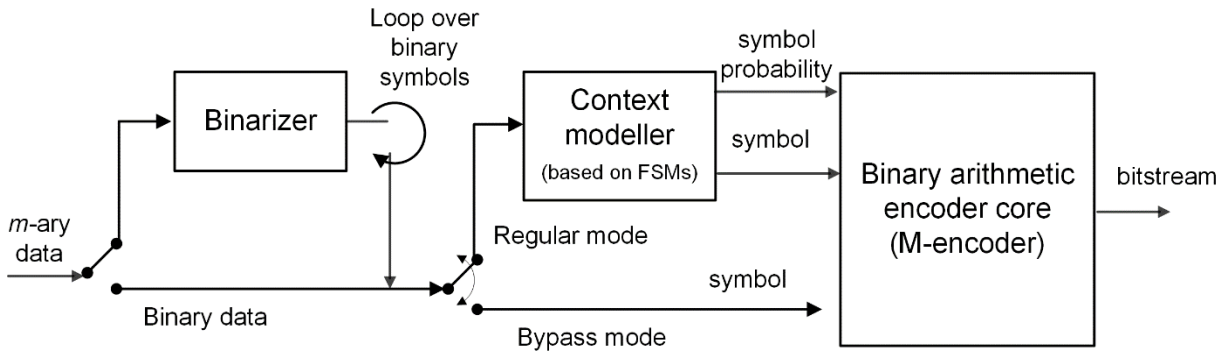


Fig. 1. General block diagram of CABAC encoder. The idea of the drawing is taken from[14].

Due to the use of many different binarization schemes in CABAC, the binarization behaves as adaptive VLC coding, so it itself reduces the size of the data stream[14]. Nevertheless, inter-symbol (inter-bin) redundancy that exists in the resulted string of binary symbols (that is, after the binarization) is additionally extra reduced in CABAC in the processes of data statistics modelling and binary arithmetic coding (see Fig. 1.). From the theory of arithmetic coding of data it is well known that the way of calculating the statistics of the coded data (that is data statistics modelling) is crucial from the point of view of the compression performance. Therefore, the methods that were applied in this part of the CABAC algorithm belong to the most advanced and powerful among the solutions practically used in video compression.

First of all, in order to track the statistics of individual syntax elements in an accurate and independent manner, a stream of binary symbols (bins) is split into a large number of individual binary sub-streams for which data statistics estimation is performed independently (it is related to the so-called regular coding mode and regular bins in CABAC). There is also one separate sub-stream whose symbols are coded in a simplified way, assuming a uniform probability distribution of binary symbols (it refers to the so- called bypass coding mode and bypass bins in CABAC). Both the way of assigning a symbol to an appropriate sub-stream and an exact number of regular

sub-streams are different for the versions of CABAC used in AVC and HEVC encoders (e.g. 460 and 173 to 225† regular sub-streams are used in AVC and HEVC, respectively).

Secondly, the conditional probability of a symbol that is coded in the regular coding mode is calculated assuming the "exponential aging" model of data[42]. Additionally, in order to limit the complexity of the arithmetic coding, a limited set of 128 quantized values of probabilities ranging in the interval [0.01875; 0.98125] is used within this model. The last two things allow for a simplified estimation of probabilities of binary symbols using pre-defined Finite State Machines (FSMs). The definition of each of FSM (i.e. transition rules between states) is exactly the same and the number of used FSMs corresponds to the number of regular sub-streams – each FSM tracks the statistics of symbols that come from an individual binary sub-stream.

The simplifications of the CABAC algorithm presented in the last paragraph reduce the computational as well as memory cost of CABAC, but of course it is obtained at the cost of compression performance reduction of the entropy encoding.

The main goal of the paper is to explore the possibilities of improving the compression performance of CABAC by applying even more accurate techniques of conditional probability estimation. A complementary goal is to test the relationship between compression performance improvement and the increase of complexity of the entropy and video codecs. As stated in the introduction of the paper, such problems were already investigated by the authors in the past[24-26] in the context of the AVC technology. These works investigated the improvements of CABAC achieved through the use of universal, more precise probability estimation techniques known from the literature. In particular, the following techniques were then investigated: the Context-Tree Weighting (CTW) method[43-45], the Prediction with Partial Matching (PPM) method[46,47], and one

† HM 3.2 version of the HEVC reference software uses 225 regular sub-streams, while in new HM 16.6 version of software there are only 173 of such sub-streams.

of the authors' method of joint application of both the CTW and PPM techniques[24]. Individual versions of the improved CABAC were tested in depth – and detailed results were presented in many publications[24-26]. The main conclusion drawn from the experiments was that the compression performance of CABAC can be reasonably increased when applying the CTW technique in the context modeller block of entropy codec.

This paper is a continuation of the previous works and focuses on the improvements of the CABAC algorithm for its application in future video compression technologies. Taking advantage of the previously obtained experience, improvements are made in performing more accurate data statistics modelling of HEVC syntax elements using the CTW method together with the newly proposed mechanism providing scalability of the entropy codec complexity.

3. Improved CABAC Algorithm

3.1. CTW-CABAC: Introduction

The improvement of CABAC proposed in this paper is to replace the original method of determining the probabilities of symbols (i.e. the one using FSMs) with a new method that uses CTW. The basis of this technique is to use a binary context tree of depth D. In this tree the information on the number of 0 and 1 symbols that occurred in a situation where a given string of symbols had appeared earlier is stored (the string of symbols that had been encoded previously makes the context information). Based on the conditional statistics gathered on the tree the final conditional probability is calculated in the root λ of the tree (see Fig. 3). This probability is then used in the core of arithmetic encoder. As a matter of fact, the idea of application of the CTW in video compression has been proposed earlier by other authors[28,29]. However, in contrast to these proposals, the authors have developed a novel, far more sophisticated method of incorporating the

CTW technique into the CABAC algorithm[24-26], that exploits significantly higher number of binary context trees. This paper presents another, extended version of the improved CABAC (that is based on CTW). The proposed entropy codec is called CTW-CABAC in this paper. The block diagram of the CTW-CABAC is presented in Fig. 2.

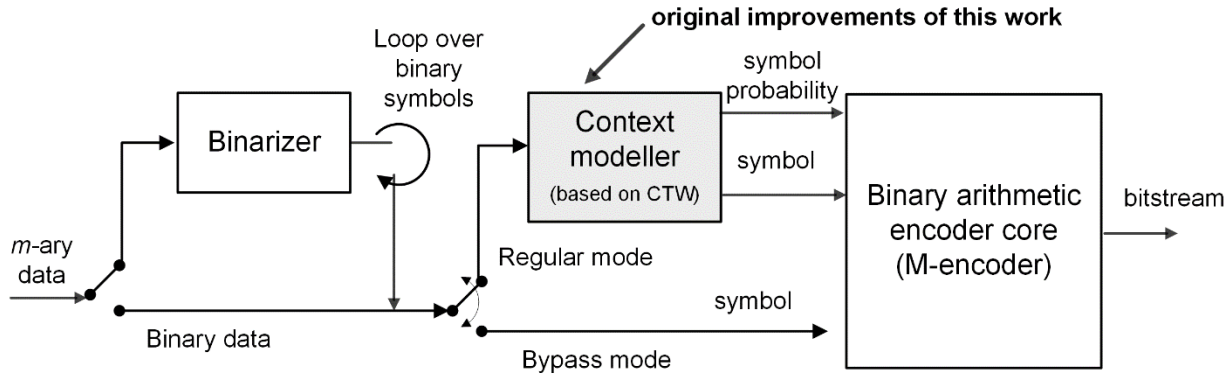


Fig. 2. General block diagram of the proposed CTW-CABAC encoder.

In the CTW-CABAC, only the context modeller block was subjected to modifications – other parts of CABAC like binarization schemes, definition of sub-streams and mechanism of assigning binary symbols to sub-streams, and the arithmetic codec core were left unchanged with respect to the original algorithm. Individual functional blocks of the CTW-CABAC will be presented in more detail in the next subsections.

3.2. CTW-CABAC: Data Statistics Modeling

In the context modeller block of the original CABAC, a simplified mechanism of symbols probabilities estimation (based on the idea of finite-state machines – FSMs) is replaced with a more accurate one using the CTW method. In order to do that, each of the FSMs is substituted with a dedicated binary context tree of depth D as shown in Fig. 3 (more details about binary context trees can be found in[43,44]).

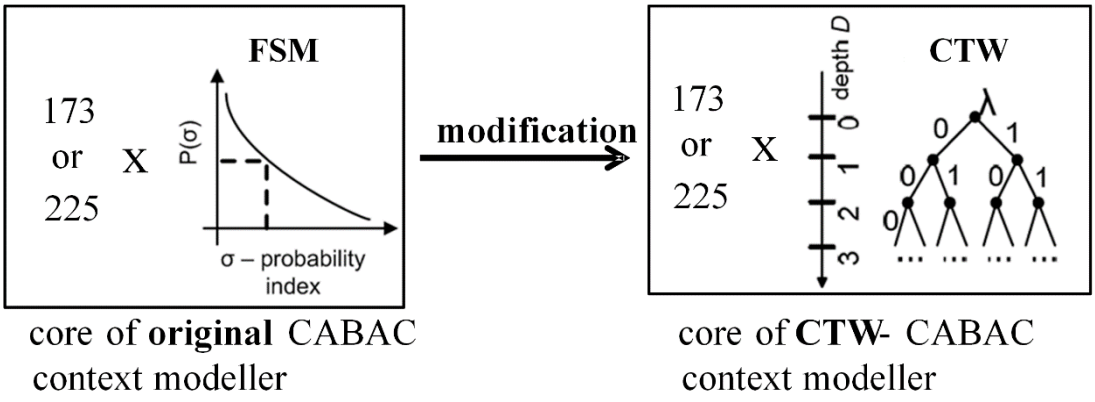


Fig. 3. Binary context trees in the context modeller of the CTW-CABAC.

This way, the total number of context trees corresponds to the number of regular binary sub-streams defined in the original CABAC (there are 225 sub-streams in the HM 3.2 version of the HEVC reference software[51] and 173 sub-streams in the newer HM 16.6 version[52] of this software that corresponds to the bitstream syntax and semantics described in the HEVC standard). An individual context tree is used to gather the statistics of symbols from a sub-stream that appear in different contexts (i.e. the previously coded symbols). The context information and context length affect the efficiency of the statistics estimation block. These aspects have been already thoroughly tested by the authors in the context of the AVC technology[24-26]. Following the main conclusions of those experiments, context trees of depth D=8 is used in the CTW-CABAC as a good compromise between coding efficiency and complexity of the entropy codec. Additionally, in the our proposal, the context contains symbols that were previously coded within sub-streams derived from a given syntax element, instead of only symbols derived from one specific sub-stream. Such an approach reduces the statistical redundancy that exists between symbols that belong to different sub-streams, but appear in the same syntax element.

The method of calculating conditional probabilities in the CTW technique to a large extent determines both the complexity and the memory demand of the algorithm. In order to speed up the

computations, the optimized scheme of probability estimation (known in the literature) is used[45,48]. The main idea of the optimized algorithm is to carry out calculations in the logarithmic domain in order to avoid time-consuming multiplication and division operations when calculating the probabilities. This approach significantly reduces the complexity of the CTW-CABAC. The amount of information that must be saved in a single node of the context tree considerably influences the storage complexity of CTW. The optimized algorithm reduces the memory requirements to a large extent – only 4 bytes of data are kept in a node of the tree in the optimized implementation, in contrast to at least 66 bytes for a standard implementation.

3.3. CTW-CABAC: Context Trees Initialization

The manner of context tree initialization strongly affects the efficiency of the CTW technique. Therefore, the initialization method that is used is based on the initialization mechanism of FSMs in the original CABAC algorithm (within the HEVC video codec). For a given context tree, the values of counters of 0 and 1 symbols are appropriately set in the node λ of the tree in order to obtain the same probability of a binary symbol as in the initialization procedure of a given FSM in the original algorithm. For the sake of simplicity, the counters of other nodes of the context tree are set to zero. In the CTW-CABAC algorithm, tree initialization is performed each time at the beginning of I slice and a slice of a new type.

3.4. CTW-CABAC: Binarization of Data

Binarization of data was not the subject of improvement in this work. However, it is known that, as in the case of data statistics modelling, the manner of the data binarization significantly affects the performance and complexity of the entropy encoder. In order to unambiguously determine the efficiency of the proposed CTW-CABAC algorithm, it has been explored for two, known from the

literature, methods of data binarization. The two methods differ in the binarization schemes for transform coefficient data only, which constitute the greater part of the bitstream. For this type of data, the first method uses a binarization scheme which is a concatenation of the truncated unary code, the truncated Golomb- Rice code and the 0-th order Exp-Golomb (EG0) code[49]. The second method uses binarization based on the concatenation of the truncated unary code and the 0-th order Exp-Golomb (EG0) code, as it was defined in the AVC technology[14]. In the case of other data types, the same binarization schemes (as defined in HEVC) are used in the two methods of binarization. In this way two versions of the CTW-CABAC algorithm have been tested: CTW- CABAC Version 1 and CTW-CABAC Version 2 with the first and the second method of data binarization, respectively. In a binarized word for the absolute value of the transform coefficient level (abs level), the number of binary symbols (bins) coded with individual binarization schemes is presented in Fig. 4 for both versions of the CTW-CABAC.

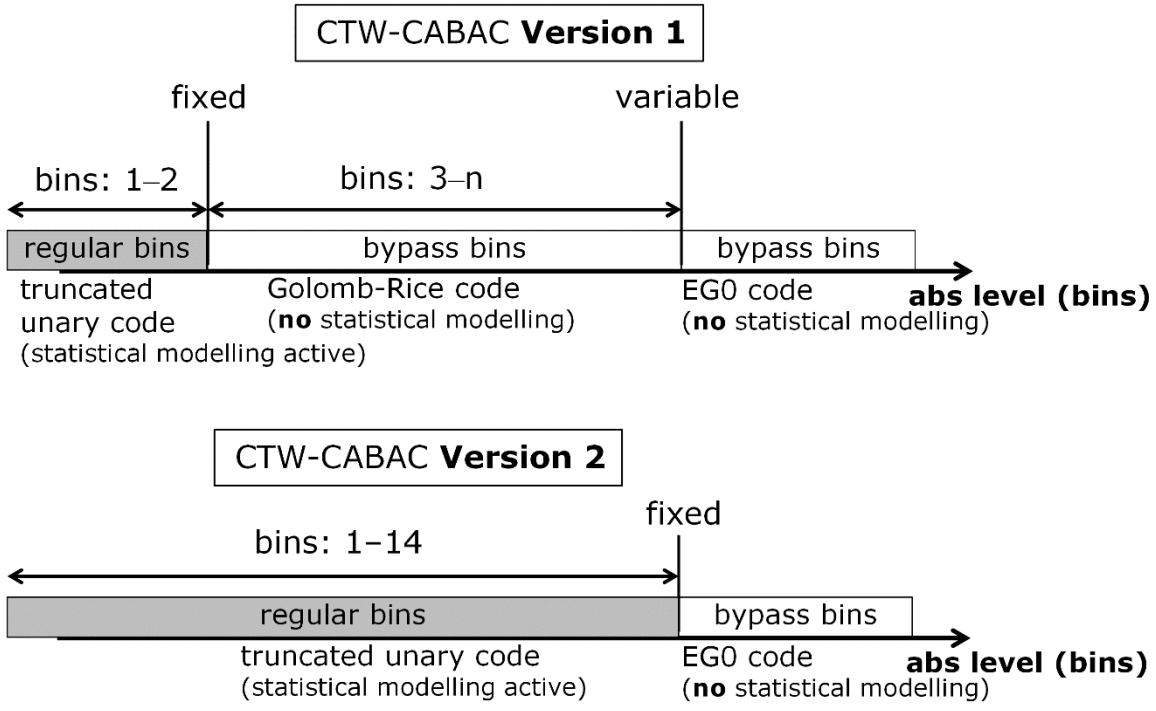


Fig. 4. Transform coefficient data – binarization schemes for the CTW-CABAC Version 1 and CTW-CABAC Version 2.

What is important, a smaller number of binary symbols (bins) is coded with a regular coding mode in the case of the CTW-CABAC Version 1 algorithm (as presented in Fig. 4). This way, two versions of CTW- CABAC considered in the paper also differ in the mechanism of statistical modelling of transform data, with more complex statistical modelling of transform data for CTW-CABAC Version 2 (due to higher number of regular bins relative to CTW- CABAC Version 1).

3.5. CTW-CABAC: Binary Arithmetic Coding

It was experimentally demonstrated that the M-encoder core is characterized by an excellent coding efficiency and complexity tradeoff[50]. For that reason, this core is also used in the considered versions of the CTW- CABAC. Nevertheless, the core was originally adapted to properly work with only a limited set of 128 quantized values of probabilities. The improved context modeller block (based on CTW) produces probabilities of symbols from a significantly larger set of values as compared to the small set defined in the original CABAC algorithm.

Due to the reason outlined above, probability values of CTW-CABAC must be mapped into one of the values from the limited set of 128 probability values. The mapping process is executed on the basis of the criterion of minimization of absolute difference between the probabilities calculated with CTW-CABAC and the original CABAC algorithm.

3.6. CTW-CABAC: Tool for Entropy Codec Complexity Scalability

The previous results achieved by the authors on the topic revealed a high complexity of the solutions based on the CTW technique[24,26]. Preliminary results achieved in the context of the HEVC technology proved that the use of the CTW-CABAC increases the complexity of the HEVC video decoder by a factor of 1.34 and 1.1 for 3 Mbps and 0.5 Mbps scenarios, respectively[41]. An essential problem to be solved is to reduce the complexity of the CTW-CABAC codec.

The problem was partially solved by applying the authors' mechanism of entropy codec complexity scalability. The main idea was to perform an accurate (and computationally intensive) calculation of the conditional probability of a symbol for every N-th data symbol using the CTW technique. In the case of intermediate symbols, the computationally complex procedure of probability estimation is omitted – by using the conditional probability that has earlier been calculated for a symbol that appeared in the same context. In order to do that, the context modeller must store information on conditional probabilities of symbols calculated with CTW for individual contexts (denoted as ctx) of depth (length) D, as presented in Fig. 5. The size of the table is equal to the number of context tree leaves. For example, there are 8 contexts (ctx 0 – ctx 7) in the case of a context tree of depth D=3, as presented in Fig. 5.

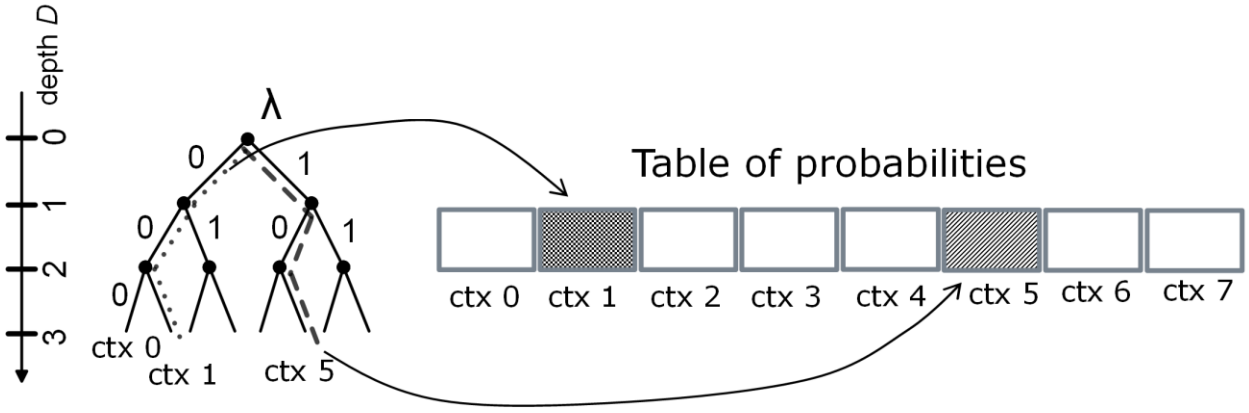


Fig. 5. Dedicated table of probabilities calculated for symbols in individual contexts.

Each time the probability of a symbol is calculated using CTW, an appropriate element of the table of probabilities is updated. What is important, the statistics of data in the context path nodes are updated each time when the data symbols are processed, regardless of the method of the probability estimation.

4. Efficiency and Complexity of the CTW-CABAC

The proposed sophisticated mechanism of data statistics estimation in CABAC influences both the compression performance and the complexity of the entropy codec. In order to explore how it affects the parameters of the entropy codec, a series of experiments were performed. The only way to assess the features of the new entropy codec is by performing experiments with a set of test video sequences. The next subsection presents the details of the experiment methodology.

4.1 Methodology of Experiments

In order to test the compression performance and the complexity of the CTW-CABAC entropy codec, the codec was implemented in the C++ programming language and activated in the framework of the HEVC video codec. In order to obtain reliable experimental results, both the video encoder and the video decoder were implemented. This way, the improved HEVC video codec (hereinafter referred to as CTW-HEVC) was built. The starting point for the implementation was the reference software of the HEVC video codec . In our experiments, the CTW-CABAC has been activated in two different versions of the HEVC reference software: 1) the older Version 3.2 (HM3.2) of software[51] and 2) the newer Version 16.6 (HM16.6) of HEVC software[52]. Both reference software versions share the same set of basic coding tools and, from the entropy coding point of view, the main difference between them is related to the number of regular sub-streams used in the CABAC algorithm (225 regular sub-streams in the HM3.2 and 173 regular sub-streams in the HM16.6). Therefore, the implementation of the improved algorithm in both versions of HEVC software (older and newer) provides reliable efficiency assessment of the proposed algorithm when working with different numbers of regular sub-streams.

The parameters of the CTW-CABAC codec (compression performance and complexity) were explored and referenced to the performance of the original CABAC algorithm within the

HEVC codec. Both versions of the CTW-CABAC (as described in Sec. 3.4) were tested. Compression performance tests were run for different values of the N parameter (N=1, N=2, and N=3). The parameter determines the level of complexity of the entropy codec (an accurate and computationally intensive calculation of a symbol probability with CTW is performed for every N-th symbol only). For both the HM3.2 and HM16.6 versions of software experiments were performed according to the following scenario:

- HD and full HD test video sequences were used: *Station* (1920x1080, 25Hz), *RiverBed* (1920x1080, 25Hz), *PoznańStreet* (1920x1088, 25Hz), *Balloons* (1024x768, 30Hz), *ChinaSpeed* (1024x768, 30Hz), *SlideEditing* (1280x720, 30Hz), *Mobcal* (1280x720, 50Hz), *ParkRun* (1280x720, 50Hz), *Shields* (1280x720, 50Hz), *Stockholm* (1280x720, 60Hz). The sequences are known and used by the video coding society in the works on video compression.
- 300 frames of each sequence were encoded (the exceptions are *RiverBed* and *PoznańStreet* sequences with 251 frames in each sequence).
- Experiments were performed for a wide range of bitrates, setting in the encoders different values of the quantization parameter (QP) equal to 22, 27, 32, 37. The value of the QP parameter determines the quality of reconstructed pictures – from excellent quality (QP=22) to poor quality (QP=37) of the video.
- IBBB… group of pictures (GOP) with hierarchical B pictures was used.
- Motion estimation was based on the Enhanced Predictive Zonal Search (EPZS) method, with the search range of 64. Two reference frames for B-slices were used for both L0 and L1 reference pictures lists.
- Coding Unit (CU) size of 64 with 4 splitting levels of CU was used.

- Rate-Distortion (RD) control mechanism and Rate-Distortion Optimized Quantization (RDOQ) were switched on. RD optimization mechanism was based on computationally simpler adaptive VLC codes (instead of full CABAC encoding) due to the common use of this approach in fast realizations of video encoders.
- Loop filters (deblocking filter, sample adaptive offset filter, and adaptive loop filter (only in case of HM3.2 since there is no such a filter in HM16.6)) were enabled.

Compression performance of CTW-CABAC was evaluated and referenced to the results of the original CABAC algorithm using the Bjontegard metric[53], popular in video compression. Having attained the compression results for 4 Rate-Distortion^{-1} (R-D^{-1}) points (for QP=22, 27, 32, 37) for both the CTW-CABAC and the original CABAC, two R-D^{-1} curves were created and compared in order to calculate the percentage saving in bitrate (BD Rate) between the two curves. In this paper, all Bjontegaard results are referred to the luma component.

The CTW-CABAC was also tested in terms of complexity. Experiments were performed for both versions of the proposed algorithm. Complexity of both the video encoder and the video decoder was considered. Complexity of the codecs (encoders and decoders) was determined as the time required by the processor to encode or to decode a video sequence. In particular, encoding and decoding times of HEVC with CTW-CABAC were referenced to results achieved for HEVC with original CABAC. As a result, a relative increase (measured in percentage) in complexity of video encoding and video decoding was calculated using the following mathematical formula:

$$\text{complexity increase [\%]} = \left(\frac{\text{complexity}_{\mathbf{CTW-HEVC}} - \text{complexity}_{\mathbf{original\ HEVC}}}{\text{complexity}_{\mathbf{original\ HEVC}}} \right) \cdot 100\%$$

Experiments were done for both the CTW-HEVC (Version 1 and Version 2) and for three different levels of entropy codec complexity determined by N parameter: N=1, N=2, and N=3.

It is obvious that the test platform influences the complexity results. Complexity tests were performed on an Intel Core i7-950 platform (4 cores, 8 threads, 3.07 GHz, 8 MB of cache memory) with 12 GB of RAM under the 64-bit Windows 7 Professional operation system. The program code of CTW-HEVC and the original HEVC codecs were compiled in the release mode for a 32-bit platform using the Visual Studio 2005 environment. Multithreaded programming techniques were not used in the implementation of the proposed mechanisms.

4.2 Compression Performance of the CTW-CABAC: Results for HM3.2 software

When operating with the higher number of regular sub-streams (225 regular sub-streams in the HM3.2) the compression performance of CABAC can be reasonably improved by applying the proposed more accurate mechanism of symbol probability estimation. In the framework of HM3.2 software, the application of the CTW-CABAC allows a 1.2%-4.5% reduction of bitrate as compared to the original CABAC algorithm. Detailed experimental results on the efficiency of the first and the second version of the CTW- CABAC within HM3.2 are presented in Table 1, with reference to individual test sequences.

Table 1 Average bitrate reduction due to the application of the CTW-CABAC within HEVC (Version 1 and Version 2 of the algorithm for N=1) in comparison to the original HEVC. Results for HM3.2 software. A positive value denotes bitrate saving (compression gain).

Test sequence	BD Rate [%] (CTW-HEVC **Ver. 1, N=1**)	BD Rate [%] (CTW-HEVC **Ver. 2, N=1**)
Station	1.78%	1.77%
RiverBed	2.90%	3.02%
PoznańStreet	2.21%	1.62%
Balloons	2.15%	1.64%
ChinaSpeed	4.50%	2.91%
SlideEditing	3.39%	3.57%
Mobcal	1.51%	2.08%
ParkRun	1.68%	2.45%
Shields	3.25%	1.79%
Stockholm	2.31%	1.26%
Average:	**2.57%**	**2.21%**

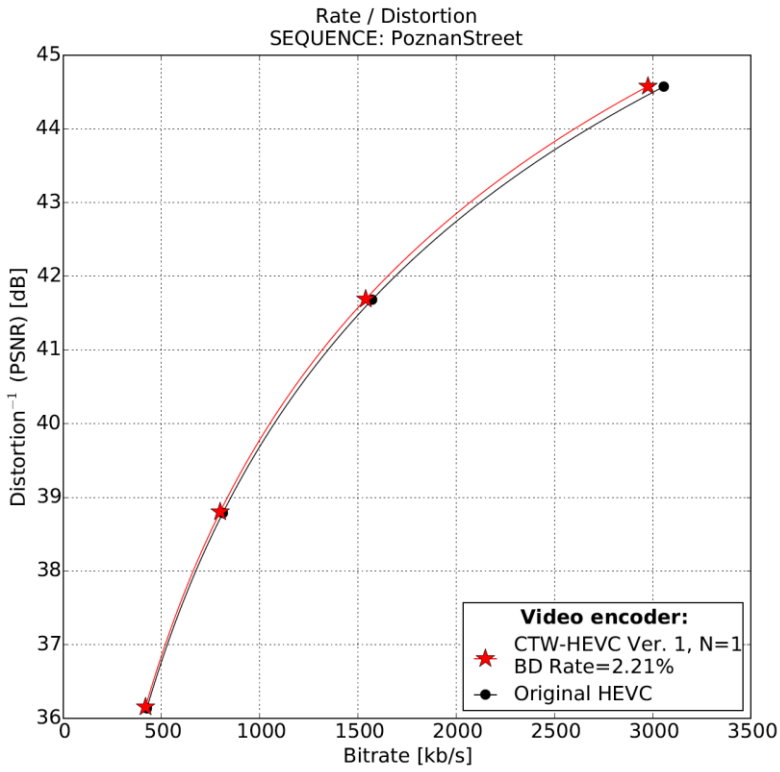
Different results were obtained for respective versions of the CTW-CABAC – with better results for the CTW-CABAC Version 1 algorithm. On average, 2.57% and 2.21% reductions of bitrate were achieved in case of CTW- CABAC Version 1 and CTW- CABAC Version 2, respectively (for N=1). As explained in Sec. 3.4, the two versions of the algorithm differ in the binarization scheme and the manner of statistical modelling for transform coefficient data. In the authors' opinion, lower compression gains for the CTW-CABAC Version 2 result from the fact of performing statistical modelling for bins 3-14 in binarized words of transform coefficient levels. From the statistical point of view, these bins appear relatively rarely in a coded data stream of HEVC – the mechanism of data statistics estimation cannot work efficiently for 'older' bins due to the context dilution problem.

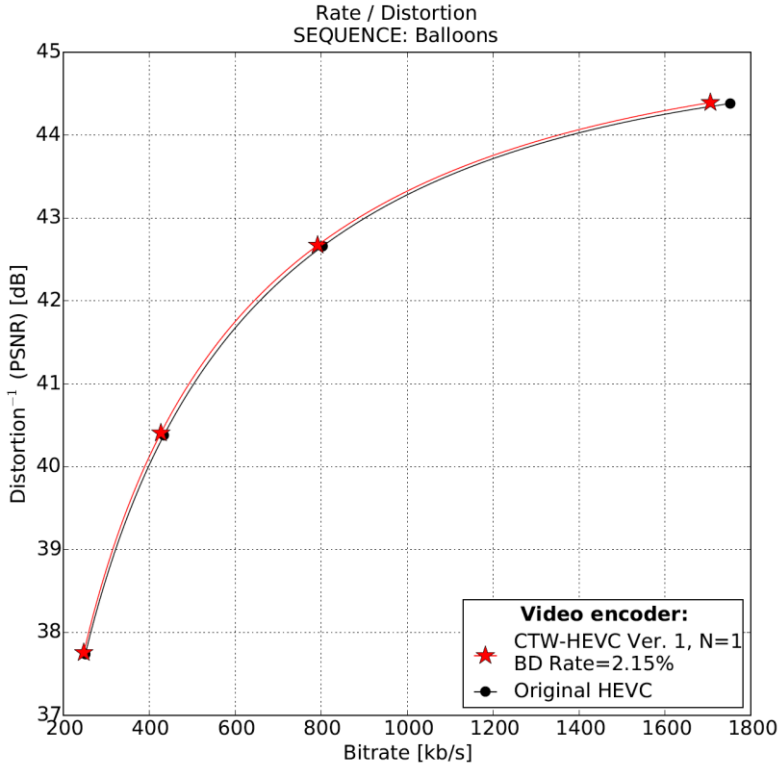
Regardless of the version of the CTW-CABAC, different results were obtained for individual test sequences – the content of a sequence affects the results to a large extent. Moderate

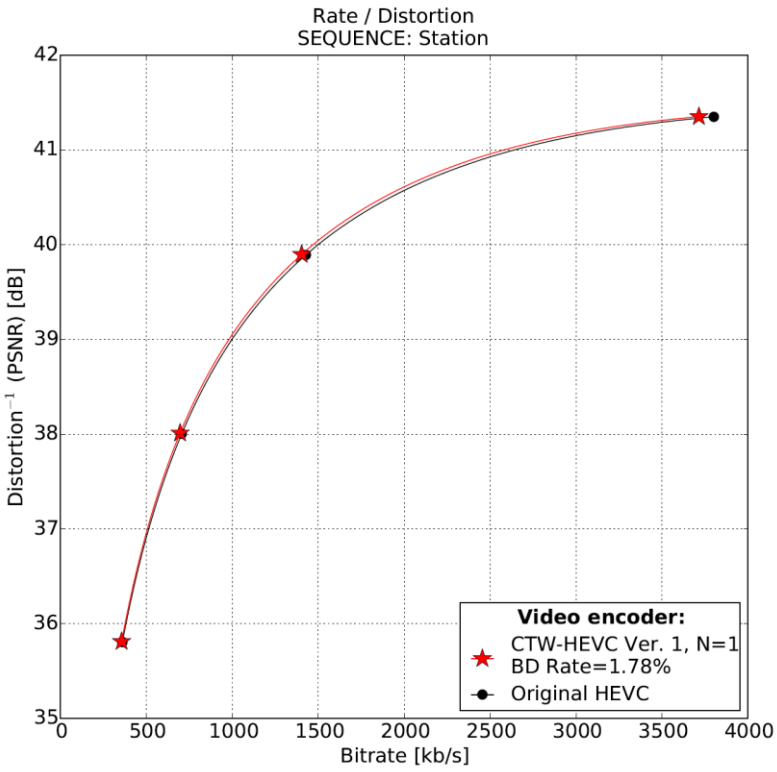
compression gains were achieved for standard test sequences – 1.51%-3.25% for the CTW-CABAC Version 1 algorithm and 1.26%-3.02% for the CTW-CABAC Version 2 algorithm. In general, better results were achieved for sequences with computer graphics and screen content (*ChinaSpeed* and *SlideEditing*) – 4.50% and 3.39% for the CTW-CABAC Version 1 and 2.91% and 3.57% for the CTW-CABAC Version 2. The original CABAC was designed taking into account the character of natural (standard) video sequences. Computer graphics and screen content sequences have a different nature – the CABAC algorithm cannot track the statistics of the signal efficiently in these cases. Additionally, the CTW-CABAC is distinguished by higher adaptability to the current signal statistics, as compared to the original algorithm. Hence, higher compression gains were observed for some non-standard test video sequences (*ChinaSpeed* and *SlideEditing*).

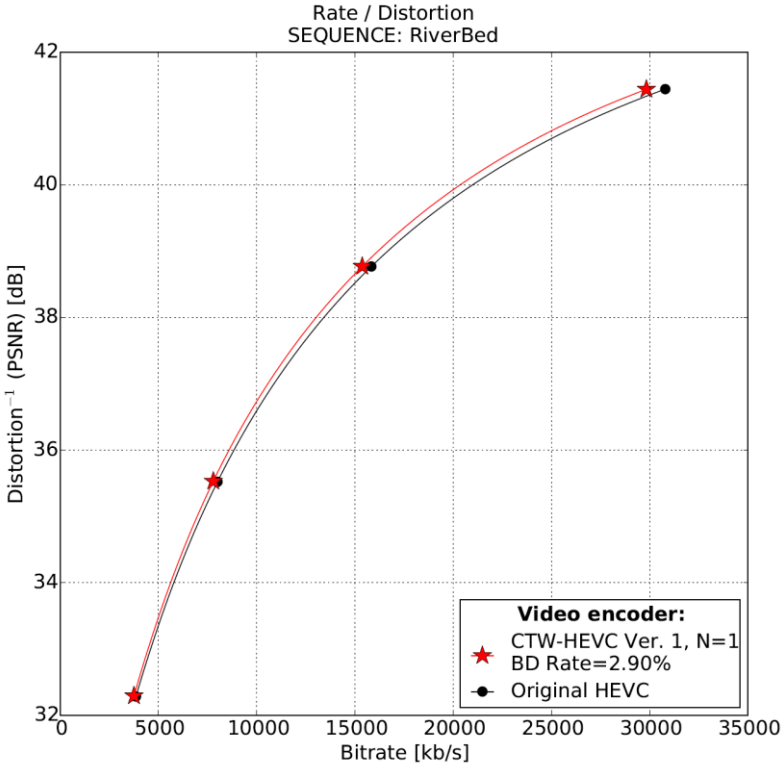
4.3 Compression Performance of the CTW-CABAC: Rate-Distortion^{-1} curves for HM3.2 software

The results presented in the previous section show average compression gains (calculated on the basis of the partial results for QP=22, 27, 32, 37), resulting from application of the CTW-CABAC within the HEVC. In order to better demonstrate the efficiency of the proposed solutions Rate-Distortion^{-1} curves were drawn for three video encoders. These are: 1) CTW-HEVC Version 1 (N=1), 2) CTW-HEVC Version 2 (N=1), and 3) Original HEVC. The charts (presented below) were developed with experimental data obtained for QP=22, 27, 32, 37 for a set of test video sequences.


Rate / Distortion
SEQUENCE: PoznanStreet
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
45
44
43
42
41
40
39
38
37
36
0
500
1000
1500
2000
2500
3000
3500
Video encoder:
CTW-HEVC Ver. 1, N=1
BD Rate=2.21%
Original HEVC


Rate / Distortion
SEQUENCE: Balloons
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
45
44
43
42
41
40
39
38
37
200
400
600
800
1000
1200
1400
1600
1800
Video encoder:
CTW-HEVC Ver. 1, N=1
BD Rate=2.15%
Original HEVC


Rate / Distortion
SEQUENCE: Station
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
42
41
40
39
38
37
36
35
0
500
1000
1500
2000
2500
3000
3500
4000
Video encoder:
CTW-HEVC Ver. 1, N=1
BD Rate=1.78%
Original HEVC


Rate / Distortion
SEQUENCE: RiverBed
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
42
40
38
36
34
32
0
5000
10000
15000
20000
25000
30000
35000
Video encoder:
CTW-HEVC Ver. 1, N=1
BD Rate=2.90%
Original HEVC

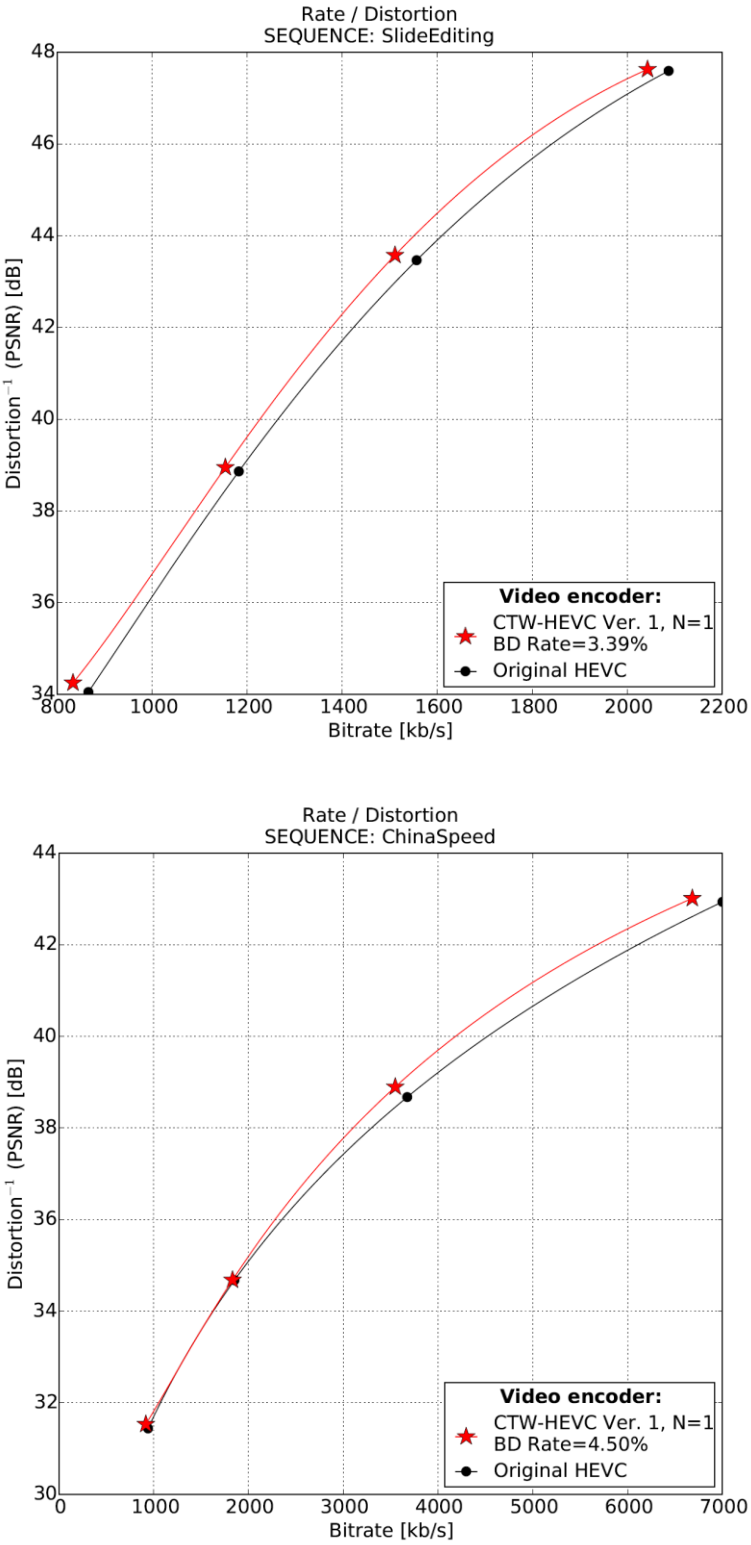
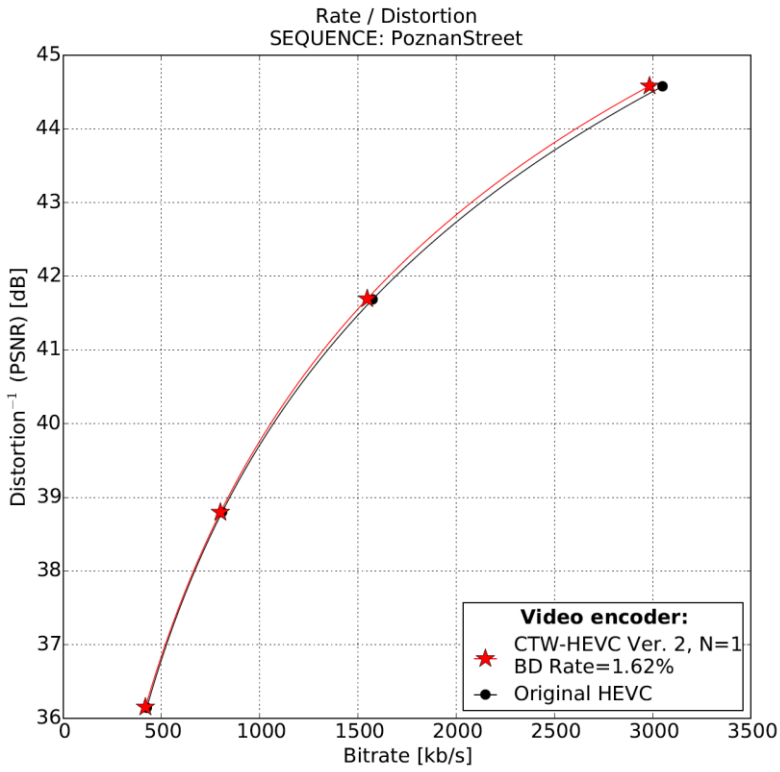
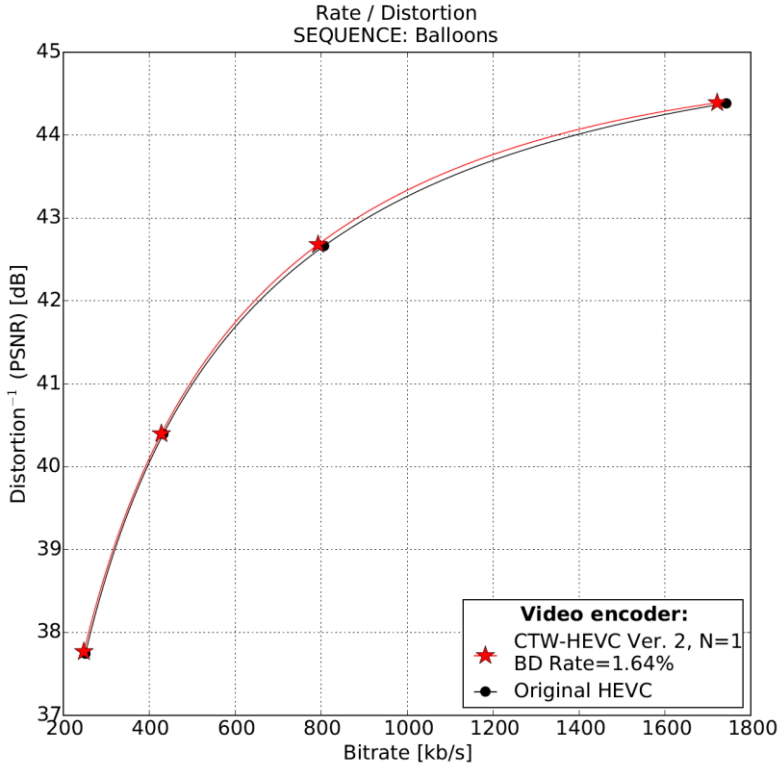
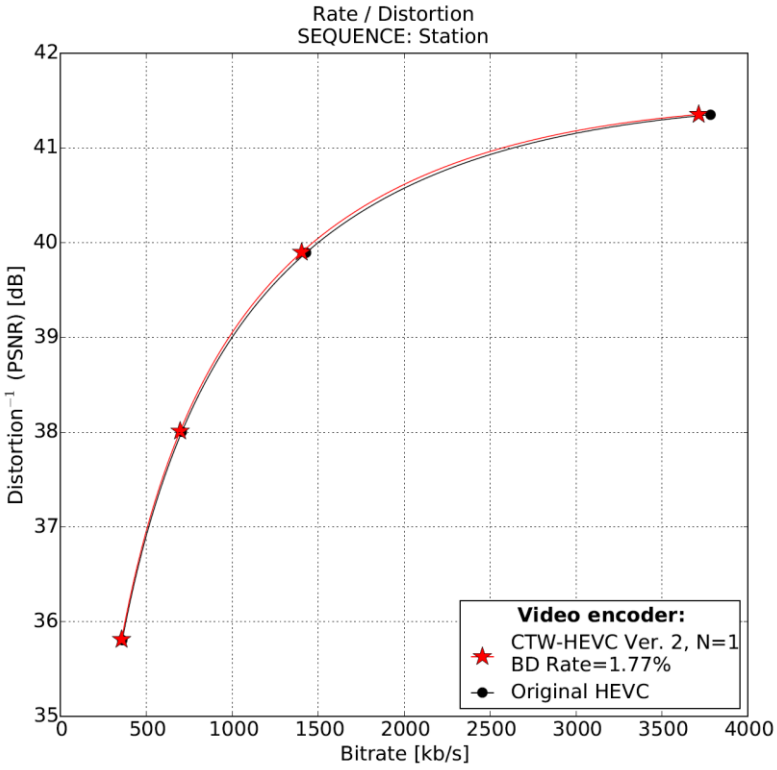
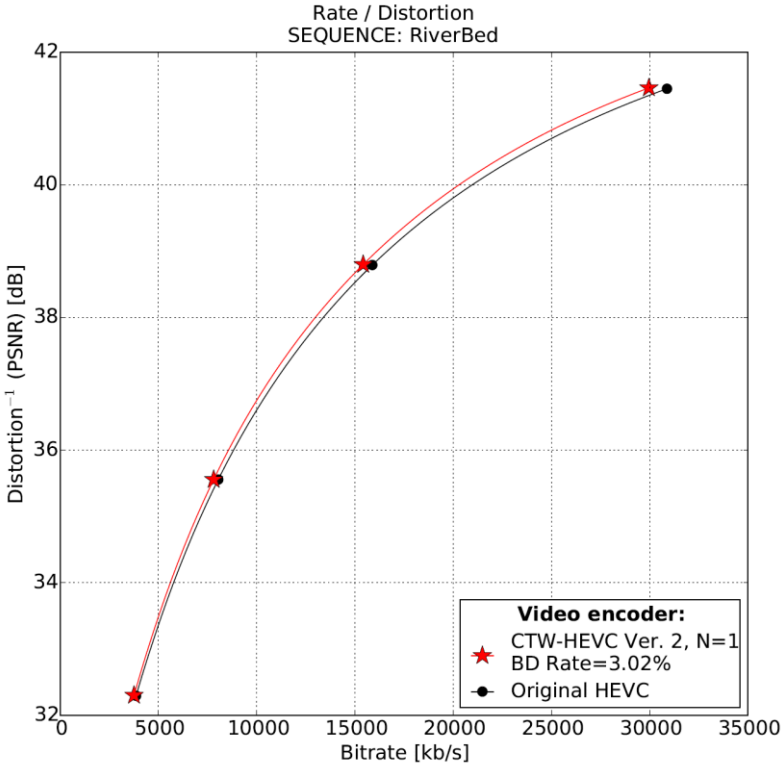


Fig. 6. Rate-Distortion^{-1} curves showing compression performance of two encoders: 1) CTW-HEVC Version 1 (N=1), and 2) Original HEVC. Results for *PoznanStreet*, *Balloons*, *Station*, *RiverBed*, *SlideEditing*, *ChinaSpeed* test sequences. Results for HM3.2 software.


Rate / Distortion
SEQUENCE: PoznanStreet
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
Video encoder:
CTW-HEVC Ver. 2, N=1
BD Rate=1.62%
Original HEVC


Rate / Distortion
SEQUENCE: Balloons
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
Video encoder:
CTW-HEVC Ver. 2, N=1
BD Rate=1.64%
Original HEVC


Rate / Distortion
SEQUENCE: Station
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
42
41
40
39
38
37
36
35
0
500
1000
1500
2000
2500
3000
3500
4000
Video encoder:
CTW-HEVC Ver. 2, N=1
BD Rate=1.77%
Original HEVC


Rate / Distortion
SEQUENCE: RiverBed
Distortion^{-1} (PSNR) [dB]
Bitrate [kb/s]
42
40
38
36
34
32
0
5000
10000
15000
20000
25000
30000
35000
Video encoder:
CTW-HEVC Ver. 2, N=1
BD Rate=3.02%
Original HEVC

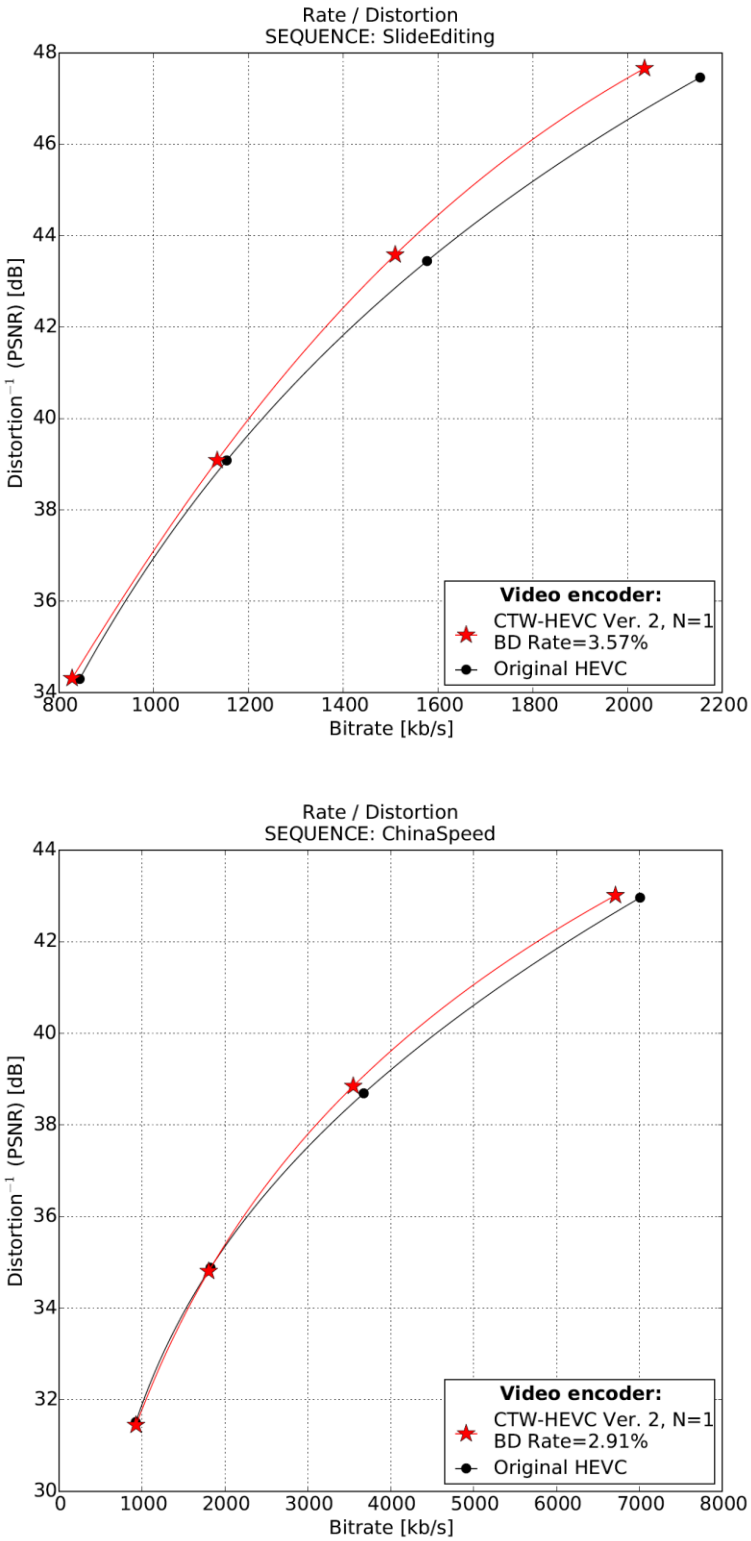


Fig. 7. Rate-Distortion^{-1} curves showing compression performance of two encoders: 1) CTW-HEVC Version 2 (N=1), and 2) Original HEVC. Results for *PoznanStreet*, *Balloons*, *Station*, *RiverBed*, *SlideEditing*, *ChinaSpeed* test sequences. Results for HM3.2 software.

The main observation from the study is that the higher bitrate the greater difference in the efficiency of tested encoders. In this case the entropy encoder calculates probabilities of symbols based on a relatively large set of video data. As a result, the CTW algorithm calculates probabilities of symbols more precisely, which leads to higher efficiency of the CTW-CABAC.

4.4 Compression Performance of the CTW-CABAC: Results for HM16.6 software

In the case of the new HM16.6 version of software experiments were done for the CTW-CABAC Version 1 of the algorithm, since binarization of data in the HM16.6 is based on truncated Golomb- Rice and the 0-th order Exp-Golomb (EG0) codes only (there are no binarization schemes known from the AVC technology). Application of the CTW-CABAC in the HM16.6 improved compression performance of the encoder, however, the observed gains of compression are noticeably smaller when compared to HM3.2 software (see Section 4.2). Detailed results of experiments were presented in Table 2.

Table 2 Average bitrate reduction due to application of the CTW-CABAC within HEVC (Version 1 of the algorithm for N=1) in comparison to the original HEVC. Results for HM16.6 software. A positive value denotes bitrate saving (compression gain).

Test sequence	BD Rate [%] (CTW-HEVC **Ver. 1, N=1**)
Station	0.89%
RiverBed	1.18%
PoznańStreet	0.69%
Balloons	0.89%
ChinaSpeed	0.93%
SlideEditing	3.42%
Mobcal	0.71%
ParkRun	0.66%
Shields	0.84%
Stockholm	0.85%
Average:	**1.11%**

As it can be seen, the CTW-CABAC gives 1.11% BD-Rate saving on average, while for HM3.2 software this profit was 2.57% on average. Analysis of both versions of HM software indicates that the actual set of most important coding tools is the same in the older and the newer version of HM, and the main difference is the number of regular sub-streams for which CABAC realizes sophisticated statistical modeling of data. Due to the smaller number of such sub-streams in the new HM16.6 software (173 instead of 225 in the HM3.2) there is a larger contribution of bypass bins that are encoded assuming uniform probability of symbols. In the opinion of the authors this is the main reason for the difference of results achieved for the HM3.2 and HM16.6. It needs to be stressed that even in the case of smaller number of regular sub-streams the proposed CTW-CABAC algorithm increases efficiency of video coding.

4.5 Entropy codec complexity scalability and its impact on coding efficiency: Results for HM3.2 software

Reduction of the complexity of entropy coding is particularly critical in case of a codec working with a large number of regular sub-streams. Since HM3.2 operates on a higher number of regular sub-streams compared to HM16.6, the entropy codec complexity scalability and its impact on coding efficiency of data was tested using older version of the reference software used as a platform for embedding CTW-CABAC.

Simplification of statistics estimation in the CTW-CABAC (full calculation of probability using CTW for every second and third binary symbol only, for N=2 and N=3, respectively) decreases the compression performance of the entropy encoder (see detailed results in Table 3 and Table 4). In the case of the CTW-CABAC Version 1 algorithm, average bitrate savings of 1.68% (N=2) and 1.50% (N=3) were noticed (in contrast to a bitrate reduction of 2.57% in the case of N=1). Bitrate savings of 1.40% (N=2) and 1.19% (N=3) were observed for the CTW-CABAC

Version 2 (in contrast to the reduction of bitrate by 2.21% for N=1). A similar, relative decrease of the compression gain was recorded for both versions of the CTW-CABAC.

Table 3 Average bitrate reduction due to the application of the CTW-CABAC within HEVC (Version 1 of the algorithm for N=1, N=2, and N=3) in comparison to the original HEVC. Results for HM3.2 software. A positive value denotes bitrate saving (compression gain).

Test sequence	BD Rate [%] (CTW-HEVC **Ver. 1, N=1**)	BD Rate [%] (CTW-HEVC **Ver. 1, N=2**)	BD Rate [%] (CTW-HEVC **Ver. 1, N=3**)
Station	1.78%	1.13%	0.80%
RiverBed	2.90%	2.77%	2.61%
PoznańStreet	2.21%	1.83%	1.60%
Balloons	2.15%	1.62%	1.47%
ChinaSpeed	4.50%	2.39%	2.43%
SlideEditing	3.39%	1.88%	1.81%
Mobcal	1.51%	1.25%	1.01%
ParkRun	1.68%	1.07%	0.95%
Shields	3.25%	0.97%	1.12%
Stockholm	2.31%	1.93%	1.22%
Average:	**2.57%**	**1.68%**	**1.50%**

Table 4 Average bitrate reduction due to the application of the CTW-CABAC within HEVC (Version 2 of the algorithm for N=1, N=2, and N=3) in comparison to the original HEVC. Results for HM3.2 software. A positive value denotes bitrate saving (compression gain).

Test sequence	BD Rate [%] (CTW-HEVC **Ver. 2, N=1**)	BD Rate [%] (CTW-HEVC **Ver. 2, N=2**)	BD Rate [%] (CTW-HEVC **Ver. 2, N=3**)
Station	1.77%	1.16%	0.89%
RiverBed	3.02%	2.89%	2.82%
PoznańStreet	1.62%	1.10%	1.20%
Balloons	1.64%	1.42%	0.89%
ChinaSpeed	2.91%	1.72%	-0.70%
SlideEditing	3.57%	0.64%	3.11%
Mobcal	2.08%	1.38%	1.22%
ParkRun	2.45%	0.96%	0.78%
Shields	1.79%	1.56%	1.30%
Stockholm	1.26%	1.17%	0.37%
Average:	**2.21%**	**1.40%**	**1.19%**

4.6 Impact of the CTW-CABAC on complexity of the HEVC codec

For the same reasons discussed in the previous section, complexity of the HEVC video codec with CTW-CABAC has been tested for HM3.2 version of software. In the following two subsections, the results on the complexity of both the HEVC-based codec are presented.

4.6.1 Complexity of the HEVC decoder with CTW-CABAC: Results for HM3.2 software

The proposed mechanism of symbol probability estimation increases the compression gain of the video encoder. However, it is obtained at the cost of higher complexity of the video decoder. Relative increase in complexity due to application of CTW-CABAC in the HEVC decoder is presented in the following tables. The tables present detailed experimental results achieved for both versions of the CTW-CABAC. In all cases the anchor was the complexity of the HEVC decoder with the original CABAC.

Table 5 Complexity of the CTW-HEVC Version 1 video decoder. Detailed results for values N=1, N=2, and N=3.

Test sequence	complexity increase of CTW-HEVC **Ver. 1** decoder (N=1) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 1** decoder (N=2) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 1** decoder (N=3) [%] (relative to original HEVC)
Station			
QP=22 (3781 kbps*)	28.6	21.7	18.8
QP=27 (1430 kbps*)	16.3	12.5	9.3
QP=32 (707 kbps*)	11.8	8.8	7.8
QP=37 (361 kbps*)	6.0	4.8	3.7
RiverBed			
QP=22 (30874 kbps*)	112.6	88.6	82.1
QP=27 (15880 kbps*)	66.2	50.5	46.1
QP=32 (8041 kbps*)	41.4	31.2	28.2
QP=37 (3858 kbps*)	28.5	21.9	21.6
PoznańStreet			
QP=22 (3050 kbps*)	31.6	24.6	21.2
QP=27 (1574 kbps*)	20.0	14.8	11.6
QP=32 (808 kbps*)	14.2	13.9	13.2
QP=37 (425 kbps*)	8.8	8.8	8.2
Balloons			
QP=22 (1742 kbps*)	26.6	21.4	19.5
QP=27 (807 kbps*)	16.5	19.4	11.8
QP=32 (433 kbps*)	9.5	8.8	7.4
QP=37 (251 kbps*)	7.9	7.0	11.5
ChinaSpeed			
QP=22 (7007 kbps*)	80.4	65.9	59.6
QP=27 (3675 kbps*)	55.7	49.1	45.5
QP=32 (1827 kbps*)	38.1	31.8	27.5
QP=37 (926 kbps*)	26.3	23.3	20.3
SlideEditing			
QP=22 (2153 kbps*)	47.7	41.2	39.4
QP=27 (1577 kbps*)	35.8	30.1	30.3
QP=32 (1154 kbps*)	29.2	23.8	24.0
QP=37 (844 kbps*)	19.8	13.9	15.6
Average:	**32.5**	**26.6**	**24.3**

* bitrate for the original HEVC (with original CABAC)

Table 6 Complexity of the CTW-HEVC Version 2 video decoder. Detailed results for values N=1, N=2, and N=3.

Test sequence	complexity increase of CTW-HEVC **Ver. 2** decoder (N=1) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 2** decoder (N=2) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 2** decoder (N=3) [%] (relative to original HEVC)
Station			
QP=22 (3781 kbps*)	33.1	27.3	23.7
QP=27 (1430 kbps*)	17.4	13.5	12.2
QP=32 (707 kbps*)	11.6	9.5	6.2
QP=37 (361 kbps*)	6.5	5.6	3.8
RiverBed			
QP=22 (30874 kbps*)	123.0	96.6	83.9
QP=27 (15880 kbps*)	69.8	55.1	48.2
QP=32 (8041 kbps*)	43.9	34.4	29.5
QP=37 (3858 kbps*)	29.4	22.7	20.5
PoznańStreet			
QP=22 (3050 kbps*)	34.9	25.2	23.3
QP=27 (1574 kbps*)	18.2	14.7	12.7
QP=32 (808 kbps*)	13.6	9.6	8.2
QP=37 (425 kbps*)	10.1	8.8	7.2
Balloons			
QP=22 (1742 kbps*)	31.8	26.1	23.1
QP=27 (807 kbps*)	19.4	15.7	15.3
QP=32 (433 kbps*)	12.4	10.6	9.1
QP=37 (251 kbps*)	8.8	8.2	6.5
ChinaSpeed			
QP=22 (7007 kbps*)	95.3	77.6	69.6
QP=27 (3675 kbps*)	65.9	53.6	46.0
QP=32 (1827 kbps*)	43.1	37.7	33.0
QP=37 (926 kbps*)	29.3	22.3	20.2
SlideEditing			
QP=22 (2153 kbps*)	71.5	57.3	50.9
QP=27 (1577 kbps*)	55.6	43.6	34.2
QP=32 (1154 kbps*)	38.4	32.1	27.3
QP=37 (844 kbps*)	26.0	22.6	20.7
Average:	**37.9**	**30.4**	**26.5**

* bitrate for the original HEVC (with original CABAC)

The application of the CTW-CABAC increases the complexity of the video decoder. The percentage increase of video decoder complexity depends on both the content of video and the bitrate of encoded data stream, the average increase is 32% (for the CTW-HEVC Version 1) and 38% (for the CTW-HEVC Version 2). Higher complexity of the CTW-HEVC Version 2 revealed from higher number of regular bins that represent the transform coefficient levels, for which complexity of decoding is higher relative to the complexity of bypass bins decoding[54]. For the range of bitrates useful in high definition television (below 4 Mbps for the HEVC technology), the CTW-CABAC increases the complexity of the video decoder by 30%-40%.

The complexity of the CTW-CABAC can be reduced when probabilities of some symbols are calculated in a simplified manner. Experiments performed for the scenarios when the full calculation of probability (with CTW) was performed for every 2nd and 3rd binary symbol (for N=2 and N=3 respectively) revealed that 4.8% and 7.5% reductions of the CTW-HEVC decoder complexity were possible. Nevertheless, the complexity reduction is obtained at the cost of compression performance loss as revealed in the Sec. 4.5, while the decoder complexity still remains high.

Presented results show the potential of further works on optimization of the CTW-CABAC. Theoretically, the algorithm can be further speeded up when by parallelization (i.e. with the use of multiple processor cores).

4.6.2 Complexity of the HEVC encoder with CTW-CABAC: Results for HM3.2 software

The proposed CTW-CABAC algorithm only marginally affects the complexity of video encoding. Extensive tests were conducted in order to provide real values showing the difference in complexity of the HEVC encoder that works with the original or with the improved CABAC algorithm. Results are in Table 7.

Table 7 Complexity of the CTW-HEVC Version 1 video encoder. Detailed results for values N=1, N=2, and N=3.

Test sequence	complexity increase of CTW-HEVC **Ver. 1** encoder (N=1) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 1** encoder (N=2) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 1** encoder (N=3) [%] (relative to original HEVC)
Station			
QP=22 (3781 kbps*)	0.2	1.7	1.3
QP=27 (1430 kbps*)	0.4	1.7	1.3
QP=32 (707 kbps*)	0.5	1.5	1.5
QP=37 (361 kbps*)	0.9	1.8	1.4
RiverBed			
QP=22 (30874 kbps*)	0.7	1.9	2.2
QP=27 (15880 kbps*)	0.2	1.2	1.3
QP=32 (8041 kbps*)	0.0	1.0	1.1
QP=37 (3858 kbps*)	-2.0	-1.7	-2.3
PoznańStreet			
QP=22 (3050 kbps*)	0.0	1.9	1.0
QP=27 (1574 kbps*)	0.2	1.8	0.7
QP=32 (808 kbps*)	-0.1	1.5	0.3
QP=37 (425 kbps*)	0.4	1.5	0.9
Balloons			
QP=22 (1742 kbps*)	0.6	1.6	1.0
QP=27 (807 kbps*)	0.7	1.7	1.1
QP=32 (433 kbps*)	0.2	1.3	0.7
QP=37 (251 kbps*)	0.9	2.6	1.7
ChinaSpeed			
QP=22 (7007 kbps*)	0.9	1.7	-3.8
QP=27 (3675 kbps*)	0.4	1.5	-3.5
QP=32 (1827 kbps*)	0.8	1.9	-0.4
QP=37 (926 kbps*)	-0.6	-0.2	-0.2
SlideEditing			
QP=22 (2153 kbps*)	-0.7	-0.2	-1.0
QP=27 (1577 kbps*)	-0.6	-0.9	-0.8
QP=32 (1154 kbps*)	-2.0	-1.5	-2.2
QP=37 (844 kbps*)	-0.7	-1.4	-2.4
Average:	**0.1**	**1.0**	**0.0**

* bitrate for the original HEVC (with original CABAC)

Table 8 Complexity of the CTW-HEVC Version 2 video encoder. Detailed results for values N=1, N=2, and N=3.

Test sequence	complexity increase of CTW-HEVC **Ver. 2** encoder (N=1) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 2** encoder (N=2) [%] (relative to original HEVC)	complexity increase of CTW-HEVC **Ver. 2** encoder (N=3) [%] (relative to original HEVC)
Station			
QP=22 (3781 kbps*)	-2.3	-2.9	-2.7
QP=27 (1430 kbps*)	-2.5	-2.9	-2.7
QP=32 (707 kbps*)	-2.4	-2.9	-2.7
QP=37 (361 kbps*)	-1.3	-1.9	-1.6
RiverBed			
QP=22 (30874 kbps*)	-3.6	-7.4	-7.5
QP=27 (15880 kbps*)	-3.5	-1.9	-3.9
QP=32 (8041 kbps*)	-1.9	-2.1	-1.8
QP=37 (3858 kbps*)	-2.0	-2.7	-1.9
PoznańStreet			
QP=22 (3050 kbps*)	-2.7	-3.0	-2.9
QP=27 (1574 kbps*)	-3.2	-3.6	-3.2
QP=32 (808 kbps*)	-2.9	-3.7	-3.7
QP=37 (425 kbps*)	-3.2	-3.8	-3.6
Balloons			
QP=22 (1742 kbps*)	-1.8	-1.8	-1.5
QP=27 (807 kbps*)	-2.0	-1.7	-1.9
QP=32 (433 kbps*)	-1.8	-1.9	-2.2
QP=37 (251 kbps*)	-0.9	-1.0	-1.0
ChinaSpeed			
QP=22 (7007 kbps*)	-1.4	-1.6	-2.5
QP=27 (3675 kbps*)	-1.6	-2.5	-2.4
QP=32 (1827 kbps*)	-1.2	-1.0	-1.7
QP=37 (926 kbps*)	-2.4	-2.8	-2.6
SlideEditing			
QP=22 (2153 kbps*)	-3.2	-3.8	-4.0
QP=27 (1577 kbps*)	-1.8	-2.3	-3.3
QP=32 (1154 kbps*)	-1.0	-1.9	-1.9
QP=37 (844 kbps*)	11.1	9.8	10.5
Average:	**-1.7**	**-2.1**	**-2.2**

* bitrate for the original HEVC (with original CABAC)

The fact that there are no significant differences in the complexity of the encoders results from relatively small contribution of entropy coding into the total encoding time. In a video encoder, the largest part of the processing time is consumed by the selection of the coding modes for image blocks (the procedure that is not performed at the decoder side). The following actions: selection of the size of image blocks, motion estimation performed for blocks that are encoded with inter-frame prediction and selection of intra-frame prediction mode require multiple encoding of blocks of image in order to select the best variant of compression. Due to the reasons above and because of the fact that full CABAC encoding is performed only once (for already selected mode of coding), the proposed improvements of CABAC have marginal impact on the complexity of the whole HEVC encoder. However, it should be noted that the used method of entropy coding (original or improved) affects the way the encoder selects the compression modes for image blocks in general. Thus, a given block does not have to be encoded with the same coding mode in the original and the improved HEVC encoders. However, it does not change the main conclusion of the experiment.

4.7 Memory Complexity of the CTW-CABAC

The original CABAC algorithm is characterized by low memory requirements – less than 500 bytes for context initialization tables (for 225 contexts), 225 bytes to store information on FSMs, and less than 400 bytes for arithmetic encoder core look-up tables. Thus 1125 bytes of data are needed in total (it must be stressed that the size of the memory needed for look-up tables can be further reduced when optimizing the implementation). The proposed mechanism of statistics estimation in CABAC creates requirements for higher memory usage. Each node of the context tree consumes 4 bytes of data to represent the statistics – 2 bytes for the counters of symbols 0 and 1, and 2 bytes for β^s balance switch used in the optimized implementation of CTW (refer to[45] for

more details). A context tree of depth D=8 contains 511 nodes. The CTW-CABAC presented in the paper utilizes 225 such context trees. Thus, it approximately gives 449 kbytes of data needed to store context trees only. Apart from this, about 14 kbytes of memory are used to store CTW- related look-up tables, less than 400 bytes for arithmetic encoder core look-up tables, and 500 bytes for context initialization tables. In total, it gives less than 470 kbytes of memory. Such amount of memory is not a problem for contemporary processing units. It is also possible to additionally compress some of the tables. Nevertheless, the memory requirements of the CTW-CABAC are significantly higher compared to the original CABAC algorithm.

5. Conclusions

Compression performance of the CABAC entropy encoder can be reasonably increased by using a more accurate mechanism of conditional probability estimation for data symbols. The paper proves that it can be efficiently done using the proposed mechanism of data statistics estimation that is based on the CTW technique. The proposed improvement of CABAC leads to improved compression performance of the state-of-the-art video encoders. The paper focuses on the results of experiments that were performed in the context of the state-of-the art HEVC video compression technology. In the framework of HEVC, a high efficiency of the improved version of CABAC was confirmed when operating with different schemes of binarization of data and different number of regular sub-streams used in the entropy codec. In each of the considered cases the improved CABAC gave bitrate saving – the proposed solution offers 0.7%-4.5% bitrate reduction of the HEVC-encoded stream, depending on the content and encoder configuration. Thus, the experimental results suggest that the proposed technique could be considered for applications in the future video encoders of the next generations.

The proposed mechanism of probability estimation affects both the complexity and the memory demand of a video decoder. When applying an algorithmically optimized version of the CTW-CABAC in a HEVC decoder, it increases the complexity of a video decoder by a factor of 1.3-1.4 for a range of bitrates below 4 Mbps. There is also a significant increase in memory usage; nevertheless, it does not create a problem for contemporary processing units as the required memory volumes remain less than a megabyte.

Acknowledgment

The work was supported with Polish public funds as a research project No. N N516 440438.

References

1. ISO/IEC 13818-2 and ITU-T Rec. H.262, Generic Coding of Moving Pictures and Associated Audio Information – Part 2: Video. (MPEG-2). November 1994.
2. ITU-T Rec. H.263, Video Coding for Low Bit Rate Communication. August 2005.
3. Society of Motion Picture and Television Engineers, VC-1 Compressed Video Bitstream Format and Decoding Process. SMPTE 421M-2006, 2006.
4. Audio Video Coding Standard Workgroup of China (AVS), The Standards of People's Republic of China GB/T 20090.2-2006, Information Technology – Advanced Coding of Audio and Video – Part 2:Video. 2006.
5. ISO/IEC 14496-10, Generic Coding of Audio-Visual Objects, Part 10: Advanced Video Coding. March 2006.
6. ISO/IEC and ITU-T, High Efficiency Video Coding (HEVC), ISO/IEC 23008-2 (MPEG-H Part 2) / ITU-T Rec. H.265, April 2013.
7. T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC Video Coding Standard. IEEE Transactions on Circuits and Systems for Video Technology 13(7), 560-576, 2003.

8. G. J. Sullivan, J. –R. Ohm, W. –J. Han, T. Wiegand, Overview of the High Efficiency Video Coding (HEVC) Standard. IEEE Transactions on Circuits and Systems for Video Technology 22(12), 1649-1668, 2012.
9. D. A. Huffman, A Method for the Construction of Minimum-Redundancy Codes. Proceedings of the I. R. E, pp. 1098-1101, September, 1952.
10. D. Salomon, G. Motta, Handbook of Data Compression. Springer-Verlag, 2010.
11. I. H. Witten, J. G. Cleary, and R. Neal, Arithmetic Coding for Data Compression. Communications of the ACM., no. 6, pp. 520-540, June 1987.
12. A. Said, Introduction to Arithmetic Coding – Theory and Practice. Imaging systems Laboratory, HP Laboratories Palo Alto, April 21, 2004.
13. Liquan Shen, Ping An, Xinpeng Zhang, Zhaoyang Zhang, Adaptive transform size decision algorithm for high-efficiency video coding inter coding, Journal of Electronic Imaging, 07/2014; 23(4):043023.
14. D. Marpe, H. Schwarz, and T. Wiegand, Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, pp. 620-636, July 2003.
15. V. Sze, M. Budagavi, High Throughput CABAC Entropy Coding in HEVC. IEEE Transactions on Circuits and Systems for Video Technology 22(12), 1778-1791, 2012.
16. Yongseok Choi, Jongbum Choi, High-throughput CABAC codec architecture for HEVC, Electronics Letters, Volume 49, Issue 18, August 2013, pp. 1145-1147.
17. T. Davies, CE1: Subtest 12, Entropy Coding Comparisons with Simplified RDO. JCTVC-G210, JCT-VC Meeting, Geneva, November 2011.
18. M. Mrak, D. Marpe, and S. Grgic, Comparison of Context-Based Adaptive Binary Arithmetic Coders in Video Compression. Proc. EC-VIP-MC'03, July 2003.
19. M. Mrak, D. Marpe, and T. Wiegand, Application of Binary Context Trees in Video Compression. Picture Coding Symposium (PCS'03), St. Malo, France, April 2003.

20. M. Mrak, D. Marpe, and T. Wiegand, A Context Modeling Algorithm and its Application in Video Coding. IEEE International Conference on Image Processing (ICIP'03), Barcelona, Spain, September 2003.
21. D. Karwowski, Improved Adaptive Arithmetic Coding in MPEG-4 AVC/H.264 Video Compression Standard. Advances in Intelligent and Soft Computing vol. 102, Image Processing and Communications Challenges 3, Springer-Verlag 2011, ss. 257-263.
22. M. Ghandi, M. M. Ghandi, and M. B. Shamsollahi, A Novel Context Modeling Scheme for Motion Vectors Context-Based Arithmetic Coding. IEEE Canadian Conference on Electrical & Computer Engineering (CCECE04), May 2-5, 2004, Ontario, Canada.
23. S. Milani, and G. A. Mian, An Improved Context Adaptive Binary Arithmetic Coder for the H.264/AVC Standard. Proc. of European Signal Processing Conference (EUSIPCO 2006), September 4-8, 2006, Florence, Italy.
24. D. Karwowski, Improved Arithmetic Coding in H.264/AVC Using Context-Tree Weighting and Prediction by Partial Matching. European Signal Processing Conf. EUSIPCO 2007, pp. 1270-1274, September 2007, Poznań, Poland.
25. D. Karwowski, and M. Domański, Improved Arithmetic Coding In H.264/AVC Using Context-Tree Weighting Method. Picture Coding Symposium PCS 2007, November 2007, Lisboa, Portugal.
26. D. Karwowski, M. Domański, Improved Context-Adaptive Arithmetic Coding in H.264/AVC. European Signal Processing Conference EUSIPCO 2009, August 2009, Glasgow, Scotland, pp. 2216-2220.
27. E. Belyaev, M. Gilmutdinov, and A. Turlikov, Binary Arithmetic Coding System with Adaptive Probability Estimation by “Virtual Sliding Window”. IEEE International Symposium on Consumer Electronics, pp. 1-5, 2006.
28. D. Hong, M. van der Schaar, B. Pesquet – Popescu, Arithmetic Coding with Adaptive Context-Tree Weighting for the H.264 Video Coders. Visual Communications and Image Processing 2004, Vol. 5308, pp. 1226-1235, January 2004.

29. M. H. Firooz, M. S. Sadri, Improving H.264/AVC Entropy Coding Engine Using CTW method. Picture Coding Symposium, April 2006.
30. T. Nguyen, M. Winken, D. Marpe, H. Schwarz, T. Wiegand, Reduced-complexity entropy coding of transform coefficient levels using a combination of VLC and PIPE. JCTVC- D336, JCT-VC Meeting, Daegu, January 2011.
31. D. Marpe, H. Schwarz, and T. Wiegand, Probability Interval Partitioning Entropy Codes. Submitted to IEEE Transactions on Information Theory, 2010. Available at iphome.hhi.de/marpe/download/pipe-subm-ieee10.pdf.
32. C. Auyeung, J. Xu, Context Reduction of Significance Map Coding with CABAC. JCTVC-G366, JCT-VC Meeting, Geneva, November 2011.
33. K. Misra, A. Segall, CABAC Simplification for Explicit Signaling Mode of AMVP. JCTVC-G705, JCT-VC Meeting, Geneva, November 2011.
34. W. -J. Chien, J. Sole, M. Karczewicz, Context Reduction for CABAC. JCTVC-G718, JCT-VC Meeting, Geneva, November 2011.
35. A. Alshin, E. Alshina, J. Park, High Precision Probability Estimation for CABAC. Visual Communications and Image Processing (VCIP) 2013, pp. 1-6.
36. A. Alshin, E. Alshina, J. Park, CE1 (subset B): Multi-Parameter Probability Up-Date for CABAC. Document of Joint Collaborative Team on Video Coding (JCT-VC), JCTVC-G0764, November 2011.
37. J. Stegemann, H. Kirchhoffer, D. Marpe, T. Wiegand, Counter-based Probability Model Update with Adapted Arithmetic Coding Engine. Document of Joint Collaborative Team on Video Coding (JCT-VC), JCTVC-G547, November 2011.
38. B. Li, G. Sullivan, J. Xu: Comparison of Compression Performance of HEVC Working Draft 5 with AVC High Profile. JCTVC- H0360, JCT-VC Meeting, San Jose, February 2012.

39. J. –R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, T. Wiegand, Comparison of the Coding Efficiency of Video Coding Standards – Including High Efficiency Video Coding (HEVC). IEEE Transactions on Circuits and Systems for Video Technology 22(12), 1669-1684, 2012.
40. M. Domański, Approximate bitrate estimation for television services, 112th MPEG Meeting, Warsaw, June 2015, Doc. M36571.
41. D. Karwowski, Improved Adaptive Arithmetic Coding for HEVC Video Compression Technology. Lecture Notes in Computer Science, no. 7594, ss. 121-128, Springer-Verlag (Proceedings of ICCVG 2012 Warsaw, Poland, September 24-26, 2012).
42. P. G. Howard, and J. S. Vitter, Practical Implementations of Arithmetic Coding. Image and Text Compression, pp. 85-112, Kluwer Academic, 1992.
43. F. M. J. Willems, Y. M. Shtarkov, and Tj. J. Tjalkens, The Context-Tree Weighting Method: Basic Properties. IEEE Transactions on Information Theory, Vol. 41, No. 3, pp. 653-664, May 1995.
44. F. M. J. Willems, The Context-Tree Weighting Method: Extensions. IEEE Transactions on Information Theory, Vol. 44, No. 2, pp. 792-797, March 1998.
45. F. M. J. Willems, and Tj. J. Tjalkens, Reducing Complexity of the Context-Tree Weighting Method. Proc. IEEE International Symposium on Information Theory, p. 347, Cambridge, Mass., August 16-21, 1998.
46. J. G. Cleary, I. H. Witten, Data Compression Using Adaptive Coding and Partial String Matching. IEEE Transactions on Communication, Vol. 32, No. 4, pp. 396-402, April 1984.
47. R. Begleiter, R. El-Yaniv, and G. Yona, On Prediction Using Variable Order Markov Models. Journal of Artificial Intelligence Research, Vol. 22 (2004), pp. 385-421, December 2004.
48. P. A. J. Volf, Weighting Techniques in Data Compression: Theory and Algorithms. Doctoral Dissertation, Technische Universiteit Eindhoven, Eindhoven 2002, available: http://alexandria.tue.nl/extra2/200213835.pdf.

49. T. Nguyen, M. Winken, D. Marpe, H. Schwarz, T. Wiegand, Reduced-complexity entropy coding of transform coefficient levels using a combination of VLC and PIPE. JCTVC- D336, JCT-VC Meeting, Daegu, January 2011.
50. D. Marpe, G. Marten, and H. L. Cycon, A Fast Renormalization Technique for H.264/MPEG4-AVC Arithmetic Coding. 51st Internationales Wissenschaftliches Kolloquium, Technische Universitat Ilmenau, September 2006.
51. F. Bossen, D. Flynn, K. Suhring, AHG Report – Software Development and HM Software Technical Evaluation. JCTVC- G003, JCT-VC Meeting, Geneva, November 2011.
52. Karsten Suehring, Karl Sharman, JCT-VC AHG report: HEVC HM software development and software technical evaluation (AHG3), JCTVC- U0003, JCT-VC Meeting, Warsaw, June 2015.
53. G. Bjøntegaard, Calculation of Average PSNR Differences between RD curves. ITU-T SG16/Q6, 13th VCEG Meeting, Austin, Texas, USA, April 2001, Doc. VCEG-M33.
54. D. Karwowski, Computational Complexity Analysis of Adaptive Arithmetic Coding in HEVC Video Compression Standard. Advances in Intelligent and Soft Computing vol. 233, Image Processing and Communications Challenges 5, ss. 133-142, Springer, 2014.

Damian Karwowski (born 1979) received his M.Sc. and Ph.D. degrees from Poznan University of Technology in 2003 and 2008 respectively. Currently he is an assistant professor at Poznan University of Technology, a member of the Chair of Multimedia Telecommunications and Microelectronics. He is an author and co-author of over 30 articles on digital video coding in both national and international conferences. He has been taking part in several industry-oriented projects that encompasses: H.264/AVC video codec software and hardware development, MPEG-4 AAC HE audio decoder DSP implementation, H.263 videoconference codec DSP implementation, MPEG-2 to H.264/AVC video transcoder implementation, automatic image quality improvement, optimization of video and audio codecs. He was Technical Program Chair of Picture Coding Symposium, PCS 2012, and member of the organizing committee of Int. Workshop on Signals, Systems and Image Processing, IWSSIP 2014, Int. Conf. Signals and Electronic Systems, ICSES 2004, 73rd Meeting of MPEG and European Signal Processing Conference, EUSIPCO 2007. He is laureate of Leader IV program that was organized by The National Centre of Research and Development in Poland.
His interests are centered on image and audio compression algorithms, entropy coding techniques for digital audio and digital video, and realization of video and audio codecs on PC, DSP and FPGA platforms.

Marek Domański (M'91) received M.S., Ph.D. and Habilitation degrees from Poznań University of Technology, Poland in 1978, 1983 and 1990 respectively. Since 1993, he is a Professor at Poznań University of Technology, where he leads the Chair (Department) of Multimedia Telecommunications and Microelectronics. Since 2005 he is the head of Polish delegation to MPEG. He authored highly ranked technology proposals submitted in response to MPEG calls for scalable video compression (2004) and 3D video coding (2011). He also leaded the team that developed one of the very first AVC decoders for tv set-top boxes (2004) and various AVC and AAC HE codec implementations and improvements. He is an author or co-author of 3 books and over 200 papers in journals and proceedings of international conferences. The contributions were mostly on image, video and audio compression, image processing, multimedia systems, 3D video and color image technology, digital filters and multidimensional signal processing. He was General Chairman and host of several international conferences: Picture Coding Symposium, PCS 2012; European Signal Processing Conference, EUSIPCO 2007; 73rd Meeting of MPEG; Int. Workshop on Signals, Systems and Image Processing, IWSSIP 1997 and 2004; Int. Conf. Signals and Electronic Systems, ICSES 2004 and others. He served as a member of various steering, program and editorial committees of international journals and international conferences. He served as Area Editor of Signal Processing: Image Communications journal in 2005-2010.