Signal Processing: *Image Communication* 25 (2010) 687–696

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Efficient entropy coding scheme for H.264/AVC lossless video coding

Seung-Hwan Kim [a], Jin Heo [b], Yo-Sung Ho [b,*]

[a] Ming Hsieh Department of Electrical Engineering and Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089-2564, USA
[b] Department of Information and Communications, Gwangju Institute of Science and Technology (GIST), Gwangju 500-712, Korea

ARTICLE INFO

Article history:
Received 19 December 2009
Accepted 24 April 2010

Keywords:
H.264/AVC
CAVLC
CABAC
Lossless video coding
Entropy coding

ABSTRACT

Context-based adaptive variable length coding (CAVLC) and context-based adaptive binary arithmetic coding (CABAC) are entropy coding methods employed in the H.264/AVC standard. Since these entropy coders are originally designed for encoding residual data, which are zigzag scanned and quantized transform coefficients, they cannot provide adequate coding performance for lossless video coding where residual data are not quantized transform coefficients, but the differential pixel values between the original and predicted pixel values. Therefore, considering the statistical characteristics of residual data in lossless video coding, we newly design each entropy coding method based on the conventional entropy coders in H.264/AVC. From the experimental result, we have verified that the proposed method provides not only positive bit-saving of 8% but also reduced computational complexity compared to the current H.264/AVC lossless coding mode.

1. Introduction

H.264/AVC improves coding performance over previous video coding standards, such as MPEG-2, H.263, and MPEG-4 part 2, by applying more sophisticated coding techniques, such as intra prediction, variable block size motion estimation, rate-distortion optimized mode decision, and entropy coding [1–4].

In order to provide improved functionality for high fidelity video coding including lossless video coding, Joint Video Team (JVT) developed extensions to the original H.264/AVC standard known as the Fidelity Range Extensions (FRExt) [5,6]. When developing the FRExt amendment, it was decided that a more effective means of lossless coding was desirable for the most demanding applications. Therefore, the FRExt included a transform-bypass [7] lossless mode that employs prediction and entropy coding, which were not previously used in the *pulse-code modulation* (PCM) macroblock mode.

In the meantime, a new intra prediction method called sample-wise *differential pulse-code modulation* (DPCM) [8–10] was developed for lossless intra prediction, which considers that a sample immediately neighboring the sample to be predicted is typically a better predictor than a sample in a neighboring block several samples farther away. As a result, sample-wise DPCM was verified to provide better compression performance without major increment of computational complexity and was subsequently adopted as a part of the new draft amendment for the H.264/AVC standard [11].

The H.264/AVC standard employs two entropy coders: context-based adaptive variable length coder (CAVLC) [12], [13] and context-based adaptive binary arithmetic coder (CABAC) [14]. Although CAVLC is supported for all profiles in H.264/AVC, the main target of CAVLC is the baseline profile of which the applications include video-conferencing, wireless communications, and video-telephony. CABAC is supported for the Main profile and High profile of which the applications include videoconferencing, television broadcasting, and video storage.

* Corresponding author.
E-mail addresses: kimseung@usc.edu (S.-H. Kim), jinheo@gist.ac.kr (J. Heo), hoyo@gist.ac.kr (Y.-S. Ho).

0923-5965/$ - see front matter
doi:10.1016/j.image.2010.04.003

Both entropy coding methods were designed to be adapted to the statistical characteristics of residual errors which are quantized transform coefficients. However, in lossless coding, residual errors are the differential pixel values between the original and the predicted pixel values without transform and quantization. Hence, the statistical characteristics of residual data from lossy and lossless video coding are quite different. Thus, the conventional entropy coding methods in H.264/AVC cannot provide the best coding performance for lossless video coding. Therefore, in this paper, we propose the improved entropy methods for lossless video coding by modifying the conventional entropy coders in H.264/AVC.

Fig. 1 shows the syntax elements employed in both CAVLC and CABAC for a macroblock (MB); here, the gray shaded syntax elements are employed to encode residual data in the MB [15]. In order to reflect the statistical characteristics of residual data, we modified the coding scheme for the corresponding syntax elements. Note that our research goal is to develop the entropy coding methods, which can be easily applied to H.264/AVC lossless video coding by modifying some semantics and decoding processes, without adding any other syntax elements to the H.264/AVC standard.

The rest of this paper is organized as follows. In the next Section, we will briefly review the coding structure of CAVLC and CABAC for residual data. In Section 3, we will introduce an improved CAVLC and CABAC scheme for lossless video coding. In Section 4, coding performance of the proposed entropy coding schemes will be shown and the paper will be completed with our conclusions presented in Section 5.

CAVLC	CABAC
Macroblock Header	
coeff_token	*coded_block_flag*
trailing_ones_sign_flag	*significant_coeff_flag*
level_prefix	*last_significant_coeff_flag*
level_suffix	*coeff_abs_level_minus1*
total_zeros	*coeff_sign_flag*
run_before	-

Fig. 1. Syntax elements for a macroblock.

2. Overview of entropy coding methods in H.264/AVC

In this section, we review the basic coding structure of conventional CAVLC and CABAC in H.264/AVC. These entropy coders are employed to encode residual data, which are zigzag scanned and quantized transform coefficients for the 4×4 sub-block. Fig. 2 illustrates the zigzag scan order for the sub-block.

2.1. Overview of CAVLC

CAVLC was originally designed to take advantage of several characteristics of residual data in lossy coding: (1) after transform and quantization, sub-blocks typically contain many zeros, especially in high frequency regions; (2) the level of the highest non-zero coefficients tends to be as small as one; and (3) the level of non-zero coefficients tends to be larger toward the low frequency regions. Then, taking into consideration the above characteristics, CAVLC employs several syntax elements such as *coeff_token*, *trailing_ones_sign_flag*, *level_prefix*, *level_suffix*, *total_zeros*, and *run_before* to encode residual data efficiently.

The syntax element *coeff_token* encodes both the number of non-zero coefficients (*numcoeff*) and the number of trailing ones (*numtrailingones*) in each sub-block. A trailing one is one of up to three consecutive non-zero coefficients having an absolute value equal to 1 at the end of a scan. If there are more than three trailing ones, only the last three ±1 coefficients are treated as trailing ones, with any others being coded as normal coefficients in the level coding stage.

The four VLC tables used for encoding *coeff_token* consist of three variable-length code tables (*Num-VLC0*, *Num-VLC1*, and *Num-VLC2*) and one fixed-length code

3	7	-1	-2
9	7	2	0
8	-3	-5	-1
2	-2	1	0

Residual data in the sub-block

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

Zigzag scan order for the sub-block

Scanning Position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Coefficient Level	Absolute Value	3	7	9	8	7	1	2	2	3	2	2	5	0	1	1	0
	Sign	+	+	+	+	+	-	-	+	-	+	-	-		-	+	

Reordered residual data according to scan order

Fig. 2. Zigzag scan order for the sub-block.

table (*FLC*). Selection of the VLC table depends on the predicted number of non-zero coefficients (*N*) in the previously coded upper and left sub-blocks as listed in Table 1.

The syntax element, *trailing_ones_sign_flag*, indicates the sign information of each trailing one; sign information is simply encoded by a one bit codeword in reverse order. If sign information is positive (+), *trailing_ones_sign_flag* is equal to zero. Conversely, if sign information is negative (−), *trailing_ones_sign_flag* is equal to one.

The level (sign and magnitude) of each remaining non-zero coefficient in the sub-block is encoded in reverse order, starting from the highest frequency and working back toward the DC coefficient. Each absolute level value is encoded by a selected *Lev-VLC* table from seven *Lev-VLC* tables, with selection of the *Lev-VLC* table based on the magnitude of each recently encoded level. The sign information is encoded in the same way as in Step 2. Choice of the *Lev-VLC* table is adapted as follows:

1) If (numcoeff $>$ 10 && numtrailingones==3)
   Initialize *Lev-VLC1*.
   Else,
   Initialize *Lev-VLC0*.
2) Encode the last scanned absolute level.
3) Encode the sign of the non-zero coefficient.
4) If the magnitude of the current encoded coefficient is larger than a predefined threshold value in Table 2, increment *Lev–VLC* table.

After the encoding process for level information, we should encode the total number of zeros and the position of each zero in the sub-block. For this reason, CAVLC employs two syntax elements, *total_zeros* and *run_before*. The syntax element, *total_zeros*, indicates the total number of zero coefficients located before the last non-zero coefficient. After encoding *total_zeros*, the position of each zero coefficient is encoded. The syntax element, *run_before*, indicates the number of consecutive zero coefficients between the non-zero coefficients and is encoded in reverse order. Note that *zerosleft* indicates the number of zeros that have not yet been encoded. The syntax element *run_before* is encoded using the VLC table which is chosen depending on zerosleft and *run_before*, starting with the highest frequency.

Table 1
Choice of VLC table.

N	VLC table
0, 1	*Num-VLC0*
2, 3	*Num-VLC1*
4, 5, 6, 7	*Num-VLC2*
8 or above	*FLC*

Table 2
Threshold values for '*Lev-VLC*' table.

VLC table for level coding	Threshold value
Lev-VLC0	0
Lev-VLC1	3
Lev-VLC2	6
Lev-VLC3	12
Lev-VLC4	24
Lev-VLC5	48
Lev-VLC6	$>$48

2.2. Overview of CABAC

CABAC consists of three main coding procedures: (1) selecting probability models for each syntax element according to the context of element; (2) adapting probability estimates based on local statistical characteristics; and (3) using arithmetic coding rather than variable-length coding. Considering these properties, CABAC employs syntax elements such as *coded_block_flag*, *significant_coeff_flag*, *last_significant_coeff_flag*, *coeff_abs_level_minus1*, and *coeff_ sign_ flag* for residual data in a sub-block. The encoding structure of CABAC for the sub-block using the given syntax elements is represented in Fig. 3.

For the sub-block, one bit symbol called *coded_block_flag* is transmitted, indicating the existence of coefficients in the current sub-block. If *coded_block_flag* indicates no coefficients, processing for the current sub-block can be stopped here. If *coded_block_flag* indicates the existence of coefficients, significance map and level information are encoded sequentially.

The significance map indicates the location (scanning position) of significant (non-zero) coefficients. In significance map coding, a binary symbol for each coefficient is transmitted along the scanning position, indicating

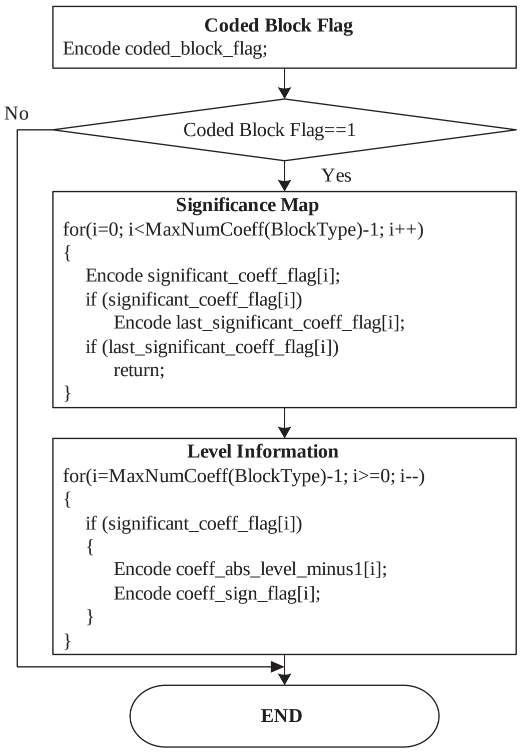

Fig. 3. Encoding structure of CABAC for residual coding.

whether the coefficient at the current position is significant or not. If this is the case, an additional one bit symbol is sent, indicating whether the current coefficient is the last significant coefficient or not in the scan. Each position in the scan is associated with a separate probability model for both the significant map and the last significant coefficient symbol.

After encoding the significance map, the levels at each significant scan position are encoded along the reverse scanning direction. They are represented by two symbols; the absolute value and the sign information. The absolute value which is subtracted by one is coded because zero coefficients are already encoded in the significance map coding. The sign is encoded using the bypass coding mode of the arithmetic coding engine.

For a successful application of context modeling and adaptive arithmetic coding, CABAC adopts binarization scheme to convert non-binary syntax element to the unique intermediate binary codeword for a given syntax element. Hence, the non-binary absolute values are converted into the binary string by so-called *unary/0-th order Exp-Golomb* (UEG0) binarization with cut-off length $S=14$. UEG0 is a binarization method concatenating truncated unary (TU) code for prefix and *0*th *order Exp-Golomb* (EG0) code for suffix. After binarization, the probability distribution of each binary symbol is estimated by its own specified context modeling and encoded arithmetically into bitstream.

2.3. *Analysis of the statistical characteristics of residual data in lossless coding*

In lossy coding, residual errors are quantized transform coefficients. Hence, the probability distribution of non-zero coefficients is likely to decrease as the scanning position increases. Moreover, the absolute value of a non-zero coefficient tends to decrease as the scanning position increases. Hence, when it comes to CAVLC, the occurrence probability of a trailing one is relatively high.

In lossless coding, residual errors are not quantized transform coefficients, but the differential pixel values between the original and predicted pixel values. The statistical characteristics of residual errors in lossless coding are as follows. First, the probability distribution of non-zero coefficients is independent of the scanning position and the number of non-zero coefficients is generally large compared to those in lossy coding. Second, the absolute value of a non-zero coefficient does not decrease as the scanning position increases and it is independent of the scanning position. Finally, the occurrence probability of a trailing one is not so high. Therefore, the trailing one does not need to be treated as a special case of encoding in CAVLC scheme.

In Fig. 4, we show the probability distribution of non-zero coefficients according to the scanning position. As mentioned earlier, a significant difference can be seen in the statistical characteristics between residual data of lossy and lossless coding. We also represent the statistical characteristics of absolute level value (*abs_level*), which depends on quantization parameter (QP) in Fig. 5.

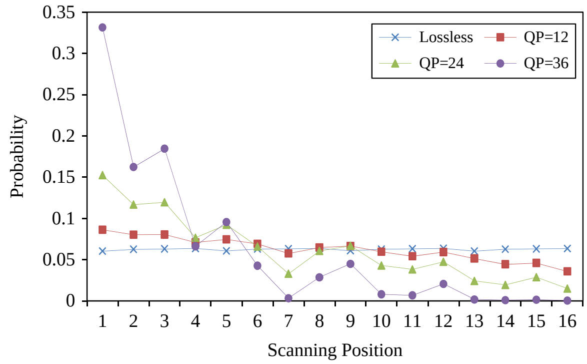

Fig. 4. Probability of non-zero coefficients ('Foreman').

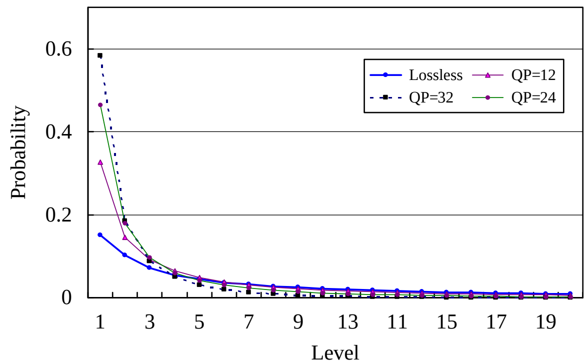

Fig. 5. Statistics of *abs_level* depends on QP ('Foreman').

Table 3 represents the occurrence probability distribution of trailing ones according to QP. In lossless coding, the occurrence probability of trailing ones turns out to be relatively lower than that of lossy coding.

Therefore, in order to more accurately reflect the above mentioned statistical characteristics of residual data, we propose more efficient entropy coding schemes for lossless video coding by modifying the conventional CAVLC and CABAC schemes.

3. Proposed entropy coding scheme in H.264/AVC

In this section, by considering statistical differences in residual data between lossy and lossless coding, we introduce an improved CAVLC and CABAC scheme for lossless video coding, respectively.

3.1. *Proposed CAVLC scheme in H.264/AVC*

Based on Section II-C, we propose a new CAVLC scheme. The coding procedure of the proposed CAVLC can be summarized by the following steps:

Step 1: Encode the total number of non-zero coefficients.
Step 2: Encode the level of all non-zero coefficients.
Step 3: Encode the number of all zeros before the last non-zero coefficient.

Step 4: Encode the number of consecutive zeros preceding each non-zero coefficient.

3.1.1. *Coding the number of non-zero coefficients*

First, we encode the number of non-zero coefficients (*numcoeff*) where we do not consider the number of trailing ones (*numtrailingones*) because the occurrence probability of trailing one turns out to be low as shown in Table 3. In the conventional CAVLC scheme, the corresponding VLC table is selected based on the predicted *numcoeff*. If the predicted numcoeff is larger than seven, the *FLC* (fixed length code) table is selected. Especially, in lossless coding, the *FLC* table is most frequently selected. From extensive experiments on various test sequences, we observed that the probability of the selection was about 95%. Hence, we determined to remove three VLC tables (*Num-VLC0*, *Num-VLC1*, and *Num-VLC2*). Thus, we do not need to consider the process for predicting *numcoeff*.

The *FLC* table assigns fixed four-bit codewords for *numcoeff* and fixed two-bit codewords for *numtrailingones*, respectively. Since we do not consider the syntax element *numtrailingones*, only *numcoeff* is considered. Hence, instead of the *FLC* table, which assigns fixed four-bit codewords for all *numcoeffs*, we newly designed a simple but effective VLC table for lossless coding.

Fig. 6 shows the cumulative probability distribution of the number of non-zero coefficients in the sub-block. A significant difference can be seen in the statistical characteristics of the number of non-zero coefficients between lossy and lossless coding. In lossless coding, the probability of the number of non-zero coefficients turns out to be very low when the number of non-zero coefficients is small. However, the probability of the number of non-zero coefficients drastically increases as the number of non-zero coefficients increases, especially the number of non-zero coefficients from 13 to 16.

In our proposed VLC table, first, we assign four-bit and two-bit codewords to numcoeff from 1 to 12 and 13 to 16, respectively. In order to enhance the coding performance, we assign the different codewords to numcoeff from 1 to 12 according to the statistics of numcoeff instead of assigning four-bit codewords uniformly. Thus, we use the phased-in code [16] which is a slight extension of fixed length code (FLC). The phased-in code consists of codewords with two different lengths. Therefore, we assign four-bit and three-bit codewords to numcoeff from 1 to 9 and 10 to 12, respectively. In order to avoid ambiguity at the decoder, we inserted a check bit into the prefix of each codeword; details regarding the codewords are further described in Table 4.

Table 3
Occurrence probability distribution of trailing ones.

QP sequence	Lossless	12	24	36
News	0.37191	0.81606	0.88247	0.94582
Container	0.33727	0.79991	0.90053	0.94346
Foreman	0.25977	0.79466	0.91170	0.95854
Silent	0.22807	0.84511	0.92457	0.95595
Paris	0.27110	0.78710	0.87534	0.93326
Mobile	0.21069	0.69623	0.85662	0.92677
Tempete	0.22740	0.78424	0.88612	0.94493

3.1.2. *Level coding*

In level coding, the absolute level value of each non-zero coefficient (*abs_level*) is adaptively encoded by a selected *Lev-VLC* table in reverse scanning order. As previously mentioned, selection of the VLC table for level coding is based on the expectation that *abs_level* is likely to increase at low frequencies. Hence, selection of the VLC table is monotonically increased according to the previously encoded *abs_level*. However, *abs_level* in lossless coding is independent of the scanning position, as shown in Fig. 7. Thus, we designed an adaptive method for *Lev-VLC* table selection that can decrease or increase according to the previously encoded *abs_level*.

CAVLC typically determines the smallest *Lev-VLC* table in a range of possible *Lev-VLC* tables based on the assumption that the next *abs_level* is likely to be larger than the current *abs_level*. However, in lossless coding, the

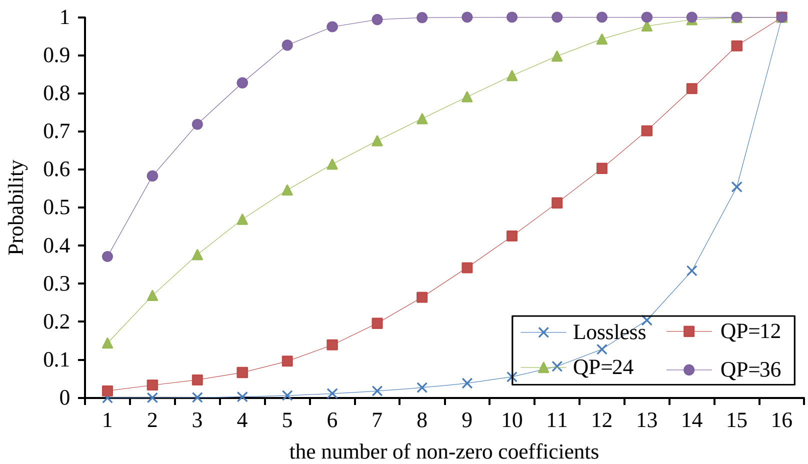

Fig. 6. Cumulative probability distribution of non-zero distribution of average absolute level value ('Tempete').

Table 4
Codeword table for 'numcoeff'.

numcoeff	Codeword		
	Check bit	Bits for *numcoeff*	Codeword length
0	1	1111	5
1	1	1110	5
2	1	1101	5
3	1	1100	5
4	1	1011	5
5	1	1010	5
6	1	1001	5
7	1	1000	5
8	1	0111	5
9	1	0110	5
10	1	010	4
11	1	001	4
12	1	000	4
13	0	00	3
14	0	01	3
15	0	10	3
16	0	11	3

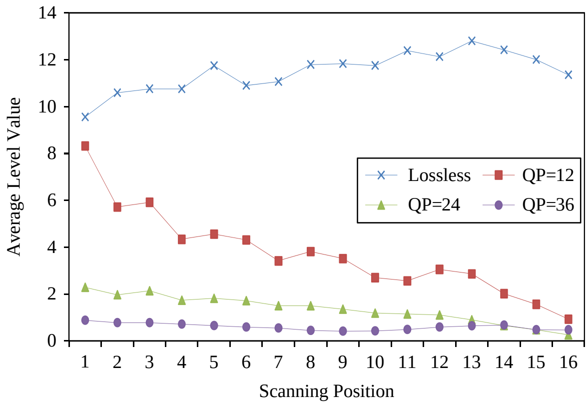

Fig. 7. Distribution of average absolute level value ('Tempete').

next *abs_level* does not necessarily increase at lower frequencies. Hence, we cannot assume that the next *abs_level* is larger than the current *abs_level*. Therefore, *Lev-VLC* table for each *abs_level* should be selected by considering the previously encoded *abs_levels*.

In order to determine an appropriate *Lev-VLC* table, we assign the weighing factors to the previously encoded *abs_levels*. The basic idea is that *abs_level* can be approximated as a weighting combination of the previously encoded *abs_levels*. The estimation procedure of *abs_level* is as follows. If the current *abs_level* position is larger than *lastcoeff*–2, *abs_level* is calculated by an average method. Here, *lastcoeff* means the position of the last scanned *abs_level*. Otherwise, *abs_level* is calculated by using weighting factors. We assign the weighting values to the average value of the previously encoded all *abs_levels* and current *abs_level* by factors of 2/3 and 1/3, respectively. In Table 5, we represent the *Lev-VLC* table for level coding according to the prediction value of *abs_level*.

In Fig. 7, we can observe that the last scanned *abs_level* is quite different between lossy and lossless coding. In level coding, encoding starts with *Lev-VLC0* or *Lev-VLC1* because the last scanned *abs_level* represents the highest frequency coefficient in lossy coding, and it tends to be small; however, in lossless coding the last scanned *abs_level* is not small enough to use *Lev-VLC0* or *Lev-VLC1*.

Table 5
Prediction value of '*abs_level*' for '*Lev-VLC*' table.

VLC table for level coding	Prediction value of *abs_level*
Lev-VLC0	0
Lev-VLC1	2
Lev-VLC2	4
Lev-VLC3	9
Lev-VLC4	19
Lev-VLC5	39
Lev-VLC6	>39

We have observed that the average value of the last scanned *abs_level* in the sub-block is approximately 10.70 in lossless coding. Based on this value, we accordingly adjusted the initial *Lev-VLC* table for level coding in lossless coding. The modified *Lev-VLC* table selection method is as follows:

1) Level coding starts with *Lev-VLC4*.
2) Encode the last scanned *abs_level*.
3) Encode the sign of the non-zero coefficient.
4) Update *Lev-VLC* table by considering previously encoded *abs_levels*.

3.2. *Proposed CABAC scheme in H.264/AVC*

For each block, we do not use the syntax element, *coded_block_flag*, because the probability that all coefficients in a sub-block become zero is very low, which is about 0.1% in lossless coding. Hence, instead of sending *coded_block_flag*, we encode all-zero values in the significance map coding part. We also modified the coded block flag coding, significance map coding, and level information coding.

3.2.1. *Significance map coding*

The significance map indicating the location of a significant coefficient is encoded by sending two syntax elements such as *significant_coeff_flag* and *last_significant_coeff_flag*. In lossless coding, the probability distribution of the existence of a significant coefficient is uniform according to the scanning position because residual errors are differential pixel values between the original and the predicted pixel values without transform and quantization. Hence, significance map coding is likely to be terminated at the end of the scanning position. Therefore, instead of using *significant_coeff_flag* and *last_significant_coeff_flag* together, we directly encode *significant_coeff_flag* for all the scanning positions.

3.2.2. *Level coding*

For level coding, UEG0 binarization with the cut-off value $S=14$ was determined experimentally for the binarization process [14]. In our research, as shown in Fig. 5, we have found that the statistical characteristics of *abs_level* in lossless coding are quite different from those in lossy coding. For easy compatibility with the existing binarization process, we derive the same UEG0 binarization but with different parameter to provide a better fit to the actual statistical characteristics of *abs_level*. Hence, we change the cut-off values for the unary prefix code because the statistics of *abs_level* can be simply controlled by the cut-off value [14].

Theoretically, the optimal cut-off value for the unary prefix code should be designed adaptively according to the statistical characteristics of *abs_level*. However, the adaptive binarization scheme requires much computational complexity and it is also far from our research goal to design an efficient and easily compatible with the H.264/AVC standard. In this research, we found that the overall coding performance has been improved as increase the cut-off value by $2^{BitDepth}$, which is maximal cut-off value because abs_level in lossless mode is absolute magnitude of temporally or spatially predicted residuals. Therefore, we adopted UEG0 binarization with a cut-off value of $2^{BitDepth}$ for the binarization process.

3.2.3. Simplified context modeling

The entity of probability models used in CABAC can be arranged in a linear fashion such that each model can be identified by the unique so-called context index. Hence, the context index (γ) for a syntax element S (*significant_coeff_flag* and *coefficient_abs_level_minus1*) for residual data is given by

$$\gamma = \Gamma_S + \Delta_S(ctx\_cat) + \chi_S \quad (1)$$

where Γ_S denotes the context index offset defined as the lower value of the context range of a syntax element S, $\Delta_S(ctx\_cat)$ denotes the context category dependent offset which depends on the block type, and χ_S denotes the context index increment of a given syntax element S. The context index offset and the context category dependent offset are determined by the corresponding syntax element and the block type, respectively. In Table 6, we represent the block types with the associated context categories [14].

The context index increment,χ_S, is designed to adapt to the statistical characteristics of residual errors which are quantized transform coefficients. Specifically, the context index increment is employed for the syntax elements, such as *significant_coeff_flag, last_significant_coeff_flag*, and *coeff_abs_level_minus1* to reflect the statistical difference according to each scanning position.

Table 6
Block types with the associated context categories.

Block type	$\Delta_S(ctx\_cat)$
Luma DC block for Intra 16 × 16	0
Luma AC block for Intra 16 × 16	1
Luma block for Intra 4 × 4	2
U-Chroma DC block for Intra	3
V-Chroma DC block for Intra	
U-Chroma AC block for Intra	4
V-Chroma AC block for Intra	
Luma block for Intra 8 × 8	5

In lossless coding, the statistics of residual samples do not follow the statistics of quantized transform coefficients in lossy coding. Hence, we do not apply the context increment for the syntax elements; *significant_coeff_flag* and *coeff_abs_level_minus1*. Therefore, the unique context index (γ) for the syntax element S from residual data is simply given by

$$\gamma = \Gamma_S + \Delta_S(ctx\_cat) \quad (2)$$

The context index offset (Γ_S) and the context category dependent offset (Δ_S) are determined according to the corresponding syntax element and the block type, respectively (Table 7).

We would like to discuss scanning patterns for lossless coding. In lossy coding, the coding performance highly depends on various scanning patterns because the residual data is quantized transform coefficients and the statistical distribution of residual data is highly skewed on small level values as depicted in Figs. 4–6. Hence, if we find a proper scanning pattern, we can enhance the coding performance by arranging residual data according to their amplitude levels. However, in lossless coding, the amplitude distribution of the residual signal is quite wide and also shown to be independent of the scanning position as depicted in Figs. 4–6. Therefore, theoretically, there is no scanning order, which can provide even better coding performance and we have also confirmed the fact by performing extensive experiments by using various scanning patterns including zigzag scanning order. As a result, in the proposed lossless coding method including our recent research works [17, 18], we have used the raster scanning pattern, instead of the zigzag scanning pattern.

4. Experimental results and analysis

In this paper, the improved entropy coding schemes for lossless video coding have been presented. To verify the efficiency of the proposed methods, experiments were performed on various test sequences with QCIF, CIF, 4CIF, and HD resolutions. We implemented our proposed method in the H.264 reference software [19]. Table 8

Table 7
Context index for syntax elements between CABAC and the proposed method.

Syntax element (S)	CABAC	Proposed
coded_block_flag	$\Gamma_S+\chi_S$	–
significant_coeff_flag	$\Gamma_S+\Delta_S(ctx\_cat)+\chi_S$	$\Gamma_S+\Delta_S(ctx\_cat)$
last_significant_coeff_flag	$\Gamma_S+\Delta_S(ctx\_cat)+\chi_S$	–
coeff_abs_level_minus1	$\Gamma_S+\Delta_S(ctx\_cat)+\chi_S$	$\Gamma_S+\Delta_S(ctx\_cat)$

Table 8
Encoding parameters.

Parameter	CAVLC	CABAC
ProfileIDC	244 (High 4:4:4)	
QPISlice	0 (lossless)	
QPPrimeYZeroTransformBypassFlag	1	
SymbolMode	0 (CAVLC)	1 (CABAC)

shows the encoding parameters for the reference software.

Note that both the proposed CAVLC and CABAC schemes were applied to H.264/AVC lossless video coding by modifying the semantics and decoding processes, without requiring any syntax elements be added to the H.264/AVC standard. In the first experiment, we compared coding performance of CAVLC and our proposed CAVLC scheme and then compared coding performance of CABAC and our proposed CABAC scheme. In the second experiment, we compared well-known lossless coding techniques, lossless joint photographic experts group (JPEG-LS) [20, 21] with our proposed methods. In the second experiment, we encoded only one frame under the lossless coding mode. Each comparison was made in terms of compression ratio differences and the percentage rate differences with respect to each H.264 entropy coding method. The changes are calculated using

$$\textit{Compression ratio} = \frac{\textit{Original image size}}{\textit{Bitrate}_{\textit{proposed}}}, \tag{3}$$

$$\Delta \textit{Saving bits}(\%) = \frac{\textit{Bitrate}_{\textit{original}} - \textit{Bitrate}_{\textit{proposed}}}{\textit{Bitrate}_{\textit{original}}} 100. \tag{4}$$

From Tables 9–12, we represent the experimental results for the proposed CAVLC and CABAC schemes. From the experiments, we can confirm that the proposed CAVLC and CABAC schemes provide better coding performance compared to the conventional CAVLC and CABAC schemes—by approximately 4% and 13% in lossless video coding, respectively. In addition, the proposed

Table 9
Comparison of the compression ratio for H.264 intra lossless coding with CAVLC.

Image	Method	Total coding bits	Compression ratio	Saving bits (%)
Foreman (QCIF, 300 frames)	H.264/AVC (CAVLC)	41638656	2.1912	0
	Proposed CAVLC	38306832	2.3818	8.002
Mobile (QCIF, 150 frames)	H.264/AVC (CAVLC)	29793808	1.5312	0
	Proposed CAVLC	26613808	1.7142	10.673
Foreman (CIF, 300 frames)	H.264/AVC (CAVLC)	155829888	2.3420	0
	Proposed CAVLC	141531684	2.5786	9.176
Mobile (CIF, 300 frames)	H.264/AVC (CAVLC)	224186128	1.6279	0
	Proposed CAVLC	193960504	1.8816	13.482
City (4CIF, 100 frames)	H.264/AVC (CAVLC)	224780488	2.1648	0
	Proposed CAVLC	213264144	2.2817	5.123
Harbour (4CIF, 100 frames)	H.264/AVC (CAVLC)	222122974	2.1907	0
	Proposed CAVLC	209553766	2.3221	5.659
Blue sky (HD, 100 frames)	H.264/AVC (CAVLC)	1022863486	2.4327	0
	Proposed CAVLC	928062062	2.6812	9.268
Sunflower (HD, 100 frames)	H.264/AVC (CAVLC)	955722846	2.6036	0
	Proposed CAVLC	866436854	2.8719	9.342
Average	H.264/AVC (CAVLC)		**2.1355**	**0**
	Proposed CAVLC		**2.3391**	**8.841**

Table 10
Comparison of the compression ratio for H.264 inter lossless coding with CAVLC.

Image	Method	Total coding bits	Compression ratio	Saving bits (%)
Foreman (QCIF, 300 frames)	H.264/AVC (CAVLC)	29239328	3.1204	0
	Proposed CAVLC	28131348	3.2433	3.789
Mobile (QCIF, 150 frames)	H.264/AVC (CAVLC)	23024882	1.9813	0
	Proposed CAVLC	21406410	2.1311	7.029
Foreman (CIF, 300 frames)	H.264/AVC (CAVLC)	118599246	3.0772	0
	Proposed CAVLC	114509616	3.1871	3.448
Mobile (CIF, 300 frames)	H.264/AVC (CAVLC)	165077620	2.2108	0
	Proposed CAVLC	154641356	2.3600	6.322
City (4CIF, 100 frames)	H.264/AVC (CAVLC)	209031660	2.3279	0
	Proposed CAVLC	199681890	2.4369	4.473
Harbour (4CIF, 100 frames)	H.264/AVC (CAVLC)	216653962	2.2460	0
	Proposed CAVLC	209833894	2.3190	3.148
Blue sky (HD, 100 frames)	H.264/AVC (CAVLC)	886122290	2.8081	0
	Proposed CAVLC	855445544	2.9088	3.462
Sunflower (HD, 100 frames)	H.264/AVC (CAVLC)	854710954	2.9113	0
	Proposed CAVLC	835062756	2.9798	2.299
Average	H.264/AVC (CAVLC)		**2.5854**	**0**
	Proposed CAVLC		**2.6958**	**4.246**

Table 11
Comparison of the compression ratio for H.264 intra lossless coding with CABAC.

Image	Method	Total coding bits	Compression ratio	Saving bits (%)
Foreman (QCIF, 300 frames)	H.264/AVC (CABAC)	37832288	2.4116	0
	Proposed CABAC	35679256	2.5572	5.691
Mobile (QCIF, 150 frames)	H.264/AVC (CABAC)	27472744	1.6605	0
	Proposed CABAC	24873016	1.8340	9.462
Foreman (CIF, 300 frames)	H.264/AVC (CABAC)	142043336	2.5693	0
	Proposed CABAC	135389848	2.6955	4.684
Mobile (CIF, 300 frames)	H.264/AVC (CABAC)	205852480	1.7728	0
	Proposed CABAC	186415008	1.9577	9.442
City (4CIF, 100 frames)	H.264/AVC (CABAC)	213746712	2.2765	0
	Proposed CABAC	204837880	2.3755	4.167
Harbour (4CIF, 100 frames)	H.264/AVC (CABAC)	219859704	2.2132	0
	Proposed CABAC	211104336	2.3050	3.982
Blue sky (HD, 100 frames)	H.264/AVC (CABAC)	745322792	3.3385	0
	Proposed CABAC	707020896	3.5194	5.138
Sunflower (HD, 100 frames)	H.264/AVC (CABAC)	667196856	3.7295	0
	Proposed CABAC	645106224	3.8572	3.310
Average	H.264/AVC (CABAC)		**2.4964**	**0**
	Proposed CABAC		**2.6376**	**5.734**

Table 12
Comparison of the compression ratio for H.264 inter lossless coding with CABAC.

Image	Method	Total coding bits	Compression ratio	Saving bits (%)
Foreman (QCIF, 300 frames)	H.264/AVC (CABAC)	27725464	3.2907	0
	Proposed CABAC	24120456	3.7826	13.002
Mobile (QCIF, 150 frames)	H.264/AVC (CABAC)	20781128	2.1952	0
	Proposed CABAC	16326120	2.7942	21.437
Foreman (CIF, 300 frames)	H.264/AVC (CABAC)	116169784	3.1415	0
	Proposed CABAC	101915112	3.5809	12.270
Mobile (CIF, 300 frames)	H.264/AVC (CABAC)	158166128	2.3074	0
	Proposed CABAC	125669728	2.9040	20.545
City (4CIF, 100 frames)	H.264/AVC (CABAC)	197330064	2.4659	0
	Proposed CABAC	167682560	2.9019	15.024
Harbour (4CIF, 100 frames)	H.264/AVC (CABAC)	210637424	2.3101	0
	Proposed CABAC	187018888	2.6019	11.212
Blue sky (HD, 100 frames)	H.264/AVC (CABAC)	642879600	3.8705	0
	Proposed CABAC	582279224	4.2734	9.426
Sunflower (HD, 100 frames)	H.264/AVC (CABAC)	626148256	3.9740	0
	Proposed CABAC	585767904	4.2479	6.449
Average	H.264/AVC (CABAC)		**2.9444**	**0**
	Proposed CABAC		**3.3858**	**13.670**

Table 13
Comparison of compression ratio for JPEG-LS, proposed CAVLC, and proposed CABAC.

Image	Original image size (bits)	Method	Total coding bits (bits)	Compression ratio
Foreman (QCIF, 1 frame)	304128	JPEG-LS	174560	1.7423
		Proposed CAVLC	129120	2.3554
		Proposed CABAC	115336	2.6368
Mobile (QCIF, 1 frame)	304128	JPEG-LS	228564	1.3306
		Proposed CAVLC	167544	1.8152
		Proposed CABAC	160392	1.8961
Paris (CIF, 1 frame)	1216512	JPEG-LS	702920	1.7307
		Proposed CAVLC	538000	2.2612
		Proposed CABAC	498160	2.4420
Tempete (CIF, 1 frame)	1216512	JPEG-LS	765624	1.5889
		Proposed CAVLC	584496	2.0813
		Proposed CABAC	559632	2.1738
City_corr (1280 × 720, 1 frame)	11059200	JPEG-LS	6123156	1.8061
		Proposed CAVLC	4848536	2.2809
		Proposed CABAC	4622608	2.3924

Table 13 (*continued*)

Image	Original image size (bits)	Method	Total coding bits (bits)	Compression ratio
Night (1280 × 720, 1 frame)	11059200	JPEG-LS	5328100	2.0756
		Proposed CAVLC	4292520	2.5764
		Proposed CABAC	3966424	2.7882
Parkrun (1920 × 1080, 1 frame)	24883200	JPEG-LS	17805588	1.3975
		Proposed CAVLC	14499376	1.7162
		Proposed CABAC	12456856	1.9975
Crowdrun (1920 × 1080, 1 frame)	24883200	JPEG-LS	15653540	1.5896
		Proposed CAVLC	11443440	2.1745
		Proposed CABAC	10951952	2.2720
Average		JPEG-LS		**1.6577**
		Proposed CAVLC		**2.1576**
		Proposed CABAC		**2.3249**

CAVLC and CABAC schemes provide better coding performance compared to JPEG-LS (Table 13).

5. Conclusions

In this paper, we propose an improved context-based adaptive variable length coder (CAVLC) and context-based adaptive binary arithmetic coder (CABAC) for lossless video coding, based on the traditional CAVLC and CABAC schemes. By considering the statistical differences in residual data between lossy and lossless coding, we designed each new entropy coder by modifying the corresponding encoding parts of each conventional entropy coder. Experimental results show that the proposed methods provide approximately 4% and 13% bit savings, when compared to coding performance of CAVLC and CABAC in the current H.264/AVC FRExt high profile, respectively.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2009-352-D00277), and in part by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-(C1090-1011-0003)).

References

[1] ITU-T Recommendation H.264 and ISO/IEC 14496-10, Advanced Video Coding for Generic Audiovisual Services, May 2003.
[2] A. Luthra, G.J. Sullivan, T. Wiegand, Introduction to the special issue on the H.264/AVC video coding standard,, IEEE Trans. Circuits Syst. Video Technol. 13 (7) (2003) 557–559.
[3] T. Wiegand, G.J. Sullivan, G. Bjøntegaard, A. Luthra, Overview of the H.264/AVC video coding standard, IEEE Trans. Circuits Syst. Video Technol. 13 (7) (2003) 560–576.
[4] G.J. Sullivan, T. Wiegand, Video compression—from concepts to the H.264/AVC standard, Proc. IEEE 1 (2005) 18–31.
[5] G.J. Sullivan, T. McMahon, T. Wiegand, A. Luthra, (Eds.), Draft Text of H.264/AVC Fidelity Range Extensions Amendment to ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/SG16 Joint Video Team Document JVT-L047, July 2004.
[6] G.J. Sullivan, P. Topiwala, A. Luthra, The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions, in: Proceedings of the SPIE conference, Special Session on Advances in the New Emerging Standard: H.264/AVC, August 2004.
[7] S. Sun, Intra lossless coding and QP range selection, ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16 Joint Video Team Document JVT-C023, May 2002.
[8] Y.L. Lee, K.H. Han, G.J. Sullivan, Improved lossless intra coding for H.264/MPEG-4 AVC, IEEE Trans. Image Process. 15 (9) (2006) 2610–2615.
[9] Y.L. Lee, K.H. Han, S.C. Lim, Lossless intra coding for improved 4:4:4 coding in H.264/MPEG-4 AVC, ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16 Joint Video Team Document JVT-P016, July 2005.
[10] Y.L. Lee, K.H. Han, Complexity of the proposed lossless intra, ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Joint Video Team Document JVT-Q035r1, October 2005.
[11] H. Yu, Ed., Draft Text of H.264/AVC Advanced 4:4:4 Profile Amendment to ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16 Joint Video Team document JVT-Q209, October 2005.
[12] G. Bjøntegaard, K. Lillevold, Context-adaptive VLC (CVLC) coding of coefficients, ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/SG16, Joint Video Team Document JVT-C028, May 2002.
[13] I.E.G. Richardson, in: H.264 and MPEG-4 Video Compression, Wiley, 2003.
[14] D. Marpe, H. Schwarz, T. Wiegand, Context-based adaptive binary arithmetic coding in the H.264/AVC video compression, IEEE Trans. Circuits Syst. Video Technol. vol. 13 (no. 7) (2003) 620–636.
[15] Information technology—coding of audio-visual objects–part 10: advanced video coding, International Standard, ISO/IEC 14496-10, Second ed., December 2005.
[16] D. Salomon, in: Variable-length Codes for Data Compression, Springer, 2007.
[17] J. Heo, S.H. Kim, Y.S. Ho, Improved CAVLC for H.264/AVC lossless intra coding, IEEE Trans. Circuits Syst. Video Technol. 20 (2) (2010) 213–222.
[18] J. Heo, Y.S. Ho, Efficient level and zero coding methods for H.264/AVC lossless intra coding, IEEE Signal Process. Lett. 17 (1) (2010) 87–90.
[19] http://iphome/hhi.de/shehring/tml/download/old_jm/jm13.2.zip, Joint Video Team, Reference Software Version 13.2.
[20] M.J. Weinberger, G. Seroussi, G. Sapiro, The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS, IEEE Trans. Image Process. 9 (8) (2000) 1309–1324.
[21] K. Sayood, in: Introduction to Data Compression, Morgan Kaufmann, San Mateo, CA, 2006.