INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2018, VOL. 64, NO. 3, PP. 277–284
Manuscript received May 21, 2018; revised July, 2018. DOI: 10.24425/123519

Increased Compression Efficiency of AVC and HEVC CABAC by Precise Statistics Estimation

Damian Karwowski and Marek Domański

***Abstract*—The paper presents Improved Adaptive Arithmetic Coding algorithm for application in future video compression technology. The proposed solution is based on the Context-based Adaptive Binary Arithmetic Coding (CABAC) technique and uses the authors mechanism of symbols probability estimation that exploits Context-Tree Weighting (CTW) technique. This paper proposes the version of the algorithm, that allows an arbitrary selection of depth *D* of context trees, when activating the algorithm in the framework of the AVC or HEVC video encoders. The algorithm has been tested in terms of coding efficiency of data and its computational complexity. Results showed, that depending on depth of context trees from 0.1% to 0.86% reduction of bitrate is achieved, when using the algorithm in the HEVC video encoder and 0.4% to 2.3% compression gain in the case of the AVC. The new solution increases complexity of entropy encoder itself, however, this does not cause an increase of the complexity of the whole video encoder.**

***Keywords*—Entropy coding, CABAC algorithm, CTW technique, Video Compression, AVC, HEVC**

I. Introduction

AT the output of contemporary video encoders entropy coding is always used. By the use of entropy coding one is able to additionally reduce redundancy that exists in the data that are produced by the preceding functional blocks of video encoder. This paper concerns this stage of video encoding. In particular, the paper focuses on methods of entropy coding used in the newest technologies of video compression.

By the way of improvements in successive generations of video encoders (MPEG-2, H.263, VC-1, AVS, AVC, HEVC) the entropy coding has been also a subject of modernizations [1]- [5]. Especially a large progress has been made in the use of arithmetic coding in video encoders [6]. In particular, in the course of years of research, the Context-based Adaptive Binary Arithmetic Coding (CABAC) [7], [17] technique has been elaborated. It has been already experimentally proven, that coding efficiency of CABAC is superior to other competitive solutions used in video encoders [7]. For the abovementioned reason, the CABAC technique found practical application firstly in Advanced Video Coding (AVC) standard (ISO/IEC MPEG-4 AVC and ITU-T Rec. H.264) [3] and lastly in the newest High Efficiency Video Coding (HEVC) technology [4], [5], that was recently elaborated by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Pictures Experts Group (MPEG). At the same time CABAC has set a new trend in entropy coding of video data and became a starting point in most research works that were carried out in last years on the topic of entropy compression [8]- [16], [21], [23], [24].

The CABAC algorithm realizes adaptive binary arithmetic coding of data, but this idea is realized in a highly sophisticated manner. First of all, the algorithm processes binary, and not m-ary, symbols of data. That is to say, it uses binary arithmetic encoder core. Thus, in the first stage of the algorithm the data are mapped to string of binary symbols in a binarizer block. For this purpose many different methods of mapping of m-ary data are used, in order to take into account specific nature of data that is encoded in the encoder. This step of the CABAC algorithm is just the advanced variable-length coding of data. The resulted stream of binary symbols is in a thoughtful way divided into smaller sub-streams (streams of binary symbols), that gathers the data whose nature is similar (similar in terms of data statistics). Symbols of individual sub-streams are then arithmetically encoded, after determining the probability of the symbols occurrence in data stream. The whole processing of data is realized according to the block diagram presented in fig. 1.

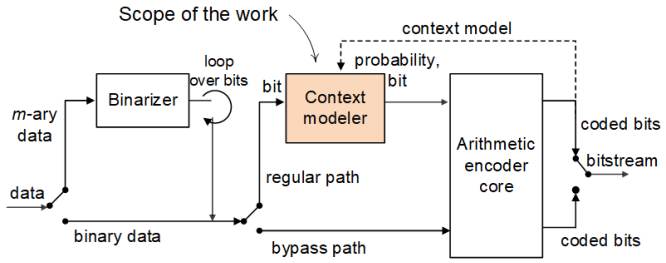

Fig. 1. Block diagram of the original CABAC. The scope of the work has been highlighted on the diagram.

It is commonly known, that the way of symbols probability estimation strongly influences efficiency of the algorithm. In the CABAC those probabilities are calculated when taking a number of simplifying assumptions. First of all, probabilities are calculated by the use of the Finite State Machines (FSM), with a pre-defined, small number of different states. The states correspond to the values of symbols probabilities, so small number of states translates into small number of different probabilities used in the algorithm. To be more precisely, only 64 values of probabilities for 0 symbol and 64 probabilities for 1 symbol are used in the CABAC. Secondly, although a large number of FSMs are used in the algorithm (one FSM for each sub-stream of data), only one, the same procedure of probabilities update is used in FSMs. Mentioned simplifications of CABAC speed up significantly processing of data, but reduce

The work was supported by National Science Centre, Poland, according to the decision DEC-2012/05/B/ST7/01279.

D. Karwowski and M. Domański are with the Chair of Multimedia Telecommunications and Microelectronics, Poznan University of Technology, POLAND, (e-mail: damian.karwowski@put.poznan.pl, marek.domanski@put.poznan.pl).

at the same time the performance of data compression. This opens the way to research on improving the efficiency of the algorithm. Such studies are at the focus of this work.

II. Attempts of Improving the CABAC and Main Goal of this Work

In the past few years a number works have been presented which tried to omit the above-mentioned simplifications of the CABAC [8]- [16], [21], [23], [24]. These were proposals for different solutions in which statistics of symbols were estimated with even greater precision than in the original algorithm.

The authors of this paper have also carried out such works. In their course the authors were developing two different approaches.

The essence of the first approach was to calculate probabilities of symbols when utilizing only local statistics of data, i.e. statistics gathered in the four, previously encoded blocks of the image (Coding Tree Unit (CTU) - in case of HEVC encoder), that are adjacent to the currently encoded block (see fig. 2).

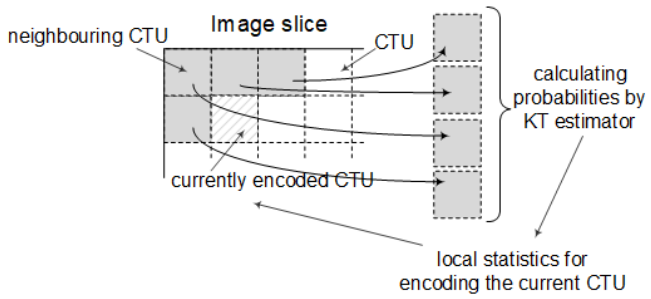

Fig. 2. Idea of symbols probabilities estimation based on statistics of local data. In the figure, KT denotes Krichevski-Trofimov estimator, which is known in the literature.

When implementing the idea in the framework of HEVC CABAC (with independent estimation of statistics for each of 173 data sub-streams), unfortunately, the efficiency of the CABAC decreased by 4% in average. A detailed analysis of the causes showed that amount of binary symbols that is presened in a single block of the image (e.g. CTU in HEVC) is too small to be able to properly evaluate the statistics of symbols, when doing it independently for each of the 173 data sub-streams. In the case of the HEVC video compression with a value of quantization parameter (QP) equal to 25 it is only 200 to 1000 binary symbols, depending on the image type and content of the test sequence.

The second approach developed by the authors exploits Context Tree Weighting (CTW) algorithm [18]-[20], to estimate probabilities of symbols with much higher precision. This idea has been explored by the authors for years, in details in the framework of the AVC, and in the framework of the HEVC video compression technologies for the selected algorithm settings [9]-[14],[24]. As it was experimentally proved, the new mechanism of probabilities estimation developed on the basis of CTW has a great potential when using in contemporary video encoders.

Today, activities are carried out towards the future video compression technology, which is to become the successor of the HEVC encoding. From this point of view, results obtained for the HEVC are of special importance. In this case the version of the improved CABAC (based on CTW) has been considered by the authors only, in which a fixed depth of context trees was used [14],[24]. Following conclusions of the results obtained for AVC it was depth $D = 8$. In the AVC encoder, such a setting was the best compromise between the complexity and efficiency of the algorithm.

However, this does not have to be the case with the HEVC video encoder. Therefore, this paper presents the new version of the algorithm, that gives the possibility to arbitrary setting up the depth D for context trees. The new version of the algorithm allows a more thorough study of parameters of the improved algorithm (like efficiency and complexity). Additionally, the paper confronts results obtained for the HEVC encoder with those for AVC encoder.

III. Proposed Solution and Specific Goals of the Work

This paper presents the improved version of the CABAC algorithm. The main idea of the improvement is to replace the original, simplified algorithm of symbols probabilities estimation by a new method, based on CTW, that calculates probabilities of symbols with much higher accuracy. As many studies have shown, a more accurate estimation of the data statistics may be a source of increasing the efficiency of entropy coding.

The proposed method uses a binary tree, of a depth D, that stores information about the number of 0 and 1 symbols that appeared in the coded data stream when D previously encoded symbols (the so-called context information) had certain values. Thanks to the information stored in the tree, the algorithm can determine the conditional probability of occurrence of 0 and 1 symbols under a given context (so, values of D previously encoded symbols). Since it is not known in advance how long the context should be to evaluate accurately the statistics of the data, the algorithm performs specific weighing of probabilities, achieved for different length of context. This is the main idea of the CTW algorithm a detailed description of way of probabilities weighting can be found in the works [18]-[20]. It is the basis of the solution presented in this work.

From the perspective of the entropy coding efficiency, the way of adopting CTW in the entropy encoder is of great importance. In the proposed solution, 173 context trees are used, which replace 173 FSMs that are used in the original CABAC technique (see fig. 3). Thus, individual context trees provide the necessary data for the estimation of statistics of 0 and 1 in each of the 173 independent binary streams that are utilized in the HEVC CABAC algorithm. It must be emphasized here that the used by the authors way of adopting CTW in the CABAC algorithm is the most advanced among methods that have been considered in the literature so far.

The depth D of context trees may have a significant impact on the performance of the improved entropy encoder. Therefore, in this work, various depths D of trees are considered to

answer the following question: how does the depth of context trees affect parameters (like efficiency and complexity) of the improved CABAC algorithm, when encoding the data of the HEVC video encoder? The purpose of this work is to examine this relationship.

A more accurate estimation of symbols statistics leads to a much broader set of different values of probabilities than it is the case in the original CABAC. Unfortunately, the core of arithmetic encoder that is used in CABAC can only work with a pre-defined limited set of quantized 128 values of probabilities. To meet the arithmetic encoder core requirements, probabilities obtained with the use of CTW are quantized to one of the values defined in the original CABAC. The criterion of probability quantization is to minimize the absolute difference between two values: probability of the original CABAC and the one calculated with the CTW algorithm.

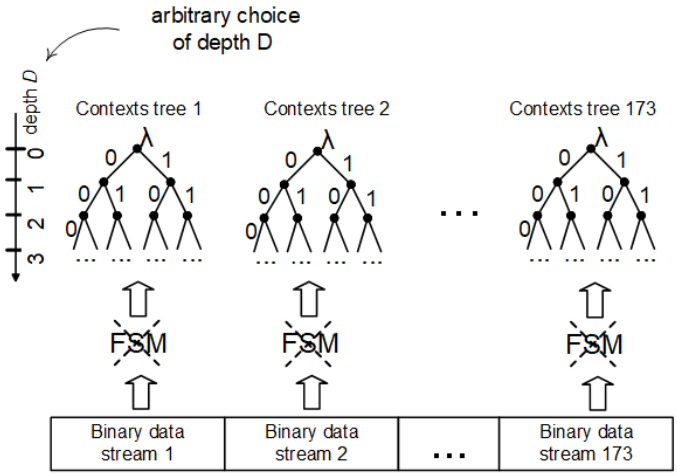

Fig. 3. The way of adopting the CTW technique in the proposed improved CABAC.

IV. Methodology of Experiments

In order to thoroughly investigate the parameters of the presented algorithm when compressing data of the latest video encoders, the algorithm was implemented and launched in the HM 16.6 model software of the HEVC codec. With the use of the prepared software performance tests of the developed method were performed, obtaining numerical values showing compression ratio of video data and computational complexity of the algorithm.

The parameters of the proposed algorithm were then confronted with the results of the original HM software, which utilizes the original version of the CABAC algorithm. The original HEVC encoder was therefore the reference point to the results obtained for the new method.

Parameters of both encoders were measured by compressing a set of high resolution test video sequences. The study was conducted using well-known test sequences: *BasketballDrive*, *BQTerrace*, *Cactus*, *Kimono1*, *ParkScene*. These sequences have been recommended by the ISO/IEC MPEG and ITU-T VCEG expert groups as appropriate testing material for studies on new technologies of video coding.

During experiments, both the encoders, the modified and the reference, were configured in order to use the compression tools listed in the ”common test conditions” recommendation [25]. In particular, the video coding scenario was assumed, which was oriented towards achieving very high compression efficiency of a video data. According to the cited recommendation, the single experiment involved 4 times compression of each test sequence, each time setting up different value of quantization parameter QP (QP = 22, 27, 32, 37). The resulting 4 measurement points were later compared to the 4 points achieved for the original HEVC encoder using a commonly used in the field of video compression Bjontegaard metric (known as BD-rate) [22]. For each single experiment (comprising encoding for QP = 22, 27, 32, 37), this metric compares the two encoders in terms of average bitrate reduction based on the four Rate-Distortion points. Computational complexity of the algorithms is measured by the time of their execution on a processor of the computer.

The extended version of the improved CABAC presented in this paper can work with arbitrarily defined depth of CTW context trees. The purpose of the experiments was to examine the influence of the depth of trees used, on selected parameters of the algorithm, like compression efficiency and coding complexity. Therefore, the coding of a set of test sequences was performed many times during the test, each time using a different depth of context trees in the improved algorithm.

The experimental results were then the basis for determining the optimal settings for the new encoder (which is the best value for the depth of the context trees), which, for HEVC technology, gives the best compromise between algorithm efficiency and time of data processing.

V. Results of Experiments

A. Gain of Compression

Presented in the paper improvement indeed increases compression performance of the standard CABAC algorithm. Obtained results clearly proved that the exact degree of compression performance improvement depends strongly on depth D of context trees.

According to the results (see Table I) , the use of ”shallow” context trees allows for only minor improvement of data coding efficiency. A good example of this is the depth of trees $D = 2$, where performance improvements oscillate only within 0.1%-0.5%. In order to get even better results, ”deeper” context trees must be used. And so, increasing the depth of trees in the range of 2 to 12 further increase of the algorithm efficiency is observed, but this gain is very negligible. Such an observation shows the results obtained for the following tree depth values, i.e. for $D = 4$, 8, 10 and 12. For these values, the efficiency of the algorithm was increased from 0.1% to 0.68% in average, with respect to the results of the original CABAC technique.

When evaluating obtained results, it should be taken into account that achieved compression gain applies to that part of the video encoder, where only lossless coding is realized.

The experiments also showed that the results obtained were not identical for the individual test sequences. This is because

the coded content also affects the results of data compression. As it is known, the content of the sequence translates into specific decisions of the encoder in terms of the coding modes used for image blocks, which in turn translates into the form of residual data that is put to entropy encoder in the final phase of video compression.

TABLE I
BITRATE SAVINGS WHEN USING IN HEVC THE IMPROVED CABAC RELATIVE TO THE STANDARD HEVC ENCODER WITH THE ORIGINAL CABAC. NEGATIVE VALUES IN THE TABLE MEANS GAIN OF COMPRESSION.

	BD-rate (CABAC CTW vs original CABAC)					
sequence	D=2	D=4	D=8	D=10	D=12	D=14
BasketballDrive	-0.25%	-0.50%	-0.62%	-0.63%	-0.64%	-0.64%
BQTerrace	-0.45%	-0.65%	-0.75%	-0.76%	-0.76%	-0.76%
Cactus	-0.10%	-0.36%	-0.49%	-0.49%	-0.49%	-0.49%
Kimono1	-0.55%	-0.74%	-0.85%	-0.86%	-0.86%	-0.86%
ParkScene	-0.40%	-0.58%	-0.67%	-0.68%	-0.68%	-0.68%
Average:	**-0.35%**	**-0.56%**	**-0.68%**	**-0.68%**	**-0.68%**	**-0.69%**

B. Complexity of the Improved Algorithm

The higher (in relation to the reference, original algorithm) efficiency of the improved algorithm is occupied by the increased complexity of the data processing. In the case of the entropy encoder alone, this is an increase in complexity even several times. As the research results show such an increase of the algorithm complexity does not lead to an increase in the complexity of the entire video encoder. Detailed results of Table II show well this conclusion. It results from the very small complexity of the entropy encoder part in relation to the operation time of the video encoder.

However, the situation is different in the case of a video decoder. Here, increasing the depth of context trees by 2 increases the complexity of the video decoder by several to dozen percent in average (see Table III). Farther, apart from depth D, also content of encoding material affects complexity factor. It results directly from differences of encoded data stream bitrate, that represents the individual sequences. The larger the data stream, the bigger contribution of entropy decoding in the total time of video decompression. From the point of view of CTW algorithm, the greater the depth D is, the longer context path on a tree, along which the algorithm calculates values of partial probabilities. Those probabilities are then mixed together to calculate the new probability that is put to arithmetic encoder core.

VI. CONCLUSIONS AND FINAL REMARKS

Relative to the solution used in the original CABAC the proposed CTW-based probability estimation algorithm allows better estimation of symbols statistics, which turn into higher efficiency of video encoding. When coding syntax elements of the new HEVC standard, 0.1% to 0.86% compression gain was achieved when using the improved solution. The exact gain depends to the greatest extent on depth D of context trees.

The smallest increase in coding efficiency was obtained for context trees with low depth. In the case of depth $D = 2$ the average increase in coding efficiency was 0.35%. Increasing the depth of trees improves the compression results, but this increase should be assessed as moderate.

The presented method was previously also the subject of intensive research by the authors in the AVC encoder, a compression technology that precedes the new HEVC standard. For the purpose of this paper efficiency of the proposed modified CABAC algorithm has been re-examined within AVC for the same set of test sequences considered in this paper. Detailed results of this research were presented in Table VI. When comparing experimental data of Tables I and VI it is easy to notice the significant differences in the achieved gains of data compression. These gains were significantly higher when using the algorithm in the AVC encoder for depth $D = 8$ of context trees the average gain was 1.8%, while in the case of the HEVC it was 0.7%. The difference in compression gains is due to two things. First of all, in the CABAC of AVC higher number of bins is encoded in the more efficient regular path (relative to bypass path see fig. 1). This creates greater possibilities for bins compression. Secondly, much more powerful image prediction tools are used in the new HEVC codec, which results in a much smaller residual signal, than in the case of AVC encoding. The form of this signal is that it is very difficult to predict its values, which limits the possibilities of its entropy compression. This is clearly seen by comparing the results of Table IV and Table V that represent the actual capability of the arithmetic encoder core to compress the binary symbols. This capability is significantly lower in the case of data in the HEVC.

Research has also revealed how difficult it is to improve further the results of such an advanced entropy encoder as CABAC is. It is easier to see this when realizing how strong the symbols present at the output of a binarizer block of CABAC are compressed in the remaining part of the algorithm. As results of Table IV and Table V show it is a reduction in the number of bits by 24% and 18% in average, for the AVC and the HEVC encoder respectively! Taking into account that the data binarizer works like a quite advanced VLC coding, such a level of data stream reduction really deserves recognition. These results simply show how little space is left in the CABAC algorithm for an additional improvement of its efficiency.

The proposed solution increases the complexity of the entropy encoder, even several times. However, this did not translate into the complexity of the entire video encoder. This is due to the small contribution of entropy encoding in the total time of the HEVC encoder. On the other hand the proposed method increases in a visible way complexity of video decoder.

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] ISO/IEC 14496-10, Generic Coding of Audio-Visual Objects, Part 10: Advanced Video Coding, March 2006.
[4] 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.
[5] 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.
[6] 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.
[7] 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.
[8] 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.
[9] 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, Pozna, Poland.
[10] 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.
[11] D. Karwowski, M. Domański, Improved Context-Adaptive Arithmetic Coding in H.264/AVC, European Signal Processing Conference EUSIPCO 2009, August 2009, Glasgow, Scotland.
[12] D. Karwowski, M. Domański, Improved Context Based Adaptive Binary Arithmetic Coding in MPEG-4 AVC/H.264 Video Codec. Lecture Notes in Computer Science, no. 6375, ss. 25-32, Springer-Verlag (Proceedings ICCVG 2010 Warsaw, Poland, September 20-22, 2010).
[13] 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, ss. 257-263.
[14] 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).
[15] 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.
[16] 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.
[17] X. Tian, T. M. Le, Y. Lian, Entropy Coders of the H.264/AVC Standard, Springer-Verlag, 2011.
[18] 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.
[19] F. M. J. Willems, The Context-Tree Weighting Method: Extensions, IEEE Transactions on Information Theory, Vol. 44, No. 2, pp. 792-797, March 1998.
[20] 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.
[21] M. H. Firooz, M. S. Sadri, Improving H.264/AVC Entropy Coding Engine Using CTW method, Picture Coding Symposium, April 2006.
[22] G. Bjntegaard, Calculation of Average PSNR Differences between RD curves, ITU-T SG16/Q6, 13th VCEG Meeting, Austin, Texas, USA, April 2001, Doc. VCEG-M33.
[23] A. Alshin, E. Alshina, J. Park, High precision probability estimation for CABAC, Visual Communications and Image Processing (VCIP) 2013, pp. 1-6.
[24] D. Karwowski, M. Domański, Context-Adaptive Binary Arithmetic Coding with Precise Probability Estimation and Complexity Scalability for High-Efficiency Video Coding, Journal of Electronic Imaging, 25(1), 013010 (January 20, 2016); DOI: 10.1117/1.JEI.25.1.013010.
[25] F. Bossen, Common test conditions and software reference configurations, Joint Collaborative Team on Video Coding (JCT-VC) of ITUT SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-J1100, Stockholm, Sweden, July 2012.

TABLE II
RELATIVE COMPLEXITY OF THE HEVC ENCODER WITH THE IMPROVED CABAC, WHEN COMPARED TO COMPLEXITY OF THE ORIGINAL HEVC ENCODER (WITH STANDARD CABAC).

			Complexity of the modified HEVC encoder relative to the HEVC with the original CABAC					
Test sequence	**QP**	**Rate, kbit/s***	**D = 2**	**D = 4**	**D = 8**	**D = 10**	**D = 12**	**D = 14**
BasketballDrive	QP=22	16612.89	0.990	1.008	1.048	1.001	0.997	1.043
	QP=27	6111.55	0.986	0.976	1.020	0.986	0.983	0.990
	QP=32	2860.27	0.992	0.993	1.000	0.992	0.990	1.003
	QP=37	1502.54	0.998	0.996	1.001	0.997	0.994	0.996
BQTerrace	QP=22	34998.69	1.019	1.008	1.088	0.997	1.005	1.039
	QP=27	7599.53	0.983	0.976	0.995	0.983	0.976	0.984
	QP=32	2716.33	0.987	0.989	1.003	0.990	0.990	1.005
	QP=37	1279.24	0.999	0.993	0.998	0.999	1.118	0.995
Cactus	QP=22	17506.91	1.015	1.012	1.118	1.051	1.048	1.081
	QP=27	5984.06	1.000	0.972	1.038	1.000	0.990	1.003
	QP=32	2845.55	0.986	1.000	0.996	0.985	1.010	0.996
	QP=37	1464.54	0.993	0.995	1.001	0.991	0.990	1.003
Kimono1	QP=22	4607.59	1.020	1.034	1.076	1.002	0.984	1.085
	QP=27	2120.24	0.991	1.029	1.044	1.011	0.985	1.012
	QP=32	1029.79	0.994	1.045	1.059	0.990	0.983	1.001
	QP=37	520.20	0.994	0.951	1.041	0.981	0.971	0.977
ParkScene	QP=22	7150.87	1.024	0.996	1.087	1.043	1.018	1.117
	QP=27	3091.82	1.030	0.981	1.071	0.982	1.001	0.992
	QP=32	1410.19	0.998	0.977	1.061	1.011	0.974	0.984
	QP=37	649.98	0.990	0.992	1.030	0.985	0.984	1.011
		Average:	1.000	0.996	1.039	0.999	1.000	1.016
	* bitrate for the original CABAC in HEVC							

TABLE III
RELATIVE COMPLEXITY OF THE HEVC DECODER WITH THE IMPROVED CABAC, WHEN COMPARED TO COMPLEXITY OF THE ORIGINAL HEVC DECODER (WITH STANDARD CABAC).

			Complexity of the modified HEVC decoder relative to the HEVC decoder with the original CABAC					
	QP	**Rate, kbit/s***	**D = 2**	**D = 4**	**D = 8**	**D = 10**	**D = 12**	**D = 14**
BasketballDrive	QP=22	16612.89	1.025	1.368	1.625	1.743	2.122	2.448
	QP=27	6111.55	0.848	1.177	1.208	1.176	1.252	1.373
	QP=32	2860.27	0.958	1.008	1.060	1.113	1.151	1.246
	QP=37	1502.54	0.947	0.989	1.049	1.063	1.093	1.144
BQTerrace	QP=22	34998.69	1.163	1.565	1.868	2.044	2.455	2.909
	QP=27	7599.53	0.985	1.049	1.162	1.237	1.310	1.447
	QP=32	2716.33	0.956	1.000	1.022	1.041	1.059	1.123
	QP=37	1279.24	1.017	1.025	1.050	1.088	1.065	1.105
Cactus	QP=22	17506.91	1.118	1.496	1.784	1.958	2.335	2.699
	QP=27	5984.06	0.905	1.164	1.204	1.158	1.227	1.402
	QP=32	2845.55	0.982	1.025	1.084	1.148	1.189	1.300
	QP=37	1464.54	0.933	0.963	1.027	1.042	1.062	1.131
Kimono1	QP=22	4607.59	0.969	1.206	1.348	1.495	1.821	1.899
	QP=27	2120.24	0.923	1.237	1.350	1.363	1.474	1.592
	QP=32	1029.79	0.936	1.201	1.278	1.221	1.258	1.394
	QP=37	520.20	0.758	1.060	1.053	0.888	0.993	0.995
ParkScene	QP=22	7150.87	1.041	1.407	1.533	1.795	2.203	2.489
	QP=27	3091.82	0.962	1.290	1.496	1.387	1.559	1.854
	QP=32	1410.19	0.957	1.335	1.318	1.298	1.376	1.476
	QP=37	649.98	0.778	0.974	1.071	0.886	0.873	0.922
		Average:	**0.958**	**1.177**	**1.280**	**1.307**	**1.444**	**1.597**
	* bitrate for the original CABAC in HEVC							

TABLE IV
RATIO OF THE NUMBER OF BITS PRODUCED BY THE CABAC ALGORITHM TO THE NUMBER OF BINARY SYMBOLS THAT ARE PRODUCED BY THE BINARIZER BLOCK OF CABAC ALGORITHM. RESULTS FOR THE HEVC VIDEO ENCODER.

Test sequence			Relation of the number of bits (at the output of CABAC encoder) to the number of binary symbols encoded in CABAC			
	QP	Rate, kbit/s*	original CABAC	D = 2	D = 4	D = 8
BasketballDrive	QP=22	16612.89	0.84	0.837	0.834	0.832
	QP=27	6111.55	0.84	0.837	0.834	0.833
	QP=32	2860.27	0.823	0.820	0.817	0.815
	QP=37	1502.54	0.796	0.794	0.791	0.789
BQTerrace	QP=22	34998.69	0.802	0.795	0.794	0.793
	QP=27	7599.53	0.829	0.823	0.821	0.820
	QP=32	2716.33	0.819	0.813	0.811	0.809
	QP=37	1279.24	0.76	0.756	0.753	0.752
Cactus	QP=22	17506.91	0.845	0.843	0.841	0.839
	QP=27	5984.06	0.847	0.845	0.842	0.841
	QP=32	2845.55	0.826	0.824	0.821	0.820
	QP=37	1464.54	0.797	0.796	0.793	0.791
Kimono1	QP=22	4607.59	0.834	0.826	0.823	0.820
	QP=27	2120.24	0.831	0.823	0.821	0.819
	QP=32	1029.79	0.813	0.806	0.804	0.803
	QP=37	520.20	0.778	0.773	0.771	0.771
ParkScene	QP=22	7150.87	0.859	0.855	0.852	0.850
	QP=27	3091.82	0.851	0.846	0.844	0.843
	QP=32	1410.19	0.83	0.824	0.822	0.821
	QP=37	649.98	0.793	0.788	0.786	0.785
		Average:	0.821	0.816	0.814	0.812
	* bitrate for the original CABAC in HEVC					

TABLE V
RATIO OF THE NUMBER OF BITS PRODUCED BY THE CABAC ALGORITHM TO THE NUMBER OF BINARY SYMBOLS THAT ARE PRODUCED BY THE BINARIZER BLOCK OF CABAC ALGORITHM. RESULTS FOR THE AVC VIDEO ENCODER.

test sequence			Relation of the number of bits (at the output of CABAC encoder) to the number of bins encoded in CABAC	
	QP	Rate, kbit/s*	original CABAC	D = 8
BasketballDrive	QP=22	44998.43	0.779	0.765
	QP=27	16823.5	0.797	0.780
	QP=32	7989.13	0.772	0.752
	QP=37	4535.68	0.731	0.707
BQTerrace	QP=22	100996.38	0.73	0.716
	QP=27	24368.81	0.773	0.758
	QP=32	7224.11	0.78	0.762
	QP=37	3223.45	0.722	0.704
Cactus	QP=22	51405.31	0.762	0.749
	QP=27	14897.89	0.813	0.798
	QP=32	6577.78	0.778	0.763
	QP=37	3375.68	0.718	0.703
Kimono1	QP=22	12171.77	0.801	0.781
	QP=27	5420.22	0.769	0.750
	QP=32	2668.97	0.715	0.696
	QP=37	1437.04	0.662	0.642
ParkScene	QP=22	16816.22	0.816	0.798
	QP=27	6943.38	0.826	0.809
	QP=32	3112.37	0.799	0.783
	QP=37	1535.05	0.74	0.724
		Average:	0.764	0.747
	* bitrate for the original CABAC in AVC			

TABLE VI
BITRATE SAVINGS WHEN USING IN AVC THE IMPROVED CABAC RELATIVE TO THE STANDARD AVC ENCODER WITH THE ORIGINAL CABAC. NEGATIVE VALUES IN THE TABLE MEANS GAIN OF COMPRESSION.

	BD-rate					
test sequence	**D=2**	**D=4**	**D=8**	**D=10**	**D=12**	**D=14**
BasketballDrive	-0.41%	-1.33%	-1.97%	-2.07%	-2.12%	-2.14%
BQTerrace	-0.75%	-1.38%	-1.74%	-1.78%	-1.80%	-1.80%
Cactus	-0.57%	-1.23%	-1.57%	-1.60%	-1.62%	-1.63%
Kimono1	-1.14%	-1.82%	-2.13%	-2.21%	-2.25%	-2.25%
ParkScene	-0.79%	-1.39%	-1.70%	-1.74%	-1.75%	-1.75%
Average:	**-0.73%**	**-1.43%**	**-1.82%**	**-1.88%**	**-1.91%**	**-1.92%**