High Throughput Entropy Coding in the HEVC Standard

Jung-Ah Choi & Yo-Sung Ho

Journal of Signal Processing Systems
for Signal, Image, and Video Technology
(formerly the Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology)

ISSN 1939-8018
Volume 81
Number 1

J Sign Process Syst (2015) 81:59-69
DOI 10.1007/s11265-014-0900-5

Springer

Your article is protected by copyright and all rights are held exclusively by Springer Science +Business Media New York. This e-offprint is for personal use only and shall not be self-archived in electronic repositories. If you wish to self-archive your article, please use the accepted manuscript version for posting on your own website. You may further deposit the accepted manuscript version in any repository, provided it is only made publicly available 12 months after official publication or later and provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com".

Author's personal copy

J Sign Process Syst (2015) 81:59–69
DOI 10.1007/s11265-014-0900-5

High Throughput Entropy Coding in the HEVC Standard

Jung-Ah Choi · Yo-Sung Ho

Received: 13 June 2013 / Accepted: 21 April 2014 / Published online: 13 May 2014
© Springer Science+Business Media New York 2014

Abstract Context-based adaptive binary arithmetic coding (CABAC) is a single entropy coding mode of the newest video coding standard, High Efficiency Video Coding (HEVC). Despite the high coding efficiency of CABAC, its data dependencies prevent effective parallelization, which also means limited throughput. Accordingly, during the HEVC standardization activity of entropy coding, both coding efficiency and throughput were considered. This paper highlights the key techniques that were proposed and adopted up to HEVC Draft International Standard (DIS). In addition, a new method is proposed for high throughput entropy coding. This method reduces the worst-case complexity of entropy coding without significant coding efficiency loss. From the experimental results, in terms of throughput improvement, we confirm that the proposed method reduces the number of context-coded bins up to 42.1 and 15.9 % on average.

Keywords High Efficiency Video Coding (HEVC) · Parallel-friendly codec · High throughput · Video coding

1 Introduction

ISO/IEC MPEG and ITU-T VCEG have formed a Joint Collaborative Team on Video Coding (JCT-VC) to develop a new international video coding standard, called High Efficiency Video Coding (HEVC) [1]. This standardization has been a response to the growing needs for compressed video representations with substantially increased coding efficiency and enhanced robustness to network environments. Experts have reported that HEVC delivers up to 50 % higher coding efficiency compared to the previous international video coding standard, H.264/AVC [2]. In particular, this efficiency benefits from various new coding tools, including flexible block structures, increased number of intra prediction directions, additional loop filters, and highly adaptive entropy coding.

Entropy coding is a lossless data compression scheme used at the last stage of encoding and the first stage of decoding. HEVC employs context-based adaptive binary arithmetic coding (CABAC) [3] as a single entropy coder. CABAC is one of the most efficient tools in HEVC owing to its context modeling. Context modeling is beneficial for coding efficiency; however, high data dependency problem transpires due to the consistent sequential processing. The serial nature of CABAC causes throughput limits which leads to difficult design of parallel processing. Thus, CABAC is regarded as the main bottleneck regarding growing throughput support for next generation video codecs [4].

In general, data dependencies hinder parallel processing, ultimately degrading throughput. A reasonable trade-off between coding efficiency and throughput is necessary since data dependencies naturally exist due to redundancy removal in coding efficiency improvement [5]. Hence, in the HEVC standardization activity of entropy coding, many tools have been proposed to eliminate data dependencies and critical path delays for parallel processing. Yet, there is still some room for improving entropy coding throughput.

In this paper, we propose a new mechanism of HEVC transform coefficient coding to achieve additional throughput of entropy coding. Based on characteristics of video contents, we selectively apply context-based adaptive variable length coding (CAVLC) transform coefficient coding [6] and the existing CABAC transform coefficient coding. The intention of this approach is to reduce the worst-case complexity of HEVC entropy coding. Unlike H.264/AVC, CAVLC is not supported in HEVC. Thus, we add CAVLC transform

J.-A. Choi (✉) · Y.-S. Ho
Gwangju Institute of Science and Technology (GIST), 123 Cheomdan-gwagiro, Buk-gu, Gwangju 500-712, Republic of Korea
e-mail: jachoi@gist.ac.kr

Y.-S. Ho
e-mail: hoyo@gist.ac.kr

Author's personal copy

coefficient coding as a high throughput binarization mode (HTM) in HEVC. The proposed method simply achieves the balance of coding performance and throughput efficiency; this can be treated as an extension of the HEVC standard.

The paper is organized as follows. In Section 2, we provide an overview of CABAC transform coefficient coding in HEVC. In Section 3, we analyze the parsing issue of throughput bottleneck and describe various techniques for throughput improvement of CABAC transform coefficient coding in the HEVC standard. We propose a high throughput entropy coding engine in Section 4 and evaluate its performance in Section 5. Finally, the paper is concluded in Section 6.

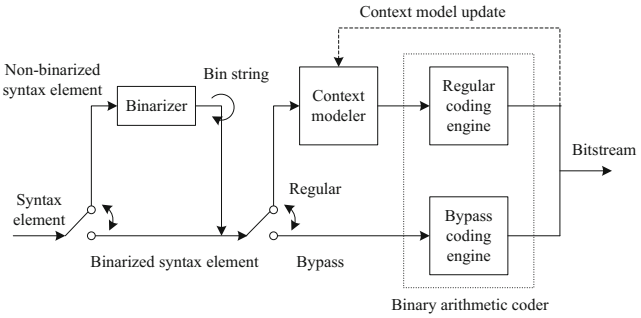

Figure 1 CABAC encoder framework.

2 CABAC Transform Coefficient Coding in HEVC

2.1 The CABAC Framework

HEVC has a single entropy coding method based on the CABAC engine that was also used in H.264/AVC. It is used after the video has been reduced to a series of syntax elements. Figure 1 shows the block diagram of CABAC for a single syntax element. The encoding process of CABAC consists of three coding steps: binarization, context modeling, and binary arithmetic coding.

In the binarization step, a non-binary valued syntax element is mapped to a corresponding binary sequence, called a *bin string*. In HEVC, four basic binarization schemes are used, including unary, truncated unary, *k*th order Exp-Golomb (EGk), and fixed length (FL). The unary code consists of x "1" bits plus a terminating "0" bit for a given unsigned integer x. For truncated unary code, the unary code is used only when $x<cMax$. If $x=cMax$, the terminating "0" bit is neglected such that the truncated unary code of $x=cMax$ is given by a codeword consisting of x "1" bits only. A EGk code is a concatenation of a prefix part with the unary code and a suffix part with Exp-Golomb code. The FL code of x is simply x with a fixed number $l=\lceil \log_2(cMax+1) \rceil$ of bits. Each syntax element uses the selected binarization scheme or the combination of two binarization schemes.

Context modeling is the distinct feature of CABAC entropy coding. It estimates the probability of the bins required to achieve high coding efficiency. The context model for each bin depends on various factors such as the type of the syntax element, bin position within the syntax element, neighboring information, etc. Different context models can be used for different bins and the probability of that context model is updated based on the values of the previously encoded bins. HEVC uses the same probability update method as H.264/AVC.

The principle of arithmetic coding is based on the recursive sub-division of the interval selection. The initial interval is set to [0, 1]. Then, the initial interval is divided into two sub-intervals according to the probability of the bin. Then, one of the two sub-intervals is chosen as the new one based on the encoded bin and the predicted most possible value. As the encoding process goes on, the range is updated to equal the selected sub-interval. Since the range and offset are limited by bit-precision, renormalization is required whenever the range falls below a certain value to prevent underflow.

As shown in Fig. 1, CABAC has two operating modes in the binary arithmetic coding stage: regular mode and bypass mode. Regular mode has a context modeling stage. The engine uses the selected context model to code the binary symbol (*bin*). After each bin is coded, the context model is updated. Bypass mode assumes an uniform probability model. This mode is simpler and allows a coding speedup and easier parallelization, because it does not require context derivation and adaptation.

Due to the large percentage of bins devoted to residual coding, it is important that transform coefficient coding design limits these dependencies to enable high throughput implementations.

2.2 CABAC Transform Coefficient Coding

For the coding of transform coefficient data within the HEVC standard, designed syntax elements are used in CABAC entropy coding mode. Figure 2 illustrates the CABAC encoding scheme for a single transform unit (TU). In Fig. 2, italic letters represent syntax elements for transform coefficient coding. In the following, a more detailed description of each of the major encoding steps of Fig. 2 is given together.

1) Last Significant Coefficient Coordinates: The first step of transform coefficient coding is the coding of the last significant coefficient flag. The position of the last significant coefficient in a TU following the forward scan order is coded. The position of the last significant coefficient in a TU is coded by explicitly signaling its (x, y) coordinates [7]. Here, x and y indicate the column and row positions, respectively.

Author's personal copy

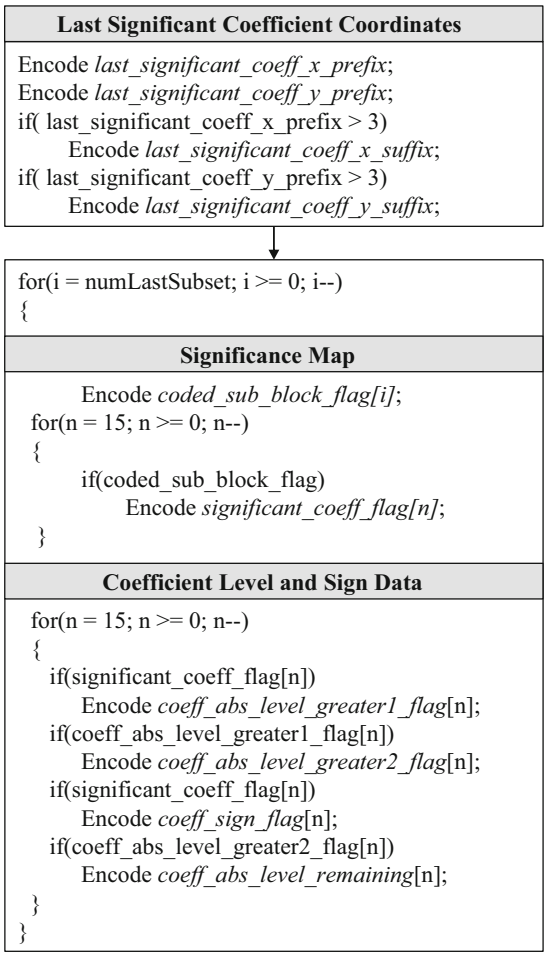

Figure 2 Encoding structure of CABAC transform coefficient coding.

2) Significance Map: The significance map identifies the positions of the non-zero coefficients (significant coefficients) in the TU. The 8×8, 16×16, and 32×32 TUs are divided into 4×4 subsets and each subset corresponds to a sub-block. Syntax elements, coded_sub_block_flag [8] and significant_coeff_flag are coded in a sub-block unit. Since the last coefficient is already known, significance map coding starts at the coefficient before the last coefficient in the scan order. *coded_sub_block_flag* is a one bit symbol, which indicates if there are significant, i.e., non-zero coefficients inside a sub-block of transform coefficients. If *coded_sub_block_flag* is zero, no further information is transmitted for the sub-block. To reduce the number of *coeded_sub_block_flag* bins, it is inferred to be one for sub-blocks containing DC and the last significant coefficient. Figure 3 shows an example of the signaling method of *coded_sub_block_flag* for an 8×8 TU. If the *coded_sub_block_flag* indicates that the sub-block has significant coefficients, a binary-valued *significant_coeff_flag* is encoded. For each coefficient in scanning order, a one-bit symbol *significant_coeff_flag* is transmitted. If the *significant_coeff_flag* symbol is one, it represents a non-zero coefficient exists at this scanning position.
3) *Coefficient Level and Sign Data*: *coeff_abs_level_greater1_flag* and *coeff_abs_level_greater2_flag* indicate whether the coefficient amplitude is larger than one and two, respectively. Accordingly, for levels smaller than three, at most two bins are coded using these flags. In order to improve throughput, these two level flags are not coded for all coefficients in the sub-block. Detail explanation will be given in Section 3.

If above level flags are both equal to one, *coeff_abs_level_remaining*, which specifies the remaining absolute value of the coefficient level, is coded. The syntax element *coeff_abs_level_minus3* is binarized by Golomb-Rice codes with the Rice parameter k [9]. The motivation of Golomb-Rice codes is to reduce the complexity of the unary/kth order Exp-Golomb code in the previous video coding standard, H.264/AVC. The complexity problem of the unary/kth order Exp-Golomb code is caused by the adaptive context modeling process. Since Golomb-Rice codes do not require any context modeling, it efficiently reduces the complexity of encoding and decoding.

Given a particular Rice parameter k, the Golomb-Rice code for a symbol s consists of two parts: a unary representation of p and a binary representation of r. The relation of p and r is shown in Eq. (1). The unary representation is formed by the p '1's, followed by a '0'. Then, the codeword for r is constructed by appending the k least significant bits of r to the binary representation. The length of the Golomb-Rice code is $k+1+p$.

$$p = \left\lfloor \frac{s}{2^k} \right\rfloor \quad where \quad r = s - p \cdot 2^k \tag{1}$$

The range of the Rice parameter k is from 0 to 3. The initial value of k is zero and it monotonically increases according to the magnitude of the level.

For sign information, *coeff_sign_flag* is encoded. Sign bins represent a substantial proportion of a compressed bitstream. Sign bins are signaled before *coeff_abs_level_remaining* bins. To improve coding efficiency, the data hiding method is used such that the sign flag for the first non-zero coefficient is not always

Author's personal copy

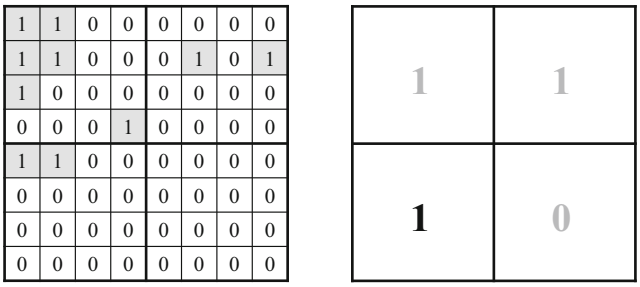

Figure 3 Example of signaling of *coded_sub_block_flag* for an 8×8 TU.

sent [5]. Figure 4 illustrates an example of the level and sign data coding process for a 4×4 sub-block.

Table 1 Last significant coefficient coordinates coding.

	Prefix (Context-coded bins)	Suffix (Bypass bins)
0	0	–
1	10	–
2	110	–
3	1110	–
4–5	11110	X
6–7	111110	X
8–11	1111110	XX
12–15	11111110	XX
16–23	111111110	XXX
24–31	111111111	XXX

3 Throughput Improvement Techniques

During HEVC standardization, there was a lot of effort spent in improving the throughput of CABAC transform coefficient coding with minimal coding loss. The throughput bottleneck is primarily due to the context selection dependencies [10]. Thus, most researches focused on how to reduce context-coded bins. Although CABAC has two coding modes: regular mode and bypass mode, most of the symbols are encoded by regular mode. Throughput of CABAC is limited for context-coded bins due to the data dependencies.

It is easier to process bypass bins in parallel since they do not have the data dependencies related to context selection. In addition, arithmetic coding for bypass bins is simpler as it only requires a right shift versus a table look up for context-coded bins. Thus, the throughput can be improved by reducing the number of context-coded bins and using bypass bins instead.

Chien et al. proposed a last position coding method [11]. The coordinate is binarized in two parts: a prefix and a suffix. The prefix part (*last_significant_coeff_x_prefix* and *last_significant_coeff_y_prefix*) represents an index to an interval. It has a truncated unary representation and the bins are coded in regular mode. The suffix part (*last_significant_coeff_x_suffix* and *last_significant_coeff_y_suffix*) represents the offset within the interval. For certain values of the prefix part, the suffix part is not present and is inferred to be 0. The suffix part has a fixed length representation and is coded in the bypass mode of CABAC. As an example, Table 1 shows the codeword structure for syntax elements of last significant coefficient coordinates for a 32×32 TU. In Table 1, 'X' means 0 or 1 and most significant bit (MSB) is signaled first.

Figure 4 Example of coding 4×4 sub-block.

18	6	-6	-1
-12	4	-4	0
7	4	2	1
2	4	-1	0

4×4 sub-block

16	14	11	7
15	12	8	4
13	9	5	2
10	6	3	1

Coding order

	0	1	-1	0	2	4	-1	-4	4	2	-6	4	7	6	-12	18
significant_coeff_flag	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
coeff_abs_level_greater1_flag		0	0		1	1	0	1	1	1	1	1	1	1	1	1
coeff_abs_level_greater2_flag					0	1		1	1	0	1	1	1	1	1	1
coeff_sign_flag		0	1		0	0	1	1	0	0	1	0	0	0	1	0
coeff_abs_remaining						1		1	1		3	1	4	3	9	15

Author's personal copy

Figure 5 Example of a new method for 4×4 sub-block coding.

18	6	-6	-1
-12	4	-4	0
7	4	2	1
2	4	-1	0

4×4 sub-block

16	14	11	7
15	12	8	4
13	9	5	2
10	6	3	1

Coding order

	0	1	-1	0	2	4	-1	-4	4	2	-6	4	7	6	-12	18
significant_coeff_flag	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
coeff_abs_level_greater1_flag		0	0		1	1	0	1	1	1						
coeff_abs_level_greater2_flag					0											
coeff_sign_flag		0	1		0	0	1	1	0	0	1	0	0	0	1	0
coeff_abs_remaining						2		2	2	0	5	3	6	5	11	17

A demand for a high throughput binarization defined by the number of bins coded in the bypass mode of CABAC relative to the number of bins coded with adaptive context models exists. There have been several proposals to improving entropy coding throughput by reducing context-coded bins for coefficients level coding.

Nguyen et al. proposed a simple coding scheme based on the existing adaptive Rice binarization of CABAC [12]. Instead of using the truncated unary binarization, the proposed scheme starts directly with the Rice binarization. Thus, all bins resulting from the binarization of a transform coefficient level are coded by CABAC bypass mode. However, this scheme showed up to a 9.4 % bit-rate increase and it is not adopted into the final HEVC standard.

Kim et al. proposed a new high throughput binarization method for CABAC [13]. They introduced two binarization methods: VLC table-based binarization and Golomb-Rice code-based binarization. The benefit of this method is that it provides a type of flexibility between coding performance and throughput efficiency by selectively applying the proposed binarization for each 4×4 coefficient block.

Lainema et al. proposed another high throughput binarization method [14]. To increase throughput of entropy coding, they utilize CABAC bypass mode for all transform coefficient data and applies CAVLC coefficient coding engine as HTM. In addition, the proposed method supports full CABAC coding engine, referred to as high efficiency binarization mode. There is a slice level indication identifying which binarization scheme is to be applied. However, these proposed methods reduce coding efficiency up to 9.5 and 11.1 %, respectively.

Chen et al. proposed a new coefficient level coding scheme [15]. They proposed to code *coeff_abs_level_greater1_flag* only for eight starting non-zero coefficients and only one *coeff_abs_level_greater2_flag* in a subset consisting of 16 coefficients, and an early switch to Golomb-Rice bypass mode coding for the remaining coefficients. Among above mentioned techniques, the Chen's method has proved to be effective and adopted in the HEVC standard. Figure 5 illustrates an example of the proposed level and sign data coding process.

Table 2 CAVLC syntax elements for transform coefficient coding.

Syntax element	Description
last_pos_level_one	The position of the last significant coefficient whether the absolute value of the coefficient is larger than one
level_minus2_and_sign	The value of a transform coefficient that has an absolute value that is larger than one
sign_flag	The sign of a non-zero coefficient
run_level_one	The number of consecutive transform coefficients in the scan with zero value before a non-zero coefficients and whether the absolute value of the non-zero coefficient is larger than one
level	The absolute value of the non-zero coefficient

(a) (b) (c)

Figure 6 Sample images. **a** Simple-textured image **b** Medium-textured image **c** Complex-textured image.

Author's personal copy

Table 3 BD-rate comparison between CAVLC and CABAC.

	T1	T2	T3	T4	T5
Complex	+1.66	+2.27	+1.07	+1.18	+1.31
Medium	+0.30	+0.13	+0.42	+0.12	+0.07
Simple	+1.99	+2.17	+2.59	+1.26	+1.68

4 Proposed Hybrid Binarization Technique for High Throughput Entropy Coding

As mentioned in Section 2, on higher bitrates the portion of transform coefficient data has a dominant role in encoded bitstream. Thus, in order to improve the throughput, we propose the codec with HTM for transform coefficient data.

Table 2 shows syntax elements and its detailed description for CAVLC transform coefficient coding [16]. CAVLC is the attractive transform coefficient coding method in terms of the throughput, since CAVLC does not require any context modeling process. Thus, it is generally regarded as the suitable entropy coding method for lower-power devices such as handheld consumer electronics devices. The only problem of CAVLC transform coefficient coding is coding efficiency degradation. It is reported that CAVLC has resulted in approximately 4.9–5.9 % bit-rate increase [17].

We encoded 128×128 pixel sample images to analyze the performance of CAVLC consequences of characteristics of video contents. For comparison, we also consider CABAC encoding results for the same sample images. We encoded 15 various sample images with different textures (complex-textured, medium-textured, and simple-textured images). For measuring the performance, we used the widely known the percentage of bit-rate savings (BD-rate) [18]. BD-rate computes an average of the bit-rate savings over the four QP points (22, 27, 32, and 37). Positive numbers indicate BD-rate losses. Figure 6 shows representatives of sample images and Table 3 shows the simulation results.

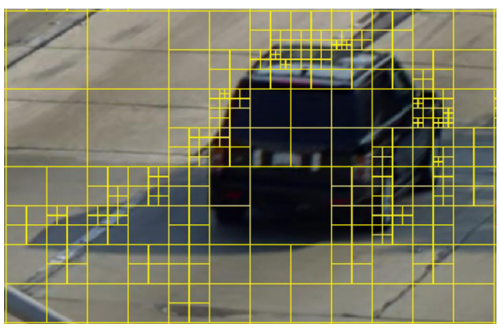

Figure 7 Part of the *Traffic* sequence (2560×1600) showing TU structure from recursive quad-tree partitioning.

```
1:  if(trSize == 8 && PrevNonZero > T8×8){
2:          HTM_codeCoeffNxN();
3:  }
4:  else if(trSize == 16 && PrevNonZero > T16×16){
5:          HTM_codeCoeffNxN();
6:  }
7:  else{
8:          codeCoeffNxN();
9:  }
```

Figure 8 Pseudo-code for the proposed transform coefficient coding method.

CABAC definitely provides better coding efficiency for complex- and simple-textured images. Higher efficiency of CABAC comes from context modeling and significance map coding processes. Using context modeling, symbols that have the same of similar nature are encoded in the same context as they share the same statistical properties. Symbols of different nature are encoded in different contexts. Since CABAC provides an accurate probability estimate, it is efficient in complex regions that have many changes in the statistical characteristics. In addition, by taking advantage of syntax element *coded_sub_block_flag*, CABAC can encode homogeneous region efficiently.

For medium-textured images, the performance degradation of CAVLC is negligible. Hence, we decided to distinguish regions containing complex, medium, and simple textures. For complex- and simple-textured regions, we keep existing CABAC to prevent coding efficiency loss. Otherwise, for medium-textured regions, we use CAVLC to improve the throughput of entropy coding and CAVLC codewords are fed to bypass coding of CABAC. In the proposed method, we regard the current TU size and the number of significant coefficients of the previously encoded TU as factors that describe characteristics of video contents.

While the H.264/AVC encoder [2] divides a picture into fixed size macroblocks of 16×16 pixels, the HEVC encoder divides a picture into coding tree units (CTU) of 16×16, 32×32 or 64×64 pixels [19].

The CTU can be further divided into smaller blocks using a quad-tree structure. Such a block, called a coding unit (CU), can be split further into prediction units (PUs) and is also a root for the transform quad-tree. Each of the child nodes of the transform quad-tree defines a TU. The size of transform and quantization can vary from 4×4 to 32×32 pixels. At the HEVC encoder side, tree-pruning algorithms exist to estimate the optimal partitioning in a rate-distortion sense. An example of TU quad-tree partitioning is given in Fig. 7.

Figure 8 presents the pseudo-code of the proposed method implemented in the HEVC software. As the optimal size of the

Author's personal copy

Table 4 Test sequences.

Sequence	Resolution	Number of frames	Frame rate (Hz)	Bit depth
Traffic	2560×1600	100	30	8
PeopleOnStreet	2560×1600	100	30	8
Kimono	1920×1080	100	24	8
ParkScene	1920×1080	100	24	8
Cactus	1920×1080	100	50	8
Vidyo1	1280×720	100	60	8
Vidyo3	1280×720	100	60	8
Vidyo4	1280×720	100	60	8
BasketballDrill	832×480	100	50	8
BQMall	832×480	100	60	8
RaceHorsesC	832×480	100	30	8
BasketballPass	416×240	100	50	8
BlowingBubbles	416×240	100	50	8

above mentioned blocks typically depends on the picture contents, we can assume that the current TU size (*trSize*) can represent characteristics of the regions. Thus, we can regard the smallest TU (4×4) as the most complex region and the largest TU (32×32) as the simplest region. We determined that other TUs (8×8 and 16×16 TUs) as ambiguous regions.

In order to determine characteristics of ambiguous regions, we additionally use the number of significant coefficients of the previously encoded TU (*PrevNonZero*). If *PrevNonZero* is greater than given threshold values, we activate CAVLC transform coefficient coding as HTM. Thresholds ($T_{8\times8}$ and $T_{16\times16}$) are determined empirically through simulations, considering trade-off between coding efficiency and throughput improvements. In the proposed method, $T_{8\times8}$ and $T_{16\times16}$ are set to 16 and 100, respectively. Since entropy coding mode is intended by *trSize* and *PrevNonZero*, we do not need to signal the identifier indicating which transform coefficient coding mode is being used. In the proposed method, even in HTM, only transform coefficients are binarized by CAVLC and bypass coded through the arithmetic coding machine. Thus, no additional entropy coding machine is required.

5 Experimental Results

In this section, we present the performance results of the proposed method in terms of PSNR, bit rate, and throughput.

Table 5 Saving of average number of context-coded bins of the proposed method with respect to HM 8.0 (Low QP set).

Sequence	QP=1 (%)	QP=5 (%)	QP=9 (%)	QP=13 (%)	Average saving (%)
Traffic	31.3	10.9	11.2	8.9	15.6
PeopleOnStreet	31.5	11.4	11.9	11.4	16.6
Kimono	42.1	14.6	16.1	12.6	21.4
ParkScene	34.1	11.2	13.2	14.0	18.1
Cactus	35.0	11.6	14.2	16.8	19.4
Vidyo1	37.5	12.5	12.5	5.8	17.1
Vidyo3	35.5	12.3	12.5	5.5	16.5
Vidyo4	37.5	12.3	12.0	9.6	17.9
BasketballDrill	19.4	8.2	8.3	8.3	11.1
BQMall	27.1	11.9	14.0	13.6	16.7
RaceHorsesC	28.7	9.9	10.6	9.3	14.6
BasketballPass	29.0	12.3	10.2	7.4	14.7
BlowingBubbles	18.8	7.6	8.7	8.9	11.0
RaceHorses	24.0	8.8	8.4	8.0	12.3
Average	30.8	11.1	11.7	10.0	15.9

Author's personal copy

Table 6 Saving of average number of context-coded bins of the proposed method with respect to HM 8.0 (Common QP set).

Sequence	QP=22 (%)	QP=27 (%)	QP=32 (%)	QP=37 (%)	Average saving (%)
Traffic	4.4	2.6	1.4	0.8	2.3
PeopleOnStreet	5.2	2.8	1.4	0.8	2.6
Kimono	0.9	0.5	0.2	0.1	0.4
ParkScene	7.1	3.9	2.1	0.8	3.5
Cactus	7.3	4.0	2.0	0.8	3.5
Vidyo1	2.4	1.5	0.8	0.5	1.3
Vidyo3	2.5	1.9	1.5	0.6	1.6
Vidyo4	3.2	1.5	1.0	0.6	1.6
BasketballDrill	2.0	1.1	0.8	0.5	1.1
BQMall	3.3	2.3	1.5	0.9	2.0
RaceHorsesC	5.9	6.9	6.5	1.4	5.2
BasketballPass	2.1	1.2	0.9	0.4	1.2
BlowingBubbles	5.1	3.3	1.8	1.1	2.8
RaceHorses	8.9	6.4	2.0	0.5	4.5
Average	4.3	2.9	1.7	0.7	2.4

In order to verify efficiency of the proposed method, we performed experiments on 14 video sequences of YUV420 format. These video sequences were selected from the 24 sequences specified in the common test conditions document from JCT-VC [20]. These sequences differ broadly from one another in terms of frame rate, motion and texture characteristics as well as spatial resolution. Table 4 presents the name, frame count, frame rate, bit depth, and spatial resolution for each video sequence. We implemented our proposed method in the reference software, HEVC Test Model version 8.0 (HM 8.0) [21].

It has been traditionally challenging to identify a reliable method that can be used to analyze the implementation complexity of different entropy codec methods and to do it for software and hardware implementations at the same time. A reliable scheme to analyze the implementation complexity of different entropy codec methods needs to focus on the worst case scenarios [22]. Implementations of the different entropy codec methods will need to be able to deal with those cases and have a design that will be able to perform it correctly in all cases.

Table 7 Saving of maximum number of context-coded bins of the proposed method with respect to HM 8.0 (Low QP set).

Sequence	QP=1 (%)	QP=5 (%)	QP=9 (%)	QP=13 (%)	Average saving (%)
Traffic	30.8	10.7	11.0	8.4	15.2
PeopleOnStreet	31.1	11.3	11.7	11.4	16.4
Kimono	41.8	13.3	15.8	13.0	21.0
ParkScene	32.4	10.8	12.8	13.4	17.4
Cactus	33.2	11.2	13.8	16.1	18.6
Vidyo1	37.9	13.8	17.0	11.2	20.0
Vidyo3	36.5	13.9	16.8	11.2	19.6
Vidyo4	36.3	11.8	12.1	9.5	17.4
BasketballDrill	18.2	7.6	7.6	7.5	10.2
BQMall	24.7	10.8	13.2	11.5	15.1
RaceHorsesC	26.0	9.2	10.1	9.0	13.6
BasketballPass	24.1	9.0	9.9	7.4	12.6
BlowingBubbles	16.8	6.8	8.0	8.8	10.1
RaceHorses	20.1	7.6	7.2	7.1	10.5
Average	29.3	10.6	11.9	10.4	15.5

Author's personal copy

Table 8 Saving of maximum number of context-coded bins of the proposed method with respect to HM 8.0 (Common QP set).

Sequence	QP=22 (%)	QP=27 (%)	QP=32 (%)	QP=37 (%)	Average saving (%)
Traffic	4.2	2.5	1.5	0.8	2.3
PeopleOnStreet	5.1	2.7	1.3	0.8	2.5
Kimono	1.1	0.7	0.2	0.0	0.5
ParkScene	6.8	3.8	2.1	0.7	3.4
Cactus	7.1	3.7	1.7	0.8	3.3
Vidyo1	2.4	1.6	0.8	0.5	1.3
Vidyo3	1.9	2.1	1.6	0.8	1.6
Vidyo4	4.1	1.7	0.8	1.1	1.9
BasketballDrill	1.8	0.9	0.8	0.6	1.0
BQMall	2.6	2.2	1.7	1.1	1.9
RaceHorsesC	4.6	4.6	7.1	1.7	4.5
BasketballPass	2.0	0.9	0.5	0.3	0.9
BlowingBubbles	5.4	3.2	2.3	0.8	2.9
RaceHorses	8.8	4.2	2.3	0.9	4.1
Average	4.1	2.5	1.8	0.8	2.3

In other words, an implementation of an entropy codec method will need to operate satisfactory in all cases, including the most complex ones. Current common conditions and software reference configurations may not be enough to analyze the implementation complexity of different entropy codec methods as they are not representative of the worst case scenario in which the entropy codec will need to operate. Thus, the proposed method is evaluated for the low QP set (QP=1, 5, 9, and 13) as well as the common QP set (QP=22, 27, 32, and 37).

Since the highest bin-rate occurs for all-intra case [23], we tested all-intra coding case to focus on the worst case. For other experimental environments, we follow the JCT-VC common test conditions [20] using HEVC main profile. Then, we compared the proposed method to HEVC transform coefficient coding with respect to context-coded bin usage, maximum context-coded bin usage, and coding efficiency.

The throughput of the proposed method is evaluated by following measures.

$$NC\,Saving(\%) = \frac{NC_{HEVC} - NC_{Proposed\ Method}}{NC_{HEVC}} \times 100 \qquad (2)$$

$$MC\,Saving(\%) = \frac{MC_{HEVC} - MC_{Proposed\ Method}}{MC_{HEVC}} \times 100 \qquad (3)$$

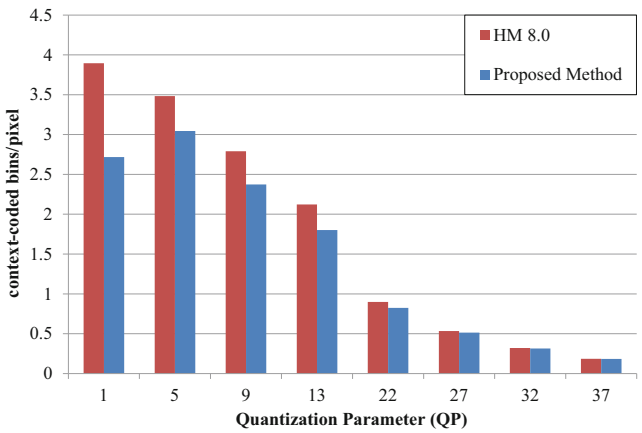

Figure 9 Comparison of the number of context-coded bins per pixel.

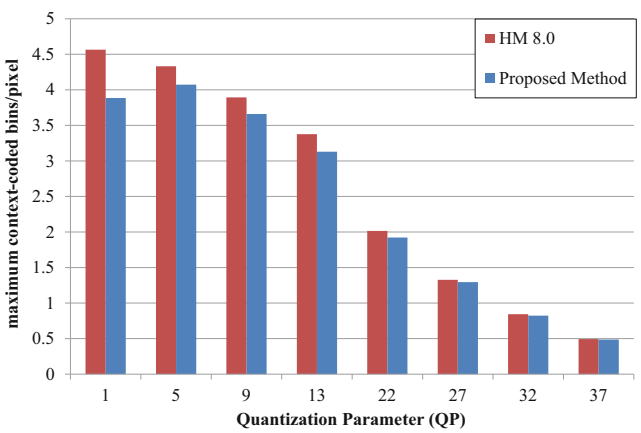

Figure 10 Comparison of the maximum context-coded bins per pixel.

Author's personal copy

Table 9 BD-rate increase.

Sequence	Low QP set	Common QP set
Traffic	1.1	0.2
PeopleOnStreet	1.1	0.3
Kimono	1.7	0.1
ParkScene	1.3	0.3
Cactus	1.5	0.3
Vidyo1	1.6	0.1
Vidyo3	1.5	0.2
Vidyo4	1.6	0.2
BasketballDrill	0.9	0.1
BQMall	1.3	0.3
RaceHorsesC	0.9	0.6
BasketballPass	1.2	0.1
BlowingBubbles	0.7	0.2
RaceHorses	0.6	0.4
Average	1.2	0.3

Here, *NC* is the number of context-coded bins per pixel and *MC* is the maximum context-coded bins per pixel. Tables 5 and 6 use Eq. (2) and show saving of average context-coded bins of the proposed method (*NC Saving*), compared to HM 8.0. Tables 7 and 8 use Eq. (3) and present the saving of maximum context-coded bins of the proposed method (*MC Saving*) with respect to HM 8.0.

On the average, the proposed method reduces the number of context-coded bins by 15.9 % in low QP set and 2.4 % in common QP set. Using proposed method, context-coded bins are dramatically reduced, especially at the high bit-rate where CABAC throughput is more problematic. Note that the proposed method reduced the number of context-coded bins up to 42.1 % at QP=1. The proposed method also reduces the maximum number of context-coded bins.

As mentioned in Section 3, the HEVC standard already employs several throughput improvement techniques and these techniques were incorporated in the HM 8.0 software. It implies that the proposed method achieves additional throughput improvements besides the current HEVC standard.

Figures 9 and 10 show comparison results between the proposed method and HEVC in terms of the number of context-coded bins per pixel and the maximum context-coded bins per pixel. Coding efficiency results are presented by BD-rate [18]. Table 9 shows the coding efficiency of the proposed method with respect to the HEVC main profile anchor. The performance impact of the proposed method is averagely 1.2 % (the low QP set) and 0.3 % (the common QP set) BD-rate losses, compared to HM 8.0. From Table 9, we can verify that the proposed method has no noticeable effect on coding efficiency.

6 Conclusions

In this paper, we proposed a method for high throughput transform coefficient coding in the HEVC standard. Considering texture characteristics of video sequences, we selectively activate the high throughput mode (HTM) based on the traditional CAVLC. The proposed HTM can be treated as an extension of the current CABAC entropy coding. Compared to the HEVC main profile, the proposed method provides context-coded bins saving up to 42.1 % without significant coding efficiency loss.

Acknowledgments This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012–0009228).

References

1. Bross, B., Han, W., Ohm, J., Sullivan, G., & Wiegand, T. (2012). High Efficiency Video Coding (HEVC) text specification draft 8. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-J1003.
2. Sullivan, G., Topiwala, P., & Luthra. A. (2004). The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions. *Proc. of SPIE conference, Special Session on Advances in the New Emerging Standard: H.264/AVC* (pp. 454–474).
3. Marpe, D., Schwarz, H., & Wiegand, T. (2003). Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology (CSVT), 13*(7), 620–636.
4. Sze, V. (2011). Context selection complexity in HEVC CABAC. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-D244.
5. Sze, V., & Budagavi, M. (2012). High throughput CABAC entropy coding in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology (CSVT), 22*(12), 1778–1791.
6. Bjøntegaard, G., & Lillevold, K. (2002). Context-adaptive VLC (CVLC) coding of coefficients. ITU-T SG16 Q.6 and ISO/IEC JTC1/SC29/WG11, Doc. JVT-C028.
7. Sole, J., Joshi, R., & Karczewicz, M. (2011). CE11: Parallel context processing for the significance map in high coding efficiency. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-E338.
8. Nguyen, N., Ji, T., He, D., Martin-Cocher, G., & Song, L. (2011). Multi-level significant maps for large transform units. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-G644.
9. Nguyen, T., Marpe, D., Schwarz, H., & Wiegand, T. (2011). Reduced-complexity entropy coding of transform coefficient levels using truncated Golomb-Rice codes in video compression. *Proc. of IEEE Int. Conf. on Image Processing (ICIP)* (pp. 753–756).

Author's personal copy

10. Sze, V., & Chandrakasan, A. (2012). A highly parallel and scalable CABAC decoder for next generation video coding. *IEEE Journal of Solid-State Circuits, 47*(1), 8–22.
11. Chien, W., Sole, J., & Karczewicz, M. (2011). Last position coding for CABAC. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-G704.
12. Nguyen, T., Marpe, D., Siekmann, M., & Wiegand, T. (2012). Non-CE1: High throughput coding scheme with rice binarization. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-H0458.
13. Kim, S., Misra, K., Kerofsky, L., & Segall, A. (2012). Non-CE1: High Throughput Binarization (HTB) method with modified level coding. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-H0510.
14. Lainema, J., Ugur, K., & Hallapuro, A. (2012). CE1.D1: Nokia report on high throughput binarization. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-H0232.
15. Chen, J., Chien, W., Joshi, R., Sole, J., & Karczewicz, M. (2012). Non-CE1: Throughput improvement on CABAC coefficients level coding. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-H0554.
16. Karczewicz, M., Wang, X., & Chien, W. (2011). CE5: Improved coefficient coding with LCEC. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-D374.
17. Davies, T. (2011). CE1: Subtest 12, entropy coding comparisons with simplified RDO. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-G210.
18. Bjøntegaard, G. (2008). Improvements of the BD-PSNR model. ITU-T SG16 Q.6, Doc. VCEG-AI11.
19. Sullivan, G., Ohm, J., Han, W., & Wiegand, T. (2012). Overview of the High Efficiency Video Coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology (CSVT), 22*(12), 1649–1668.
20. Bossen, F. (2012). HM 8 common test conditions and software reference configurations. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-J1100.
21. High Efficiency Video Coding (HEVC) Reference Software Model (HM 8.0), available in https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-8.0/.
22. Duenas, A., Arora, P., Patino, O., & Roncero, F. (2012). Complexity analysis of high throughput CABAC entropy codecs. ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Doc. JCTVC-H0489.
23. Sole, J., Joshi, R., Nguyen, N., Ji, T., Karczewicz, M., Clare, G., et al. (2012). Transform coefficient coding in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology (CSVT), 22*(12), 1765–1777.

Jung-Ah Choi received her B.S. degree in Electronic engineering and Avionics from Korea Aerospace University, Korea, in 2007 and M.S. degree in Information and Communication Engineering from Gwangju Institute of Science and Technology (GIST), Korea, in 2008. She is currently a Ph.D. student in the School of Information and Communications at GIST, Korea. Her research interests are digital image and video coding, H.265/HEVC, H.264/AVC, rate-distortion optimization, and signal processing.

Yo-Sung Ho received the B.S. and M.S. degrees in electronic engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, Santa Barbara, in 1990. He joined the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea, in 1983. From 1990 to 1993, he was with Philips Laboratories, Briarcliff Manor, NY, where he was involved in development of the advanced digital high-definition television system. In 1993, he rejoined the Technical Staff of ETRI and was involved in development of the Korea direct broadcast satellite digital television and high-definition television systems. Since 1995, he has been with the Gwangju Institute of Science and Technology, Gwangju, Korea, where he is currently a Professor in the Department of Information and Communications. His research interests include digital image and video coding, image analysis and image restoration, advanced coding techniques, digital video and audio broadcasting, 3-D television, and realistic broadcasting.