9

Adaptive Entropy Coder Design Based on the Statistics of Lossless Video Signal

Jin Heo and Yo-Sung Ho
Gwangju Institute of Science and Technology (GIST)
261 Cheomdan-gwagiro, Buk-gu, Gwangju 500-712,
Republic of Korea

1. Introduction

H.264/AVC is the latest international video coding standard. It is currently the most powerful and state-of-the-art standard; thus, it can provide enhanced coding efficiency for a wide range of video applications, including video telephony, video conferencing, TV, storage, streaming video, digital cinema, and many others (Luthra et al., 2003; Sullivan & Wiegand, 2005; Wiegand et al., 2003). To date, since H.264/AVC has been developed by mainly focusing on lossy coding, its algorithms have reached a quite mature stage for lossy video compression.

Lossless compression has long been recognized as another important option for application areas that require high quality such as source distribution, digital document, digital cinema, and medical imaging. Recently, as the number of services and popularity for higher quality video representation are expanding, the interest and importance for lossless or near lossless video coding is also increasing (Brunello et al., 2003). However, since the majority of research pertaining to the H.264/AVC standard has focused on lossy video coding, it does not provide good coding performance for lossless video coding.

In order to provide improved functionality for lossless coding, the H.264/AVC standard first included a *pulse-code modulation* (PCM) macroblock coding mode, and then a transform-bypass lossless coding mode (Joint video Team of the International Telecommunications Union-Telecommunication and the International Organization for Standardization/International Electrotechnical Commission [JVT of ITU-T and ISO/IEC], 2002) that employed two main coding processes: prediction and entropy coding which were not previously used in the PCM macroblock coding mode in the fidelity range extensions (FRExt) (JVT of ITU-T and ISO/IEC, 2004; Sullivan et al., 2004). However, since the algorithms for lossless coding are not efficient, more efficient coding techniques for prediction and entropy coding are still required.

Recently, instead of developing a block-based intra prediction, new intra prediction methods, referred to as sample-wise *differential pulse-code modulation* (DPCM) (JVT of ITU-T and ISO/IEC, 2005; Lee et al., 2006) were introduced for lossless coding. As a result, they have been shown to provide better compression performance.

Two entropy coding methods: *context-based adaptive variable length coding* (CAVLC) (JVT of ITU-T and ISO/IEC, 2002; Richardson, 2003) and *context-based adaptive binary arithmetic coding* (CABAC) (Marpe et al., 2003) in the H.264/AVC standard were originally developed

for lossy video coding; they were designed by taking into consideration the typically observed statistical properties of residual data, i.e., the quantized transform coefficients. However, in lossless coding, residual data are just prediction residuals without transform and quantization (Malvar et al., 2003). Thus, the statistical characteristics of residual data from lossy and lossless coding are quite different. As such, the use of conventional entropy coding methods in H.264/AVC is inappropriate for lossless video coding. Nevertheless, most researches into lossless coding in the H.264/AVC standard have focused on improving its prediction ability, rather than on the development of entropy coders (Heo et al., 2010).
Therefore, in this chapter, we have tried to improve coding performance of entropy coders in H.264/AVC for lossless intra coding. After we analyzed the statistical differences of residual data between lossy and lossless coding, we explained an improved CAVLC and CABAC methods for lossless intra coding based on the observed statistical characteristics of lossless coding. Note that our research goal is to improved coding performance of CAVLC and CABAC, which can then be easily applied to H.264/AVC lossless intra coding by modifying the semantics and decoding processes without requiring any other syntax elements in the H.264/AVC standard.

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

In this section, we review two entropy coding methods: CAVLC and CABAC in H.264/AVC. The entropy coders are employed to encode residual data; zigzag scanned the quantized transform coefficients, for a 4×4 sub-block. Fig. 1 illustrates the zigzag scan order for the 4×4 sub-block.

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. 1. Zigzag scan order for the sub-block

2.1 Overview of CAVLC

The encoding structure of CAVLC for a 4×4 sub-block is depicted in Fig. 2. First, both the number of non-zero coefficients and the number of trailing ones are encoded using a combined codeword (*coeff_token*). Second, the sign of each trailing one is encoded using a 1-bit codeword in reverse order (*trailing_ones_sign_flag*). Third, the absolute value of the level of each remaining non-zero coefficient is encoded in reverse order using one of the seven predefined *Lev-VLC* tables and the sign information is encoded (*level*). Fourth, the number of

all zeros before the last non-zero coefficient is encoded (*total_zeros*). Last, the number of consecutive zeros preceding each non-zero coefficient is encoded in reverse order (*run_before*).

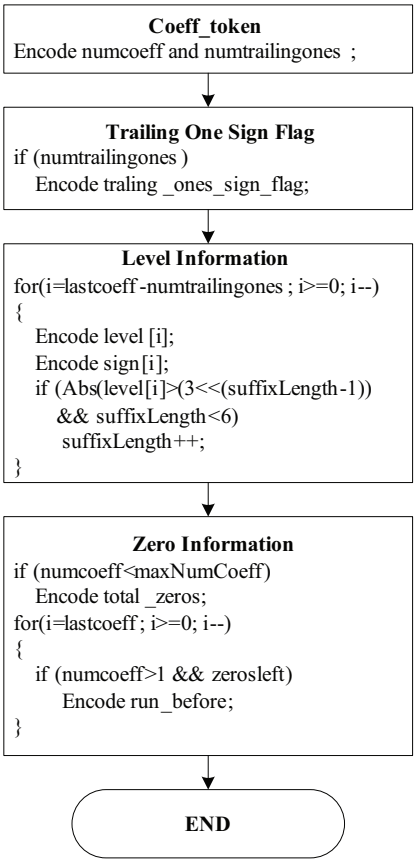

Fig. 2. Encoding structure of CAVLC for residual data coding

More details of each coding step are described below.

Step 1. Encode the number of non-zero coefficients (*numcoeff*) and the number of trailing ones (*numtrailingones*).

A trailing one is one of up to three consecutive non-zero coefficient at the end of the scan of non-zero coefficients having an absolute value equal to 1. If there are more than three trailing ones, only the last three are treated as trailing ones, with any others being coded as normal coefficients.

The four VLC tables used for encoding *coeff_token* are comprised of three variable-length code tables (*Num-VLC0*, *Num-VLC1*, and *Num-VLC2*) and one fixed-length code table (*FLC*). The choice of VLC table depends on the number of non-zero coefficients in the previously coded upper and left sub-blocks. If both the upper and left sub-blocks are available, $N=round(N_U+N_L)/2$. If only the upper sub-block is

available, $N=N_U$; if only the left sub-block is available, $N=N_L$. If neither is available, N is set to zero. Where N is the number of predicted non-zero coefficients in the current sub-block, and N_U and N_L represent the number of non-zero coefficients in the upper and left previously encoded sub-blocks, respectively. Thus, based on the parameter N, an appropriate VLC table for the current sub-block is selected from Table 1.

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

Table 1. Choice of VLC table

Step 2. Encode the sign of each trailing one.
The trailing one sign flag indicates the sign information of a trailing one coefficient; the sign information is simply encoded by a 1-bit codeword in reverse order. If the sign information is positive (+), *trailing_ones_sign_flag* is equal to zero. Conversely, if the sign information is negative (-), *trailing_ones_sign_flag* is equal to one.

Step 3. Encode the levels.
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 among seven *Lev-VLC* tables (Table 2), with selection of the *Lev-VLC* table dependent on the magnitude of each recently encoded level. The choice of *Lev-VLC* table is adapted as follows:

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

Lev-VLC table	Threshold to increment *Lev-VLC table*
Lev-VLC0	0
Lev-VLC1	3
Lev-VLC2	6
Lev-VLC3	12
Lev-VLC4	24
Lev-VLC5	48
Lev-VLC6	> 48

Table 2. Thresholds for determining whether to increment *Lev-VLC* table

Step 4. Encode the total number of zeros.
After the encoding process for level information, zeros remain. CAVLC encodes the syntax element, *total_zeros* which represents the number of zero coefficients located before the last non-zero coefficient.

Step 5. Encode each run of zeros.
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 with *zerosleft* in reverse order. Note that *zerosleft* indicates the number of zeros that has not yet been encoded. The syntax element *run_before* is encoded for each non-zero coefficient, with two exceptions:

1. If there are no *zerosleft* to encode, processing can be stopped.
2. Processing can be stopped to encode *run_before* for the final (lowest frequency) non-zero coefficient.

2.2 Overview of CABAC

2.2.1 CABAC framework

The encoding process of CABAC consists of four coding steps: binarization, context modeling, binary arithmetic coding, and probability update. The block diagram for encoding a single syntax element in CABAC is depicted in Fig. 3.

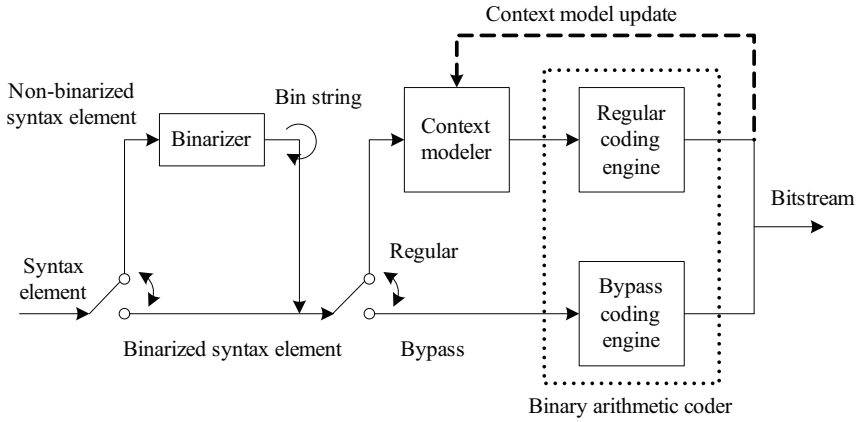

Fig. 3. CABAC encoder framework

In the first step, a given non-binary valued syntax element is uniquely mapped to a binary sequence (*bin string*); when the binary valued syntax element is given, the first step is bypassed. In the regular coding mode, each binary value (*bin*) of the binary sequence enters the context modeling stage, where a probability model is selected based on the previously encoded syntax elements. Then, the arithmetic coding engine encodes each binary value with its associated probability model. Finally, the selected context model is updated according to the actual coded binary value. Alternatively, in the bypass coding mode, each binary value is directly encoded via the bypass coding engine without using an explicitly assigned model.

2.2.2 CABAC for residual data coding

Fig. 4 illustrates the CABAC encoding structure for a 4×4 sub-block of the quantized transform coefficients. First, the coded block flag is transmitted for the given sub-block unless the coded block pattern or the macroblock mode indicates that the specific sub-block has no non-zero coefficient. If the coded block flag is zero, no further information is transmitted for the current sub-block; otherwise, the significance map and level information are sequentially encoded. More details of each coding step are described below.

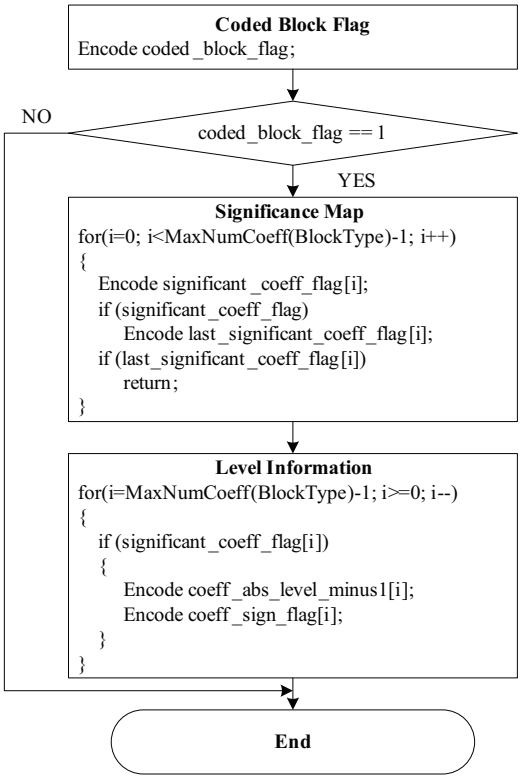

Fig. 4. Encoding structure of CABAC for residual data coding.

Step 1. Encode coded block flag.
For each 4×4 sub-block, a 1-bit symbol *coded_block_flag* is transmitted to indicate that a sub-block has significant coefficients. If *coded_block_flag* is zero, no further information is transmitted and the coded block flag coding process is terminated for the current sub-block. However, if *coded_block_flag* is one, the significance map and level information coding processes are continued.

Step 2. Encode significance map.
If *coded_block_flag* indicates that a sub-block has significant coefficients, a binary-valued significance map is encoded. For each coefficient, a 1-bit syntax element *significant_coeff_flag* is encoded in scanning order. If *significant_coeff_flag* is one, i.e., if a non-zero coefficient exists at this scanning position, a further 1-bit syntax

element *last_significant_coeff_flag* is encoded. This syntax element states whether the current significant coefficient is the last coefficient inside the sub-block or not. Note that *significant_coeff_flag* and *last_significant_coeff_flag* for the last scanning position of a sub-block are not encoded.

Step 3. Encode level information.

After the encoded significance map determines the locations of all significant coefficients inside a sub-block, the values of the significant coefficients are encoded by using two syntax elements: *coeff_abs_level_minus1* and *coeff_sign_flag*. The syntax element *coeff_sign_flag* is encoded by a 1-bit symbol, whereas the *Unary/0th order Exponential Golomb* (UEG0) binarization method is used to encode the values of *coeff_abs_level_minus1* representing the absolute value of the level minus 1. The values of the significant coefficients are encoded in reverse scanning order.

3. Analysis of the statistical characteristics of residual data in lossless coding

In lossy coding, residual data represent the quantized transform coefficients. The statistical characteristics of residual data in lossy coding are as follows. In a given sub-block, the probability of a non-zero coefficient existing 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, the occurrence probability of a trailing one is relatively high.

In lossless coding, however, residual data do not represent the quantized transform coefficients, but rather the differential pixel values between the original and predicted pixel values. Therefore, the statistical characteristics of residual data in lossless coding are as follows. First, the probability of a non-zero pixel existing is independent of the scanning position, and the number of non-zero pixels is generally large, compared to the number of non-zero coefficients in lossy coding. Second, the absolute value of a non-zero pixel does not decrease as the scanning position increases and is independent of the scanning position. Finally, the occurrence probability of a trailing one is not so high.

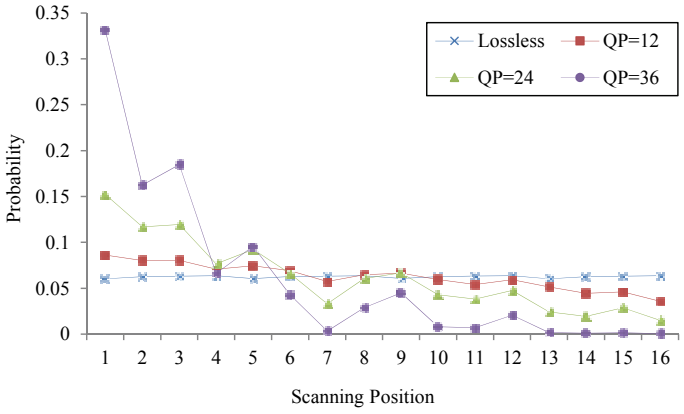

Fig. 5. Probability distribution of non-zero coefficients according to the scanning position.

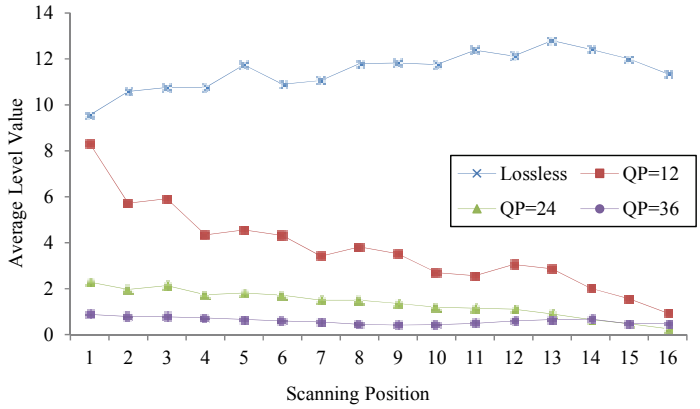

Fig. 6. Distribution of average absolute value according to the scanning position.

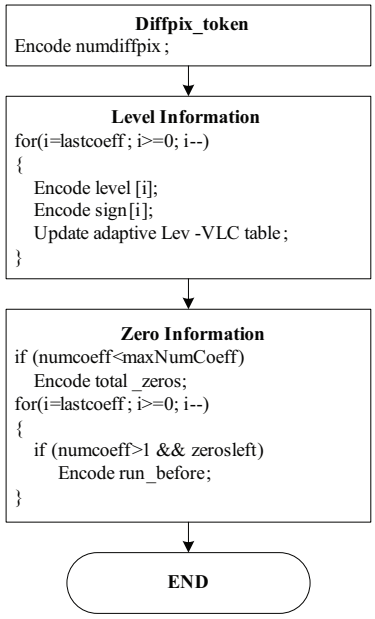

Fig. 7. Encoding structure of the proposed CAVLC for differential pixel value coding.

Figs. 5 and 6 represent the probability distribution of non-zero coefficients existing and the distribution of average absolute value according to the scanning position, respectively. As expected, significant differences can be seen in the statistics between the residual data of lossy and lossless coding.

Therefore, based on the above statistical characteristics of residual data in lossless coding, we propose more efficient CAVLC and CABAC methods for lossless compression in H.264/AVC by modifying the relevant coding parts of each entropy coder.

4. Improved CAVLC

In this section, we introduce an improved CAVLC for lossless intra coding. In Fig. 7, we depict the encoding structure of the proposed method for encoding the differential pixel value in lossless coding. The encoding procedure of the proposed CAVLC method can be summarized in the following steps:

Step 1. Encode the total number of non-zero differential pixels (*diffpix_token*).
Step 2. Encode the level (sign and magnitude) of all non-zero differential pixels (*level*).
Step 3. Encode the number of all zeros before the last non-zero differential pixel (*total_zeros*).
Step 4. Encode the number of consecutive zeros preceding each non-zero differential pixel (*run_before*).

Further details of these coding methods are described in the following subsections.

4.1 Coding the number of non-zero differential pixels

Table 3 represents the occurrence probability distribution of trailing ones according to the quantization parameter (QP). Since, in lossless coding, the occurrence probability of trailing ones turns out to be relatively lower than that in lossy coding, the trailing one does not need to be treated as a special case of encoding. Therefore, in this step, we encode the total number of non-zero differential pixels (*numdiffpix*) but do not consider the number of trailing ones (*numtrailingones*). Since the trailing ones are treated as normal coefficients, they are encoded in the level coding step, which thereby enabled the removal of Step 2, a coding stage of the sign information for each trailing one.

Sequence	QP			
	0 (Lossless)	12	24	36
News	0.3719	0.8161	0.8825	0.9458
Foreman	0.2598	0.7947	0.9117	0.9585
Mobile	0.2107	0.6962	0.8566	0.9268
Tempete	0.2274	0.7842	0.8861	0.9449
City_corr	0.2100	0.8140	0.8914	0.9507
Crowdrun	0.1813	0.7673	0.9240	0.9478

Table 3. Occurrence probability distribution of trailing ones according to the QP

In CAVLC, the corresponding VLC table is selected based on the predicted *numcoeff*; further details have already been explained in Section 2.1. Note that if the predicted *numcoeff* is larger than seven, the fixed length code (*FLC*) table is selected, as described in Table 1. In lossless coding, the *FLC* table is most often selected because *numdiffpix* is generally larger than seven, as shown in Table 4. From extensive experiments on lossless intra coding with various test sequences, we observed that the *FLC* table was selected about 95% of the time. Hence, we could remove three VLC tables (*Num-VLC0* to *Num-VLC2*) in this step. Since only the *FLC* table is used, we do not need to consider the process for predicting *numcoeff*.

The *FLC* table consists of 4 bits for *numcoeff* and 2 bits for *numtrailingones* in lossy coding; since *numtrailingones* does not need to be considered in lossless coding; only 4 bits for *numdiffpix* remain. However, instead of using the *FLC* table, which uniformly assigns 4 bits

for all *numdiffpix*s, we designed a simple but effective VLC table according to the statistical characteristics of *numdiffpix* in lossless coding.

Sequence	QP			
	0 (Lossless)	12	24	36
News	12.5791	6.3077	3.3553	1.5448
Foreman	13.7457	7.8073	3.3253	1.0017
Mobile	14.6338	10.9796	6.6945	2.4879
Tempete	13.9684	9.0754	4.9113	1.6940
City_corr	14.4775	6.5353	3.3869	0.9449
Crowdrun	14.9614	10.4297	4.0696	1.3231

Table 4. Average number of non-zero coefficients in a sub-block

Fig. 8 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 differential pixels turns out to be very low when the number of non-zero differential pixels is small (the number of non-zero differential pixels < 10). However, the probability of the number of non-zero differential pixels drastically increases as the number of non-zero differential pixels increases, especially the number of non-zero differential pixels from 13 to 16.

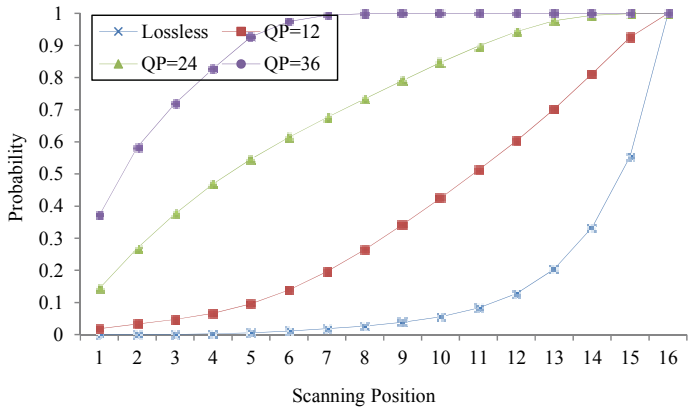

Fig. 8. Cumulative probability distribution of the number of non-zero coefficients.

In our proposed VLC table, first, we assign 4-bit and 2-bit codewords to *numdiffpix* from 1 to 12 and 13 to 16, respectively. In order to enhance coding performance, we assign the different length of codeword to *numdiffpix* from 1 to 12 according to the statistical characteristics of *numdiffpix* instead of assigning 4-bit codewords uniformly. Thus, we use the phased-in code (Salomon, 2007) which is a slight extension of fixed length code (*FLC*). The phased-in code consists of codewords with two different lengths. Therefore, we assign 4-bit and 3-bit codewords to *numdiffpix* 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 5.

numdiffpix	Codeword		
	Check bit	Bits for *numdiffpix*	Codeword length
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

Table 5. Codeword table for *numdiffpix*

4.2 Level coding

In level coding, the absolute value of each non-zero coefficient (*abs_level*) is adaptively encoded by a selected *Lev-VLC* table from the seven predefined *Lev-VLC* tables (*Lev-VLC0* to *Lev-VLC6*) in reverse scanning order. Each *Lev-VLC* table is designed to encode efficiently in a specified range of *abs_level*, as described in Table 2. As previously mentioned, selection of the *Lev-VLC* table for level coding in CAVLC is based on the expectation that *abs_level* is likely to increase at low frequencies. Hence, selection of the *Lev-VLC* table number monotonically increases according to the previously encoded *abs_level*.

However, the absolute value of the differential pixel (*abs_diff_pixel*) in lossless coding is independent of the scanning position, as shown in Fig. 6. Therefore, we designed an adaptive method for *Lev-VLC* table selection that can decrease or increase according to the previously encoded *abs_diff_pixel*.

In lossy coding, CAVLC typically determines the smallest *Lev-VLC* table in the range of possible *Lev-VLC* tables based on the assumption that the next *abs_level* to be coded is going to be larger. However, in lossless coding, the next *abs_diff_pixel* does not necessarily increase at lower frequencies—we cannot assume that the next *abs_diff_pixel* is larger than the current *abs_diff_pixel*. Therefore, the *Lev-VLC* table for each *abs_diff_pixel* should be selected by considering the previously encoded *abs_diff_pixels* because we cannot predict whether or not the next *abs_diff_pixel* will increase.

In order to determine the most appropriate *Lev-VLC* table, we assign a weighting value to the previously encoded *abs_diff_pixels*. The basic idea for this concept is that the *Lev-VLC* table for the next *abs_diff_pixel* can be determined using the weighted sum of the previously encoded *abs_diff_pixels*. The decision procedure for determining the *Lev-VLC* table is described as follows.

$$T(abs\_diff\_pixel_i) = \frac{1}{a_i + 1}\{a_i \cdot avg_i + abs\_diff\_pixel_i\}, \quad (1)$$

$$a_i = \begin{cases} 0, & i = lastdiffpix \\ 1, & i = lastdiffpix - 1,\ lastdiffpix - 2, \\ 2, & \text{otherwise} \end{cases} \quad (2)$$

$$avg_i = \frac{1}{(lastdiffpix - i + 1)}\left\{ \sum_{k=lastdiffpix}^{i} abs\_diff\_pixel_k \right\}, \quad (3)$$

where a_i and $abs\_diff\_pixel_i$ are the weighting coefficient and *abs_diff_pixel* value, respectively, where both values are related to the current scanning position *i*. In addition, $T(abs\_diff\_pixel_i)$ and *lastdiffpix* represent the threshold value for selecting the corresponding *Lev-VLC* table used to encode the next *abs_diff_pixel* ($(i\text{-}1)^{th}$ *abs_diff_pixel*) and the scanning position number of the last non-zero differential pixel, respectively. Note that *abs_diff_pixel* is encoded in reverse order. In Table 6, we represent the *Lev-VLC* table for level coding according to $T(abs\_diff\_pixel_i)$. From extensive experiments on lossless intra coding using various test sequences, we could determine these optimal threshold values.

Lev-VLC table	$T(abs\_diff\_pixel_i)$
Lev-VLC0	0
Lev-VLC1	2
Lev-VLC2	4
Lev-VLC3	9
Lev-VLC4	19
Lev-VLC5	39
Lev-VLC6	> 39

Table 6. New thresholds for determining the *Lev-VLC* table

In Fig. 6, we can note that the last scanned absolute values are 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 is likely to be small. However, in lossless coding, the last scanned *abs_diff_pixel* is not small enough to use either *Lev-VLC0* or *Lev-VLC1*. Table 7 represents the average absolute value of the last scanned level for the sub-blocks. In Table 7, the last scanned *abs_diff_pixel* in lossless coding is larger than the last scanned *abs_level* in lossy coding. The average absolute value of the last scanned differential pixel in the sub-blocks is approximately 10.09 in lossless coding. Based on this value, we adjusted the initial *Lev-VLC* table for level coding. The modified *Lev-VLC* table selection method is follows.

1. Level coding starts with *Lev-VLC4*.
2. Encode the absolute value of the last scanned differential pixel.
3. Encode the sign of the last scanned differential pixel.
4. Update the *Lev-VLC* table by considering the previously encoded *abs_diff_pixel*s and new threshold for each *Lev-VLC* table.

Sequence	QP			
	0 (Lossless)	12	24	36
News	9.9371	2.5228	2.0837	1.9113
Foreman	9.2097	2.2939	1.8902	1.8881
Mobile	16.3748	3.0935	2.1962	1.7870
Tempete	11.5824	2.6421	1.9960	1.7655
City_corr	6.9376	1.2654	1.1340	1.0572
Crowdrun	8.8483	1.3609	1.0906	1.0611

Table 7. Average absolute value of the last non-zero coefficient for the sub-blocks

5. Improved CABAC

In this section, we describe an improved CABAC for lossless intra coding. In Fig. 9, we depict the encoding structure of the proposed method for encoding the differential pixel value in lossless coding. The encoding procedure of the proposed CABAC can be summarized in the following steps:

Step 1. Encode whether the current sub-block contains non-zero pixel values (*coded_block_flag*)

Step 2. Encode whether the differential pixel value at each scanning position is non-zero to the last scanning position (*significant_diff_pixel_flag*).

Step 3. Encode the absolute value of a differential pixel value minus 1 with modified binarization method (*abs_diff_pixel_minus1*).

Step 4. Encode the sign of a differential pixel value (*diff_pixel_sign_flag*).

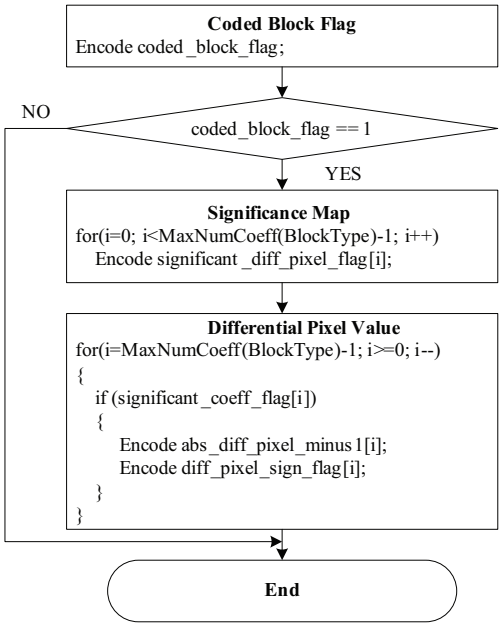

Fig. 9. Encoding structure of the proposed CABAC for differential pixel value coding.

Further details of these coding methods are described in the following subsections.

5.1 Significance map coding

In lossy coding, the occurrence probability of a non-zero coefficient is likely to decrease as the scanning position increases because residual data are the quantized transform coefficients. Therefore, the significant coefficient tends to be located at earlier scanning positions. In this case, *last_significant_coeff_flag* plays an important role in the early termination of significance map coding.

However, in lossless coding, since neither transform nor quantization is performed, the occurrence probability of a non-zero differential pixel is independent of the scanning position, as shown in Fig. 5. Thus, the last non-zero differential pixel is terminated at the end of the scanning position, as shown in Table 8. In this case, it is meaningless to encode *last_significant_coeff_flag* to indicate the position of the last significant differential pixel. Therefore, we remove the *last_significant_coeff_flag* coding process and directly encode *significant_diff_pixel_flag*s at all scanning positions from 1 to 16 in the proposed significance map coding.

Sequence	QP			
	0 (Lossless)	12	24	36
News	14.5484	9.8368	6.6348	4.0463
Foreman	14.7669	12.5503	6.9935	2.8340
Mobile	14.7906	12.8480	10.4510	6.2891
Tempete	14.7868	12.4645	9.0398	3.8092
City_corr	14.7806	10.4116	5.6708	2.3080
Crowdrun	14.8204	13.6042	6.6730	2.6177

Table 8. Average location of the last non-zero coefficient in a sub-block

Fig. 10 represents an example of significance map coding for CABAC in lossy coding when the scanning position of the last significant coefficient is 14; the gray shaded *significant_coeff_flag* and *last_significant_coeff_flag* are encoded in significance map coding. Note that both *significant_coeff_flag* and *last_significant_coeff_flag* for the last scanning position of a sub-block are never encoded.

Scanning position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Transform coefficient level	9	0	-5	3	0	-7	4	0	8	-11	-6	0	3	1	0	0
significant_coeff_flag	1	0	1	1	0	1	1	0	1	1	1	0	1	1		
last_significant_coeff_flag	0		0	0		0	0		0	0	0		0	1		

Fig. 10. Example of significance map coding in lossy coding.

However, since we removed the *last_significant_coeff_flag* coding process in lossless coding, *significant_diff_pixel_flag* is unconditionally encoded up to the last scanning position. Fig. 11 presents an example of significance map coding in lossless coding. All gray shaded *significant_diff_pixel_flag*s are encoded in the proposed significance map coding.

Scanning position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Differential pixel value	9	0	-5	3	0	-7	4	0	8	-11	-6	0	3	1	0	0
significant_diff_pixel_flag	1	0	1	1	0	1	1	0	1	1	1	0	1	1	0	0

Fig. 11. Example of significance map coding in lossless coding.

5.2 Binarization for differential pixel value

For the absolute value of the quantized transform coefficient (*abs_level*) in lossy coding, the *Unary/k*th *order Exponential Golomb* (UEGk) binarization method is applied. The design of the UEGk binarization method is motivated by the fact that the unary code is the simplest prefix-free code in terms of implementation cost and it permits the fast adaptation of individual symbol probabilities in the subsequent context modeling stage. These observations are only accurate for small *abs_level*s; however, for larger *abs_level*s, adaptive modeling has limited functionality. Therefore, these observations have led to the idea of concatenating an adapted truncated unary (TU) code as a prefix and a static Exp-Golomb code (Teuhola, 1978) as a suffix.

The UEGk binarization of *abs_level* has a cutoff value $S = 14$ for the TU prefix and the order $k = 0$ for the Exp-Golomb (EG0) suffix. Previously, Golomb codes have been proven to be optimal prefix-free codes for geometrically distributed sources (Gallager & Voorhis, 1975). Moreover, EG0 is the optimal code for a *probability density function* (pdf) as follows:

$$p(x) = 1/2 \cdot (x+1)^{-2} \quad with \; x \geq 0 \tag{4}$$

The statistical properties of the absolute value of the differential pixel (*abs_diff_pixel*) in lossless coding are quite different from those of *abs_level* in lossy coding. In lossy coding, the statistical distribution of *abs_level* is highly skewed on small values. However, in lossless coding, the statistical distribution of *abs_diff_pixel* is quite wide; note the large variation and wide tails, shown in Fig. 12. Moreover, we can also observe that the TU code is a fairly good model for the statistical distribution of *abs_level* in lossy coding; whereas, it is not appropriate for the statistical distribution of *abs_diff_pixel* in lossless coding. Therefore, as UEG0 binarization was originally designed for lossy coding, it is not appropriate for lossless coding.

In order to efficiently encode *abs_diff_pixel* in lossless coding, we adjusted the cutoff value S of the TU prefix in UEG0 binarization. In Fig. 12, the optimal pdf curve for the TU code and the statistical distribution curve for *abs_diff_pixel* in lossless coding intersect at an absolute value of 5. Moreover, as the absolute value increases, the statistical difference between the TU code and *abs_diff_pixel* in lossless coding becomes larger. Therefore, we determined a new cutoff value $S = 5$ for the TU prefix in the proposed binarization method.

In order to provide a good prefix-free code for lossless coding, we also determined an appropriate parameter k for the EGk code. The prefix of the EGk codeword consists of a unary code corresponding to the value $l(x) = \left\lfloor \log_2(x/2^k + 1) \right\rfloor$. The suffix is then computed as the binary representation of $x + 2^k(1 - 2^{l(x)})$ using $k + l(x)$ significant bits. Consequently, for EGk binarization, the number of symbols having the same code length of $k + 2l(x) + 1$ grows geometrically. Then, by inverting Shannon's relationship between the ideal code length and the symbol probability, we can find each pdf corresponding to an EGk having an optimal code according to a parameter k.

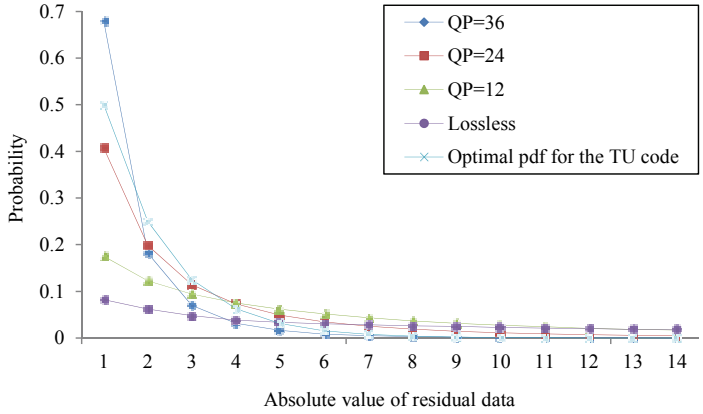

Fig. 12. Probability distribution of the absolute value of residual data and the optimal pdf of the TU code.

$$p_k(x) = 1/2^{k+1} \cdot (x/2^k + 1)^{-2} \quad with \; x \geq 0 \tag{5}$$

where $p_k(x)$ is the optimal pdf corresponding to the EGk code for a parameter k. This implies that for an appropriately chosen parameter k, the EGk code represents a fairly good prefix-free code for tails of typically observed pdfs.

Fig. 13 represents the probability distribution of $p_k(x)$ for k = 0, 1, 2, and 3 and the occurrence probability distribution of *abs_diff_pixel* from 6 to 20, where *abs_diff_pixel*s up to 5 are specified by the TU code. In the figure, the probability distribution of $p_k(x)$ for k = 3 is well matched to the occurrence probability distribution of *abs_diff_pixel*. This result implies that the EG3 code represents a fairly good approximation of the ideal prefix-free code for encoding *abs_diff_pixel* in lossless coding.

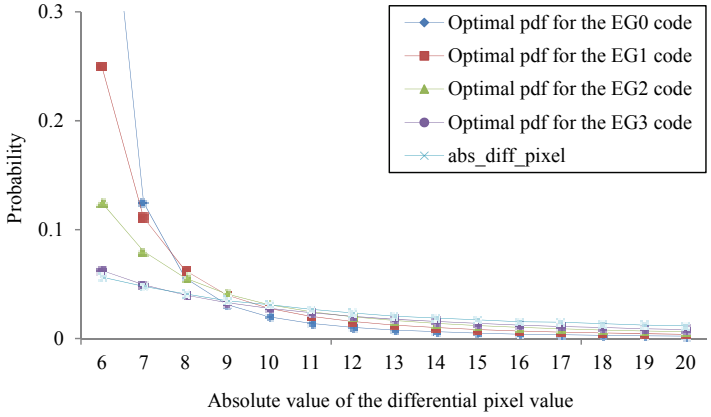

Fig. 13. Probability distribution of the optimal pdf corresponding to the EGk code for k = 0, 1, 2, and 3 and the probability distribution of the absolute value of the differential pixel.

Based on these observations, we designed an efficient binarization algorithm to encode *abs_diff_pixel* in lossless coding. In the proposed algorithm, UEGk binarization of *abs_diff_pixel* is specified by the cutoff value $S = 5$ for the TU prefix and the order $k = 3$ for the EGk suffix. Table 9 shows the proposed UEG3 binarization for *abs_diff_pixel*.

abs_diff_pixel	Bin string											
	TU prefix					EG3 suffix						
1	0											
2	1	0										
3	1	1	0									
4	1	1	1	0								
5	1	1	1	1	0							
6	1	1	1	1	1	0	0	0	0			
7	1	1	1	1	1	0	0	0	1			
8	1	1	1	1	1	0	0	1	0			
9	1	1	1	1	1	0	0	1	1			
10	1	1	1	1	1	0	1	0	0			
11	1	1	1	1	1	0	1	0	1			
12	1	1	1	1	1	0	1	1	0			
13	1	1	1	1	1	0	1	1	1			
14	1	1	1	1	1	1	0	0	0	0	0	
15	1	1	1	1	1	1	0	0	0	0	1	
...	...					...						
Bin index	1	2	3	4	5	6	7	8	9	10	11	...

Table 9. Proposed UEG3 binarization for encoding the absolute value of the differential pixel

6. Experimental results and analysis

In this chapter, we introduced the improved CAVLC and CABAC methods for lossless intra coding. In order to verify coding efficiency of the proposed methods, we performed experiments on several test sequences of YUV 4:2:0 and 8 bits per pixel format with QCIF, CIF, and HD resolutions. We implemented our proposed methods in the H.264/AVC reference software version JM13.2 (Fraunhofer Institute for Telecommunications Heinrich Hertz Institute, 2011). Table 10 shows the encoding parameters for the reference software.

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

Table 10. Encoding parameters

In order to evaluate coding performance for the proposed CAVLC and CABAC methods, we consider two sections based on the following settings in each method.

Entropy coder	Method	Description
CAVLC	*Method I*	Modify *numdiffpix* coding
	Method II	Method I + modify level coding
CABAC	*Method III*	Modify significance map coding
	Method IV	Method III + modify binarization

Table 11. Each two coding variations in the proposed CAVLC and CABAC methods

Note that these proposed methods were applied to H.264/AVC lossless intra coding by modifying the semantics and decoding processes, without adding any syntax elements to the H.264/AVC standard. The proposed method was implemented on top of the previous sample-wise DPCM prediction method, and it further enhanced coding efficiency for lossless intra coding in H.264/AVC.

To verify efficiency of the proposed methods, we performed two kinds of experiments. In the first experiment, we compared coding perforamcne of the origianl CAVLC and proposed CAVLC methods and then coding performance of the original CABAC and proposed CABAC methods in Tables 12 and 13, respectively. In the second experiment, we encoded only one frame (first frame) for each test sequence using our proposed methods (*Method II* and *Method IV*) and a well-known lossless coding techniques, lossless joint photographic experts group (JPEG-LS) (Sayood, 2006; Weinberger et al., 2000) used as a comparison for coding performance of our proposed methods.

Sequence	Size (bits)	Method	Total bits (bits)	Bit saving (%)
News (QCIF, 176×144) 100 frames	30412800	H.264/AVC CAVLC	13319592	0
		Method I	12772832	4.105
		Method II	12260784	7.949
Foreman (QCIF, 176×144) 100 frames	30412800	H.264/AVC CAVLC	14151096	0
		Method I	13595488	3.926
		Method II	12960664	8.412
Mobile (CIF, 352×288) 100 frames	121651200	H.264/AVC CAVLC	72406600	0
		Method I	70186288	3.066
		Method II	62959352	13.047
Tempete (CIF, 352×288) 100 frames	121651200	H.264/AVC CAVLC	65508056	0
		Method I	63271688	3.414
		Method II	58310688	10.987
City_corr (HD, 1280×720) 100 frames	1105920000	H.264/AVC CAVLC	517138472	0
		Method I	497092864	3.876
		Method II	478721808	7.429
Crwodrun (HD, 1920×1080) 100 frames	2488320000	H.264/AVC CAVLC	1177553944	0
		Method I	1131495264	3.911
		Method II	1080073896	8.278
Average		H.264/AVC CAVLC		0
		Method I		**3.716**
		Method II		**9.350**

Table 12. Comparison of bit savings for H.264/AVC CAVLC and the proposed CAVLC

Comparisons were made in terms of bit-rate percentage differences (Tables 12 and 13) and compression ratio differences (Table 14) with respect to the origianl entropy coding methods in H.264/AVC and JPEG-LS, respectively. These changes were calculated as follows:

$$\Delta Saving\ Bits(\%) = \frac{Bitrate_{H.264/AVC} - Bitrate_{Method}}{Bitrate_{H.264/AVC}} \times 100 \tag{6}$$

$$Compression\ Ratio = \frac{Original\ image\ size}{Bitrate_{Method}} \tag{7}$$

Sequence	Size (bits)	Method	Total bits (bits)	Bit saving (%)
News (QCIF, 176×144) 100 frames	30412800	H.264/AVC CABAC	13941080	0
		Method III	12563136	9.884
		Method IV	11760776	15.639
Foreman (QCIF, 176×144) 100 frames	30412800	H.264/AVC CABAC	14344176	0
		Method III	12857928	10.361
		Method IV	12572368	12.352
Mobile (CIF, 352×288) 100 frames	121651200	H.264/AVC CABAC	91371512	0
		Method III	85034984	6.935
		Method IV	68152408	25.412
Tempete (CIF, 352×288) 100 frames	121651200	H.264/AVC CABAC	79063136	0
		Method III	72756080	7.977
		Method IV	60830560	23.061
City_corr (HD, 1280×720) 100 frames	1105920000	H.264/AVC CABAC	565080864	0
		Method III	507403880	10.207
		Method IV	470393024	16.757
Crowdrun (HD, 1920×1080) 100 frames	2488320000	H.264/AVC CABAC	1250235376	0
		Method III	1120777696	10.355
		Method IV	1047171240	16.242
Average		H.264/AVC CABAC		0
		Method III		**9.287**
		Method IV		**18.244**

Table 13. Comparison of bit savings for H.264/AVC CABAC and the proposed CABAC

In Tables 12 and 13, we confirmed that the proposed CAVLC and CABAC methods provided better coding performance compared to the conventioanl CAVLC and CABAC methods—by approximately 9% and 18% bit savings, respectively. Table 14 presents the experimental results comparing a well-known lossless coding techniques, JPEG-LS, in terms of lossless intra coding, which again shows that the proposed methods displayed better coding performance compared to JPEG-LS in lossless coding.

Lossless compression techniques, such as JPEG-LS and H.264/AVC lossless mode consist of two independent coding parts; prediction based on modeling and entropy coding of prediction residuals. In JPEG-LS, a simple predictive coding model called *differential pulse-code modulation* (DPCM) is employed. This is a model in which predictions of the sample values are estimated from the neighboring samples that are previously coded in the image.

Most predictors take the average of the samples immediately above and to the left of the target sample. In H.264/AVC, a similar DPCM is employed to predict the original pixel value, but it employs rate-distortion optimization (RDO) (Sullivan & Wiegand, 1998) method to find the best prediction. Hence, H.264/AVC requires the additional coding bits to send the prediction mode but it can reduce more coding bits in the residual coding. However, when the residual data is entered into the entropy coding part, JPEG-LS provides better coding performance than H.264/AVC lossless mode because H.264/AVC still employs CAVLC or CABAC which are mainly designed for discrete cosine transform(DCT)-based lossy coding. As a result, JPEG-LS and H.264/AVC lossless mode provide quite similar coding performance. Since, in this chapter, we have proposed the improved CAVLC and CABAC methods for lossless intra coding, coding performance of the H.264/AVC lossless mode based on the proposed methods is better than that of JPEG-LS, as shown in Table 14.

Sequence	Method	Compression ratio
News (QCIF, 176×144) 1 frame	JPEG-LS	2.0872
	Method II	**2.4660**
	Method IV	**2.5948**
Foreman (QCIF, 176×144) 1 frame	JPEG-LS	1.8179
	Method II	**2.3554**
	Method IV	**2.4528**
Mobile (CIF, 352×288) 1 frame	JPEG-LS	1.4865
	Method II	**1.9515**
	Method IV	**1.7992**
Tempete (CIF, 352×288) 1 frame	JPEG-LS	1.6556
	Method II	**2.0813**
	Method IV	**2.0013**
City_corr (HD, 1280×720) 1 frame	JPEG-LS	1.9079
	Method II	**2.2809**
	Method IV	**2.3248**
Crwodrun (HD, 1920×1080) 1 frame	JPEG-LS	1.6802
	Method II	**2.1745**
	Method IV	**2.1628**
Average	JPEG-LS	1.7726
	Method II	**2.2183**
	Method IV	**2.2226**

Table 14. Comparison of compression ratio for JPEG-LS, *Method II*, and *Method IV*

Let us now give some information on why we do not address lossless inter coding and why it is outside the scope of our work. Since transform and quantization are not used in lossless video coding, the statistical properties of residual data highly depend on prediction. In general, since video sequences contain more redundancy in time than in space, the accuracy of inter prediction is better than that of intra prediction. Thus, there are significant statistical differences in residual data between lossless intra and lossless inter coding. In other words, for lossless intra prediction, the distribution of the amplitude of residual data is quite wide; in contrast, for lossless inter prediction, the distribution of the amplitude of residual data is

skewed on small values. Finally, it is not easy to determine the best entropy coding method that can generally be used for lossless video coding (for both intra and inter coding). Therefore, in this chapter, we focused on the improvement of an appropriate entropy coder for lossless intra coding—though there could be a future work focused on improving upon current entropy coders for lossless video coding.

In terms of scanning patterns for lossy coding, coding performance can change according to various scanning patterns because residual data are the quantized transform coefficients, and the statistical distribution of these coefficients is highly skewed on small values, as depicted in Figs. 5, 6, and 12. Hence, if we can determine an appropriate scanning pattern, we can enhance coding performance by arranging the quantized transform coefficients according to their amplitudes. However, in lossless coding, the statistical distribution of residual data is quite wide; the figures also show that large variations and wide tails are independent of the scanning position. Therefore, theoretically, there is no scanning order that can provide better coding efficiency than that obtained here; we subsequently confirmed this fact by performing extensive experiments using various scanning patterns, including experiments using a zigzag scanning order. Finally, determining the optimum scanning pattern that can be generally accepted for lossless coding is rather difficult. However, a future work may be based on this topic.

7. Conclusion

In this chapter, we proposed the improved *context-based adaptive variable length coding* (CAVLC) and *context-based adaptive binary arithmetic coding* (CABAC) methods for lossless intra coding. 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 based on the observed statistical characteristics of residual data in lossless coding. Experimental results show that the proposed CAVLC and CABAC methods provided approximately 9% and 18% bit savings, compared to the original CAVLC and CABAC methods in the H.264/AVC FREext high profile, respectively.

8. Acknowledgment

This research was supported 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-2011-(C1090-1111-0003)).

9. References

Bjontegaard G. & Lillevold K. (2002). Context-Adaptive VLC (CVLC) Coding of Coefficients. *Document of Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16*, Fairfax, Virginia, USA, May 6-10, 2002

Brunello D., Calvagno G., Mian G. A., & Rinaldo R. (2003). Lossless Compression of Video Using Temporal Information. *IEEE Transactions on Image Processing*, Vol.12, No.2, (February 2003), pp. 132-139, ISSN 1057-7149

Fraunhofer Institute for Telecommunications Heinrich Hertz Institute. Joint Video Team, *H.264/AVC Reference Software Version 13.2* [Online], January 2011, Available from: http://iphome/hhi.de/shehring/tml/download/old_jm/jm13.2.zip

Gallager R. G. & Van Voorhis D. C. (1975). Optimal source codes for geometrically distributed integer alphabets. *IEEE Transactions on Information Theory*, Vol.21, No.2, (March 1975), pp. 228–230, ISSN 0018-9448

Heo J., Kim S.H., & Ho. Y.S. (2010). Improved CAVLC for H.264/AVC Lossless Intra Coding. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.20, No.2, (February 2010), pp. 213–222, ISSN 1051-8215

Lee Y.-L., Han K.-H., & Lim S.-C. (2005). Lossless Intra Coding for Improved 4:4:4 Coding in H.264/MPEG-4 AVC. *Document of Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16*, Poznan, Poland, July 24-29, 2005

Lee Y.-L., Han K.-H., & Sullivan G. (2006). Improved lossless intra coding for H.264/MPEG-4 AVC. *IEEE Transactions on Image Processing*, Vol.15, No.9, (September 2006), pp. 2610–2615, ISSN 1057-7149

Luthra A., Sullivan G., & Wiegand T. (2003). Introduction to the special issue on the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.13, No.7, (July 2003), pp. 557–559, ISSN 1051-8215

Malvar H., Hallapuro A., Karczewicz M., & Kerofsky L. (2003). Low-Complexity transform and quantization in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.13, No.7, (July 2003), pp. 598–603, ISSN 1051-8215

Marpe D., Schwarz H., & Wiegand T. (2003). Context-based adaptive binary arithmetic coding in the H.264/AVC video compression. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.13, No.7, (July 2003), pp. 620–636, ISSN 1051-8215

Richardson I. E. G. (2003). *H.264 and MPEG-4 video compression*. New York: Wiley, ISBN 0-470-84837-5, England

Salomon D. (2007). *Variable-length Codes for Data Compression*. Springer, ISBN 978-1-84628-958-3, England

Sayood K. (2006). *Introduction to Data Compression*, CA: Morgan Kaufmann, ISBN 978-0-12-620862-7, USA

Sullivan G. & Wiegand T. (1998). Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, Vol. 15, (Nov. 1998), pp. 74-90, ISSN 1053-5888

Sullivan G., McMahon T., Wiegand T., Marpe D., & Luthra A. (2004). Draft Text of H.264/AVC Fidelity Range Extensions Amendment. *Document of Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16*, Redmond, WA, USA, July 17-23, 2004

Sullivan G., Topiwala P., & Luthra A. (2004). The H.264/AVC advanced video coding standard: Overview and introduction to the fidelity range extensions. *Proceedings of SPIE Conference, Special Session on Advances in the New Emerging Standard: H.264/AVC*, ISBN 978-1-59593-695-0, Denver, Colorado, USA, August 2004

Sullivan G. & Wiegand T. (2005). Video compression–from concepts to the H.264/AVC standard. *Proceedings of the IEEE*, Vol.93, No.1, (January 2005), pp. 18–31, ISSN 0018-9219

Sun S. (2002). Intra Lossless Coding and QP Range Selection. *Document of Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16*, Fairfax, Virginia, USA, May 6-10, 2002

Teuhola J. (1978). A compression method for clustered bit-vectors. *Information Processing Letters*, Vol.7, (October 1978), pp. 308–311, ISSN 0020-0190

Weinberger M. J., Seroussi G., & Sapiro G. (2000). The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS. *IEEE Transactions on Image Processing*, Vol.9, No.8, (August 2000), pp. 1309–1324, ISSN 1057-7149

Wiegand T., Sullivan G., Bjøntegaard G., & Luthra A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.13, Vo.7, (July 2003), pp. 560–576, ISSN 1051-8215