Improved CAVLC for H.264/AVC Lossless Intra-Coding

Jin Heo, *Student Member, IEEE,* Seung-Hwan Kim, and Yo-Sung Ho, *Senior Member, IEEE*

***Abstract*—Context-based adaptive variable length coding (CAVLC) for the H.264/advanced video coding (AVC) standard was originally designed for lossy video coding, and as such does not yield adequate performance for lossless video coding. In this paper, we propose an improved CAVLC for lossless intra-coding by considering the statistical differences in residual data between lossy and lossless coding. From experimental results, we confirm that the proposed method provides approximately 9% bit saving in terms of a compression ratio compared with the current H.264/AVC fidelity range extensions high profile.**

***Index Terms*—Context-based adaptive variable length coding (CAVLC), H.264/AVC, intra-coding, lossless video coding.**

I. Introduction

THE latest video coding standard, H.264/advanced video coding (AVC), was developed through the Joint Video Team (JVT) based on standardization of the International Telecommunications Union-Telecommunication (ITU-T) Video Coding Experts Group and the International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) Moving Picture Experts Group. Currently, due to its high-compression performance H.264/AVC has become a promising video compression standard for a wide range of applications, including multimedia streaming and video conferencing [1]–[4].

To date, however, the H.264/AVC standard has been developed by mainly focusing on lossy coding; lossless video coding is also important in several application areas such as source distribution, digital cinema, and medical imaging. Hence, to provide improved functionality for lossless video coding, the H.264/AVC standard first included a so-called *pulse-code modulation* (PCM) macroblock coding mode, where the values of the original image samples are sent directly without prediction, transform, and quantization. Actually, by imposing a minimum upper bound on the number of bits that can be used to represent a macroblock with sufficient accuracy, the PCM mode was designed to be simple, though this aspect also reduced its coding efficiency [5].

After finalizing the standardization of the first version of H.264/AVC, JVT developed extensions to the original standard known as the fidelity range extensions (FRExt) [6], [7]. When developing the FRExt amendment, it was decided that a more effective means of lossless coding was desirable for most demanding of applications. Therefore, FRExt also includes a transform-bypass [8] lossless mode that employs two main coding processes, entropy coding and prediction, which were not previously used in the PCM macroblock mode. To further enhance the coding performance for lossless coding, however, more efficient coding techniques for prediction and entropy coding are still required.

In the meantime, instead of developing a block-based intra-prediction, a new intra-prediction method called sample-wise *differential pulse-code modulation* (DPCM) [5], [9], [10] was introduced 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 has been shown to provide better compression performance without incurring a major increase in the computational complexity, and was subsequently adopted as a part of the new draft amendment for the H.264/AVC standard [11].

For lossless intra-coding in FRExt, the original sample values are coded by two main coding processes, intra-prediction and entropy coding, because transform and quantization are not used. In other words, for lossless coding, the sample values obtained from intra-prediction are directly coded by the entropy coder; conversely, for lossy coding, quantized transform coefficients [12] are entered into the entropy coder. Hence, there are significant statistical differences between lossy and lossless coding. Therefore, in order to design more efficient entropy coding technique for lossless coding, we need to modify the conventional entropy coder in H.264/AVC.

In this paper, we have tried to improve the performance of context-based adaptive variable length coding (CAVLC) [13], [14] for lossless intra-coding. Previously, CAVLC was unable to provide optimum coding performance for lossless video coding because it was designed primarily for use in lossy video coding. Thus, we propose a more efficient CAVLC design for lossless intra-coding.

The baseline entropy coding method uses the zero-order Exp-Golomb code [15] for all syntax elements with the

Manuscript received October 30, 2008; revised April 6, 2009. First version published September 1, 2009; current version published February 5, 2010. 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-2009-(C1090-0902-0017)]. This paper was recommended by Associate Editor L. Chen.

J. Heo and Y.-S. Ho are with the Department of Information and Communications, Gwangju Institute of Science and Technology (GIST), Gwangju 500-712, Korea (e-mail: jinheo@gist.ac.kr; hoyo@gist.ac.kr).

S.-H. Kim is with the Department of Electrical Engineering, University of Southern California (USC), Los Angeles, CA 90089 USA (email: kimseung@usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2009.2031392

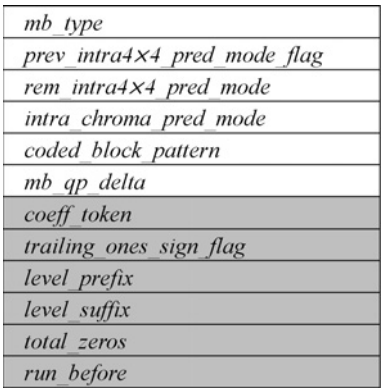

Fig. 1. Syntax elements for a macroblock.

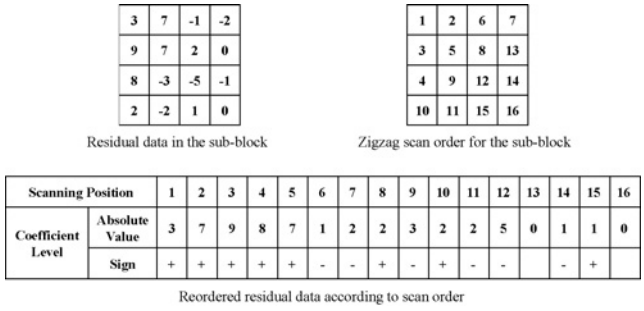

Fig. 2. Zigzag scan order for the sub-block.

exception of the residual data, which are coded using CAVLC. Fig. 1 shows the syntax elements employed in CAVLC for a macroblock (MB); here, the gray shaded syntax elements are used to encode residual data in the macroblock [1]. Next, by considering the statistics of the residual data, we modified or removed the coding method for the corresponding syntax elements. Note that our research goal is to improve the coding performance of CAVLC, which can be easily applied to H.264/AVC lossless intra-coding by modifying some semantics and decoding processes, without requiring the addition of other syntax elements to the H.264/AVC standard. From our experimental results, we found that the proposed method provides approximately 9% bit saving in terms of compression ratio compared with the current H.264/AVC FREext high profile.

The rest of this paper is organized as follows. In the next section, we briefly review the coding structure of CAVLC for residual data. In Section III, we propose an improved CAVLC for lossless intra-coding. In Section IV, the coding performance of the proposed coding technique is compared with other well-known lossless coding methods. Finally, the paper is completed with our conclusions presented in Section V.

II. Overview of CAVLC in H.264/AVC

In this section, we review CAVLC in H.264/AVC. CAVLC is employed to encode residual data, zigzag scanned quantized transform coefficients, for a 4×4 sub-block. Fig. 2 illustrates the zigzag scan order for the 4×4 sub-block.

In. H.264/AVC, CAVLC was designed to take advantage of several characteristics of residual data in lossy coding:

TABLE I
CAVLC Syntax Elements for Residual Data

Syntax Elements	Description
coeff_token	Encodes the number of nonzero coefficients and trailing ones
trailing_ones_sign_flag	Sign of trailing one value
level_prefix	First part of code for nonzero coefficient
level_suffix	Second part of code for nonzero coefficient
total_zeros	Encodes the total number of zeros occurring after the first nonzero coefficient
run_before	Encodes the number of zeros preceding each nonzero coefficient

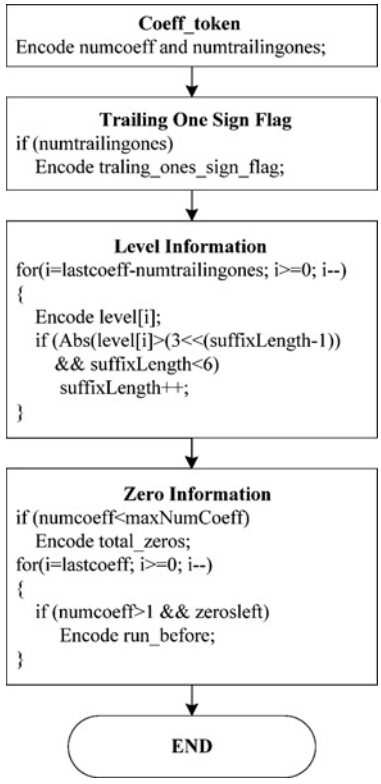

Fig. 3. Encoding structure of CAVLC for residual coding.

1) after transform and quantization, sub-blocks typically contain many zeros, especially in high-frequency regions; 2) the level of the highest nonzero coefficients tends to be as small as one; and 3) the level of nonzero coefficients tends to be larger toward the low-frequency regions. Therefore, taking into consideration the above characteristics, CAVLC employs the syntax elements *coeff_token*, *trailing_ones_sign_flag*, *level_prefix*, *level_suffix*, *total_zeros*, and *run_before* to efficiently encode the residual data. The specific function of each syntax element is described in Table I. The encoding structure of CAVLC for a sub-block using the given syntax elements is depicted in Fig. 3.

The detailed coding procedure of CAVLC is as follows.

1) *Step 1:* Both the total number of nonzero coefficients (*numcoeff*) and the number of trailing ones (*numtrailingones*) are encoded using a combined codeword (*coeff_token*).

TABLE II
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

2) *Step 2:* The sign of each trailing one is encoded using a one bit codeword in reverse order (*trailing_ones_sign_flag*).
3) *Step 3:* The absolute value of the level of each remaining nonzero coefficient is encoded in reverse order using one of the seven predefined VLC tables (from *Lev-VLC0* to *Lev-VLC6*) and the sign information is encoded in the same way as Step 2 (*level*).
4) *Step 4:* The number of all zeros before the last nonzero coefficient is encoded (*total_zeros*).
5) *Step 5:* The number of consecutive zeros preceding each nonzero coefficient is encoded in reverse order (*run_before*).

A. Encode the Number of Nonzero Coefficients and the Number of Trailing Ones

The syntax element *coeff_token* encodes both *numcoeff* and *numtrailingones* in the sub-block. A trailing one is one of up to three consecutive nonzero coefficients at the end of the scan of nonzero 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 nonzero coefficients in the previously coded upper and left sub-blocks. If both the upper and left sub-blocks are available, the number of predicted nonzero coefficients in the current sub-block is calculated by

$$N = round\left(\frac{N_U + N_L}{2}\right) \tag{1}$$

where N represents the number of predicted nonzero coefficients in the current sub-block, and N_U and N_L are the number of nonzero coefficients in the upper and left previously encoded sub-blocks, respectively. Note that 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. Thus, based on the parameter N, an appropriate VLC table for the current sub-block is selected from Table II.

B. 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 one bit codeword in reverse order. If the sign information is positive (+), the *trailing_ones_sign_flag* is equal to zero. Conversely, if the sign information is negative (−), the *trailing_ones_sign_flag* is equal to one.

TABLE III
THRESHOLDS FOR DETERMINING WHETHER TO INCREMENT VLC TABLE

VLC Table for Level Coding	Threshold to Increment VLC Table
Lev-VLC0	0
Lev-VLC1	3
Lev-VLC2	6
Lev-VLC3	12
Lev-VLC4	24
Lev-VLC5	48
Lev-VLC6	>48

C. Encode the Levels

The level (sign and magnitude) of each remaining nonzero 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 VLC table from among seven VLC tables (Table III), with selection of the VLC table dependent on the magnitude of each recently encoded level. The choice of VLC table is adapted as follows.

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

D. Encode the Total Number of Zeros and Each Run of Zeros

After the encoding process for level information, zeros remain, and each run of zeros is coded to indicate the position of each zero coefficient. For this task, CAVLC employs two syntax elements, *total_zeros* and *run_before*, where the syntax element *total_zeros* indicates the total number of zero coefficients located before the last nonzero coefficient. After encoding *total_zeros*, the position of each zero coefficient is then encoded. The syntax element *run_before* indicates the number of consecutive zero coefficients between the nonzero coefficients and is encoded in reverse order.

Note that *zerosleft* indicates the number of zeros that has not yet been encoded. The syntax element *run_before* is encoded at each nonzero 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) nonzero coefficient.

III. PROPOSED METHOD

In this section, by considering the statistical differences in residual data between lossy and lossless coding, we introduce an improved CAVLC for lossless intra-coding. More details of the statistical differences will be described in the following section.

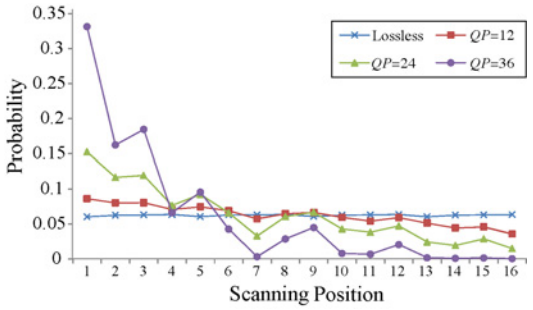

Fig. 4. Probability distribution of existence of nonzero coefficients according to the scanning position ('*Foreman*', QCIF).

TABLE IV
OCCURRENCE PROBABILITY DISTRIBUTION OF TRAILING ONES

Sequence	*QP*			
	0 (Lossless)	12	24	36
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
City_corr	0.21003	0.81396	0.89141	0.95066
Crowdrun	0.18128	0.76725	0.92403	0.94784
Parkrun	0.14710	0.44303	0.78293	0.95821
Breeze	0.07290	0.25576	0.63821	0.94739

A. Analysis of the Statistical Characteristics of Residual Data in Lossless Coding

In lossy coding, residual data represent quantized transform coefficients. The statistical characteristics of residual data in lossy coding are as follows. In a given sub-block, the probability of existence of a nonzero coefficient is likely to decrease as the scanning position increases. Moreover, the absolute value of a nonzero coefficient tends to decrease as the scanning position increases. Hence, the occurrence probability of a trailing one is relatively high.

In lossless coding, residual data do not represent quantized transform coefficients, but rather the differential pixel values between the original and intra-predicted pixel values. Therefore, the statistical characteristics of the residual data in lossless coding are as follows. First, the probability of existence of a nonzero coefficient is independent of the scanning position, and the number of nonzero coefficients is generally large, compared with those in lossy coding. Second, the absolute value of a nonzero coefficient 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; therefore, the trailing one does not need to be treated as a special case of encoding.

Fig. 4 shows the probability distribution of existence of nonzero coefficients according to the scanning position. As expected, a significant difference can be seen in the statistics between the residual data of lossy and lossless coding.

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

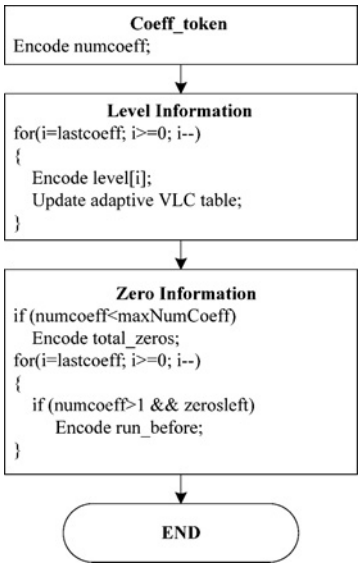

Fig. 5. Encoding structure of the proposed method for residual data coding.

Therefore, in order to reflect the above statistical characteristics of residual data more accurately, we propose a more efficient CAVLC for lossless intra-coding in H.264/AVC by modifying the relevant coding parts of CAVLC.

In Fig. 5, we depict the encoding structure of the proposed method for residual data coding. The coding procedure of the proposed CAVLC can be summarized in the following steps.

1) *Step 1:* Encode the total number of nonzero coefficients.
2) *Step 2:* Encode the level of all nonzero coefficients.
3) *Step 3:* Encode the number of all zeros before the last nonzero coefficient.
4) *Step 4:* Encode the number of consecutive zeros preceding each nonzero coefficient.

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

B. Coding the Number of Nonzero Coefficients

In this step, we encode the total number of nonzero coefficients (*numcoeff*) but do not consider the number of trailing ones (*numtrailingones*). In CAVLC, the corresponding VLC table is selected based on the predicted *numcoeff* obtained from equation (1); further details have already been explained in Section II-A. Note that if the predicted *numcoeff* is larger than seven, the FLC table is selected, as described in Table II.

In lossless coding, the FLC table is most often selected because *numcoeff* is generally larger than seven, as shown in Table V. 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*, respectively; since *numtrailingones* does not need to be considered; only 4 bits for *numcoeff* remain. However, instead of using the FLC table, which uniformly

TABLE V
AVERAGE NUMBER OF NONZERO COEFFICIENTS IN A SUB-BLOCK

Sequence	*QP*			
	0 (Lossless)	12	24	36
Foreman	13.7457	7.8073	3.3253	1.0017
Silent	14.1030	8.2938	3.3532	0.8971
Paris	13.8449	8.0153	4.0863	1.6657
Mobile	14.6338	10.9796	6.6945	2.4879
City_corr	14.4775	6.5353	3.3869	0.9449
Crowdrun	14.9614	10.4297	4.0696	1.3231
Parkrun	14.5677	11.6905	5.9627	1.3494
Breeze	14.8397	13.1772	7.9119	1.4960

TABLE VI
CODEWORD TABLE FOR '*numcoeff*'

numcoeff	Codeword	
	Check Bit	Bits for *numcoeff*
0	1	1111
1	1	0000
2	1	0001
3	1	0010
4	1	0011
5	1	0100
6	1	0101
7	1	0110
8	1	0111
9	1	1000
10	1	1001
11	1	1010
12	1	1011
13	0	00
14	0	01
15	0	10
16	0	11

assigns 4 bits for all *numcoeff*, we designed a simple but effective VLC table according to the statistics of *numcoeff* in lossless coding.

In our proposed VLC table, *numcoeff* from 1 to 12 and 13 to 16 have 4-bit and 2-bit codewords, respectively. In order to avoid ambiguity at the decoder, we insert a check bit into the prefix of each codeword; details of the codewords are further described in Table VI.

C. Level Coding

In level coding, the absolute level value of each nonzero coefficient (*abs_level*) is adaptively encoded by a selected VLC table from among the seven predefined VLC tables (*Lev-VLC0* to *Lev-VLC6*) in reverse scanning order. Each VLC table is designed to encode efficiently in a specified range of *abs_level*, as described in Table III. As previously mentioned, selection of the 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 VLC table number monotonically increases according to the previously encoded *abs_level*.

However, *abs_level* in lossless coding is independent of the scanning position, as shown in Fig. 6. Therefore, we designed an adaptive method for VLC table selection that

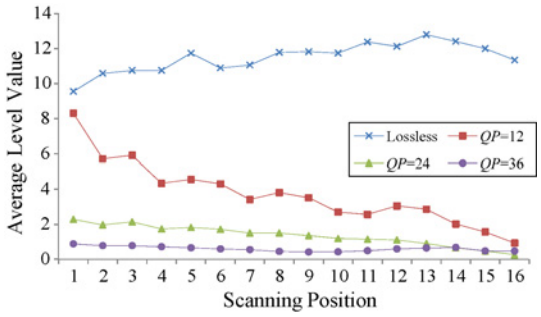

Fig. 6. Distribution of average absolute level value according to the scanning position ('*Tempete*', CIF).

can decrease or increase according to the previously encoded *abs_level*.

In lossy coding, CAVLC typically determines the smallest VLC table in the range of possible 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_level* does not necessarily increase at lower frequencies—we cannot assume that the next *abs_level* is larger than the current *abs_level*. Therefore, the VLC table for each *abs_level* should be selected by considering the previously encoded *abs_levels* because we cannot predict whether or not the next *abs_level* will increase.

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

$$T(abs\_level_i) = \frac{1}{a_i + 1}\{a_i \cdot avg_i + abs\_level_i\} \tag{2}$$

$$a_i = \begin{cases} 0, & i = lastcoeff \\ 1, & i = lastcoeff - 1, lastcoeff - 2 \\ 2, & \text{otherwise} \end{cases} \tag{3}$$

$$avg_i = \frac{1}{(lastcoeff - i + 1)}\left\{\sum_{k=lastcoeff}^{i} abs\_level_k\right\} \tag{4}$$

where a_i and abs_level_i are the weighting coefficient and *abs_level* value, respectively, where both values are related to the current scanning position i. In addition, $T(abs\_level_i)$ and *lastcoeff* represent the threshold value for selecting the corresponding VLC table used to encode the next *abs_level* [$(i-1)$th *abs_level*] and the scanning position number of the last nonzero coefficient, respectively. Note that *abs_level* is encoded in reverse order. In Table VII, we represent the VLC table for level coding according to $T(abs\_level_i)$. From extensive experiments on lossless intra-coding using various test sequences, we could determine these optimal threshold values.

In Fig. 6, we can note 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

TABLE VII
NEW THRESHOLDS FOR DETERMINING THE VLC TABLE

VLC Table for *abs_level*	$T(abs\_level_i)$
Lev-VLC0	0
Lev-VLC1	2
Lev-VLC2	4
Lev-VLC3	9
Lev-VLC4	19
Lev-VLC5	39
Lev-VLC6	–

TABLE VIII
AVERAGE ABSOLUTE VALUE OF THE LAST NONZERO COEFFICIENT FOR THE SUB-BLOCKS

Sequence	*QP*			
	0 (Lossless)	12	24	36
Foreman	9.2097	2.2939	1.8902	1.8881
Silent	8.3415	2.2446	1.9687	2.0028
Paris	10.5532	2.6227	2.1445	1.8243
Mobile	16.3748	3.0935	2.1962	1.7870
City_corr	6.9376	1.2654	1.1340	1.0572
Night	8.8483	1.3609	1.0906	1.0611
Parkrun	9.0530	2.0832	1.4104	1.1251
Crowdrun	8.8483	1.3609	1.0906	1.0611

coefficient in lossy coding, and it is likely to be small. However, in lossless coding, the last scanned *abs_level* is not small enough to use either *Lev-VLC0* or *Lev-VLC1*. Table VIII represents the average absolute value of the last scanned *abs_level* for the sub-blocks. In Table VIII, the average absolute value of the last scanned *abs_level* in lossless coding is larger than that in lossy coding. The average absolute value of the last scanned *abs_level* in the sub-blocks is approximately 10.09 in lossless coding. Based on this value, we adjusted the initial VLC table for level coding. The modified 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 *abs_level*.
4) Update the VLC table by considering the previously encoded *abs_levels* and new threshold for each VLC table.

D. Example of the Proposed Encoding Process

In Fig. 7, we present an example of the entire encoding process for the proposed method. In this example, we can observe that the *coeff_token*, *trailing_one_sign_flag*, and level information coding are modified. The coding procedure of the proposed CAVLC is as follows.

1) *Step 1:* Encode *numcoeff* = 14.
2) *Step 2:* Encode the level of all nonzero coefficients in reverse order.
3) *Step 3:* Encode *total_zeros* = 1.
4) *Step 4:* Encode *run_before* = 0 with *zerosleft* = 1 and *run_before* = 1 with *zerosleft* = 1 in reverse order.

Scanning Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Coefficient Level	3	7	9	8	7	-1	-2	2	-3	2	-2	-5	0	-1	1	0
Encoding																
coeff_token	total number of non-zero coefficients=14															
Level coding	1, -1, -5, -2, 2, -3, 2, -2, -1, 7, 8, 9, 7, 3															
total_zeros	number of consecutive zeros preceding the last non-zero coefficient=1															
run_before	zerosleft=1; run_before=0															
run_before	zerosleft=1; run_before=1															

Fig. 7. Example of the proposed encoding process.

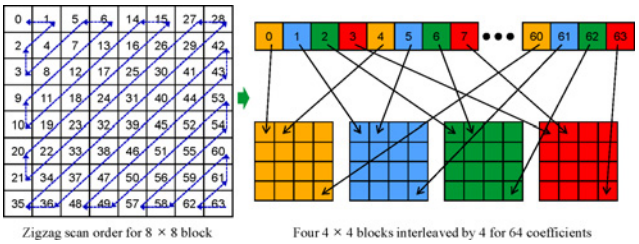

Fig. 8. 8×8 intra-block CAVLC.

E. 8×8 Intra-Coding Mode

H.264/AVC supports three different types of intra-coding modes: 4×4, 8×8, and 16×16 intra-mode. Among these modes, only 8×8 intra-mode employs 8×8 block CAVLC including four 4×4 block CAVLC and the coefficients in each 4×4 block are rearranged as shown in Fig. 8. Therefore, for 8×8 block CAVLC, our proposed 4×4 block CAVLC is used and coefficients in each 4×4 block are arranged in the same way as shown in Fig. 8.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this paper, an improved CAVLC for lossless intra-coding has been presented. In order to verify efficiency of the proposed method, we performed experiments on several test sequences of YUV420 and 8 bits per pixel (b/p) format with quarter common intermediate format (QCIF), common intermediate format (CIF), and high-definition (HD) resolutions. Moreover, we experimented on several RGB444 and 10 b/p format test sequences with HD resolution. We implemented our proposed method in the H.264/AVC reference software version JM 13.2 [16]. The encoding parameters for the reference software were as follows.

1) *ProfileIDC* = 244 (High 4:4:4).
2) *IntraPeriod* = 1 (only intra-coding).
3) *QPISlice* = 0.
4) *SymbolMode* = 0 (CAVLC is used).
5) *QPPrimeYZeroTransformBypassFlag* = 1 (lossless).

The proposed method consists of three parts.

1) *Method I:* Modify *coeff_token* + remove *trailing_one_sign_flag*.
2) *Method II:* Modify level information coding.
3) *Method III: Method I* + *Method II*.

Note that these proposed methods were applied to H.264/AVC lossless intra-coding by modifying the semantics

TABLE IX
COMPARISON OF COMPRESSION RATIO AND SAVING BITS FOR H.264 LOSSLESS INTRA-CODING, JPEG-LS(LOSSLESS), MOTION JPEG2000 LOSSLESS (M-JP2K), AND PROPOSED METHODS WITH QCIF AND CIF RESOLUTION SEQUENCES

Image	Original Image Size (bits)	Method	Total Bits (bits)	Compression Ratio	Saving Bits (%)
News (QCIF, 8 b/p) YUV420, 300 frames	91 238 400	H.264/AVC (CAVLC)	39 829 112	2.29074653	0
		JPEG-LS	38 493 000	2.37025952	3.35461
		M-JP2K	44 094 160	2.06917197	−10.70837
		Method I	38 188 480	2.38916029	4.11918
		Method II	38 282 712	2.38327943	3.88259
		Method III	36 684 720	2.48709545	**7.89471**
Container (QCIF, 8 b/p) YUV420, 300 frames	91 238 400	H.264/AVC (CAVLC)	40 415 808	2.25749291	0
		JPEG-LS	40 503 200	2.25262201	−0.21623
		M-JP2K	44 423 256	2.05384314	−9.91555
		Method I	38 778 432	2.35281121	4.05133
		Method II	39 022 216	2.33811427	3.44814
		Method III	37 443 104	2.43672106	**7.35530**
Foreman (QCIF, 8 b/p) YUV420, 300 frames	91 238 400	H.264/AVC (CAVLC)	41 638 656	2.19119464	0
		JPEG-LS	43 903 664	2.07815001	−5.43968
		M-JP2K	48 250 840	1.89091837	−15.87992
		Method I	39 970 056	2.28266881	4.00733
		Method II	39 925 312	2.28522698	4.11479
		Method III	38 306 832	2.38177879	**8.00176**
Silent (QCIF, 8 b/p) YUV420, 300 frames	91 238 400	H.264/AVC (CAVLC)	44 395 976	2.05510517	0
		JPEG-LS	44 656 200	2.04312950	−0.58614
		M-JP2K	47 552 944	1.91866985	−7.11093
		Method I	42 721 256	2.13566755	3.77223
		Method II	42 013 136	2.17166364	5.36724
		Method III	40 373 360	2.25986641	**9.06077**
Paris (CIF, 8 b/p) YUV420, 300 frames	364 953 600	H.264/AVC (CAVLC)	179 800 256	2.02977242	0
		JPEG-LS	179 265 368	2.03582880	0.29749
		M-JP2K	196 161 712	1.86047315	−9.09980
		Method I	173 222 520	2.10684846	3.65836
		Method II	169 306 776	2.15557586	5.83619
		Method III	162 905 992	2.24027119	**9.39613**
Mobile (CIF, 8 b/p) YUV420, 300 frames	364 953 600	H.264/AVC (CAVLC)	224 186 128	1.62790447	0
		JPEG-LS	231 103 384	1.57917895	−3.08550
		M-JP2K	240 223 216	1.51922701	−7.15347
		Method I	217 530 848	1.67770964	2.96864
		Method II	200 478 320	1.82041430	10.57506
		Method III	193 960 504	1.88158719	**13.48238**
Tempete (CIF, 8 b/p) YUV420, 260 frames	316 293 120	H.264/AVC (CAVLC)	167 897 968	1.88384126	0
		JPEG-LS	166 747 848	1.89683479	0.68501
		M-JP2K	175 970 768	1.79741853	−4.80816
		Method I	162 071 152	1.95156952	3.47045
		Method II	155 229 904	2.03757853	7.54510
		Method III	149 562 648	2.11478684	**10.92051**
Average		H.264/AVC (CAVLC)		2.04800820	0
		JPEG-LS		2.03657194	−0.71292
		M-JP2K		1.87281743	−9.23946
		Method I		2.12806221	3.72107
		Method II		2.17026472	5.82416
		Method III		2.25744385	**9.44451**

and decoding processes, without adding any syntax elements to the H.264/AVC standard. The proposed method is implemented on top of the previous sample-wise DPCM prediction method, and it further enhanced the coding efficiency for lossless intra-coding in H.264/AVC. To verify efficiency of the proposed method, we have performed two kinds of experiments. In the first experiment, seven YUV420 format test sequences with QCIF and CIF resolutions are tested as shown in Table IX and we compared several well-known lossless coding techniques, including JPEG-LS (lossless) [17], [18], Motion JPEG2000 lossless [19] with our proposed method. For the Motion JPEG2000 comparison, we used "Jasper-1.701.0" software downloaded from the JPEG site [20]. In the second experiment, four YUV420 and two RGB444 format test sequences with HD resolution are tested as shown in Table X. Comparisons were made in terms of bit-rate percentage differences and compression ratio differences with respect to H.264/AVC using sample-wise DPCM. These changes were

TABLE X
COMPARISON OF COMPRESSION RATIO AND SAVING BITS FOR H.264 LOSSLESS INTRA-CODING AND PROPOSED METHODS WITH HD RESOLUTION SEQUENCES

Image	Original Image Size (bits)	Method	Total Bits (bits)	Compression Ratio	Saving Bits (%)
City_corr (1280 × 720, 8 b/p) YUV420, 100 frames	1 105 920 000	H.264/AVC (CAVLC)	517 521 408	2.13695508	0
		Method I	497 521 752	2.22285759	3.86451
		Method II	498 239 584	2.21965503	3.72580
		Method III	478 757 784	2.30997811	**7.49025**
Night (1280 × 720, 8 b/p) YUV420, 100 frames	1 105 920 000	H.264/AVC (CAVLC)	452 440 816	2.44434180	0
		Method I	432 214 144	2.55873163	4.47057
		Method II	446 389 160	2.47747952	1.33756
		Method III	426 810 984	2.59112357	**5.66479**
Parkrun (1920 × 1080, 8 b/p) YUV420, 100 frames	2 488 320 000	H.264/AVC (CAVLC)	1 210 958 808	2.05483455	0
		Method I	1 165 454 520	2.13506401	3.75771
		Method II	1 143 925 528	2.17524650	5.53555
		Method III	1 100 358 800	2.26137147	**9.13326**
Crowdrun (1920 × 1080, 8 b/p) YUV420, 100 frames	2 488 320 000	H.264/AVC (CAVLC)	1 177 625 784	2.11299721	0
		Method I	1 131 576 888	2.19898447	3.91032
		Method II	1 124 143 280	2.21352566	4.54155
		Method III	1 080 073 896	2.30384237	**8.28378**
Breeze (1920 × 1080, 10 b/p) RGB444, 50 frames	3 110 400 000	H.264/AVC (CAVLC)	2 589 657 000	1.20108570	0
		Method I	2 525 544 376	1.23157606	2.47572
		Method II	2 287 913 472	1.35949197	11.65187
		Method III	2 227 628 200	1.39628328	**13.97980**
Man_in_restaurant (1920 × 1080, 10 b/p) RGB444, 50 frames	3 110 400 000	H.264/AVC (CAVLC)	2 264 348 968	1.37363986	0
		Method I	2 203 168 544	1.41178486	2.70190
		Method II	2 159 022 456	1.44065199	4.65151
		Method III	2 097 439 464	1.48295102	**7.37119**
Average		H.264/AVC (CAVLC)		1.88730903	0
		Method I		1.95983310	3.53012
		Method II		1.98100845	5.24064
		Method III		2.05759164	**8.65385**

calculated as follows:

$$\Delta Saving\,Bits(\%) = \frac{Bit\text{-}rate_{H.264/AVC} - Bit\text{-}rate_{Method}}{Bit\text{-}rate_{H.264/AVC}} \times 100 \tag{5}$$

$$Compression\,Ratio = \frac{Original\ image\ size}{Bit\text{-}rate_{Method}}. \tag{6}$$

In Table IX, we confirm that the proposed method provided the best coding performance compared with several well-known lossless coding techniques, such as JPEG-LS and Motion JPEG2000 in lossless intra-coding. In addition, the proposed method provided better coding performance compared with the conventional CAVLC—by approximately 9.4% with QCIF and CIF resolutions. Table X shows experimental results for six HD resolution test sequences. For HD resolution, the proposed method provided approximately 8.6% bit saving compared with the conventional CAVLC.

In order to evaluate the influence of coding efficiency of the proposed method, we compare coding bits of the proposed CAVLC and those of the ideal CAVLC which selects the correct *Lev-VLC* table for encoding the absolute level value of each nonzero coefficient (*abs_level*) because most coding bits are consumed on level coding in lossless coding. From extensive experiments on lossless intra-coding with various test sequences, we have observed that bit savings of the proposed CAVLC and the ideal CAVLC are approximately 9% and 20%, compared with the conventional CAVLC, respectively. This result shows that there is still some room for reducing coding bits by modifying the level coding part more efficiently.

Here, we discuss scanning patterns for lossless coding. In lossy coding, coding performance can be changed according to various scanning patterns because the residual data are quantized transform coefficients (QTCs) and the statistical distribution of QTCs is highly skewed on small level values (including a lot of zeros) as depicted in Figs. 4 and 6. Hence, if we find an appropriate scanning pattern, we can enhance coding performance by arranging QTCs according to their amplitude levels. However, in lossless coding, the distribution of the amplitude (or significance) of residual signal is quite wide and also shown to be independent of the scanning position, as shown in Figs. 4 and 6. Therefore, theoretically, there is no scanning order which can provide even better coding efficiency and we have also confirmed the fact by performing extensive experiments by using various scanning patterns including the zigzag scanning order. Finally, it is not easy to find the best scanning pattern which can be widely accepted for lossless coding. However, there could be a potential

work for finding the best scanning pattern for each residual block.

In recent research works [21], [22], a selective residual coding algorithm in the spatial and frequency domain has been introduced because the prediction residual signal may have low-spatial-correlation and transform coding could cause a loss of coding efficiency. Specifically, they select an appropriate residual coding algorithm in the spatial and frequency domain based on rate-distortion (RD). In addition, they newly design an adaptive scanning order for the spatial domain coding. In this contribution [21], the scanning pattern in the spatial domain was to be determined by the magnitude of the gradient of the prediction (motion compensated) signal. However, coding performance of the previous research works [21], [22] only guaranteed only if there is a dependency in the magnitude of the gradient between prediction and residual signal. The research works provide good coding performance in the normal coding condition ($QP > 20$). However, in near lossless ($QP < 10$) and lossless coding condition, coding performance is not guaranteed because the residual signal is quite random and there is not so much dependency in the magnitude of the gradient between prediction and residual signal. Therefore, there could be a potential work for finding the best scanning pattern for each residual block; however, we need to consider the trade-off between the additional coding bits required to indicate the scanning pattern and the bit-savings obtained by using the optimal scanning pattern.

Finally, 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 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 [17]. In H.264/AVC, a similar DPCM is employed to predict the original pixel value, but it employs rate-distortion optimization [23] 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 are 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 context-based adaptive binary arithmetic coding [24] which are mainly designed for discrete cosine transform-based lossy coding. As a result, JPEG-LS and H.264/AVC lossless mode provide quite similar coding performance. In this paper, we have proposed a new entropy coding algorithm that can adapt to the lossless coding condition.

V. Conclusion

In this paper, we proposed an improved CAVLC for lossless intra-coding, based on a traditional CAVLC. Considering the statistical differences in residual data between lossy and lossless coding, we modified the CAVLC encoding mechanism. Experimental results show that the proposed method provides approximately 9% bit saving, compared with the H.264/AVC FREXt high profile.

References

[1] *Editors' draft revision to ITU-T Rec. H.264 | ISO/IEC 14496-10 Advanced Video Coding – in preparation for ITU-T SG 16 AAP Consent*, document JVT-AD205.doc, Joint Video Team of ISO/IEC 14496-10 AVC, ISO/IEC JTC1/SC29/WG11, and ITU-T Q.6/SG16, Feb. 2009.

[2] A. Luthra, G. J. Sullivan, and T. Wiegand, "Introduction to the special issue on the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 557–559, Jul. 2003.

[3] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[4] G. J. Sullivan and T. Wiegand, "Video compression: From concepts to the H.264/AVC standard," *Proc. IEEE*, vol. 93, no. 1, pp. 18–31, Jan. 2005.

[5] Y.-L. Lee, K.-H. Han, and G. J. Sullivan, "Improved lossless intra-coding for H.264/MPEG-4 AVC," *IEEE Trans. Image Process.*, vol. 15, no. 9, pp. 2610–2615, Sep. 2006.

[6] *Draft Text of H.264/AVC Fidelity Range Extensions Amendment to ITU-T Rec. H.264*, document JVT-L047.doc, Joint Video Team of ISO/IEC 14496-10 AVC, ISO/IEC JTC1/SC29/WG11, and ITU-T Q.6/SG16, Jul. 2004.

[7] G. J. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC standard: Overview and introduction to the fidelity range extensions," in *Proc. SPIE Conf., Special Session Adv. New Emerg. Standard: H.264/AVC*, Denver, CO, Aug. 2004, pp. 454–474.

[8] *Intra-Lossless Coding and QP Range Selection*, document JVT-C023.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, May 2002.

[9] *Lossless Intra-Coding for Improved 4:4:4 Coding in H.264/MPEG-4 AVC*, document JVT-P016.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Jul. 2005.

[10] *Complexity of the Proposed Lossless Intra*, document JVT-Q035r1.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Oct. 2005.

[11] *Draft Text of H.264/AVC Advanced 4:4:4 Profile Amendment to ITU-T Rec. H.264 |ISO/IEC 14496-10 AVC*, document JVT-Q209.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q.6/SG16, Oct. 2005.

[12] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 598–603, Jul. 2003.

[13] *Context-Adaptive VLC (CVLC) Coding of Coefficients*, document JVT-C028.doc, Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T Q6/SG16m, May 2002.

[14] I. E. G. Richardson, "H.264/MPEG-4, part 10," in *H.264 and MPEG-4 Video Compression*. New York: Wiley, 2003, ch. 6, pp. 201–207.

[15] K. Sayood, "Universal codes," in *Lossless Compression Handbook*. New York: Academic, 2003, ch. 3, pp. 62–64.

[16] *Joint Video Team*, Reference Software Version 13.2 [Online]. Available: http://iphome/hhi.de/shehring/tml/download/old_jm/jm13.2.zip

[17] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000.

[18] K. Sayood, "Lossless image compression," in *Introduction to Data Compression*. San Mateo, CA: Morgan Kaufmann, 2006, ch. 7, pp. 170–172.

[19] *Motion JPEG2000 Final Committee Draft*, document N2117.doc, ISO/IEC JTC 1/SC29/WG 1, Mar. 2001.

[20] http://www.t2-project.org/packages/jasper.html

[21] *Adaptive Prediction Error Coding in Spatial and Frequency Domain for H.264/AVC*, document VCEG-AB06.doc, ITU-T Q.6/SG16, Jan. 2006.

[22] *Adaptive Prediction Error Coding in Spatial and Frequency Domain With a Fixed Scan in the Spatial Domain*, document VCEG-AD07.doc, ITU-T Q.6/SG16, Oct. 2006.

[23] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[24] D. Marpe, H. Schwarz, and 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, pp. 620–636, Jul. 2003.

Jin Heo (S'09) received the B.S. degree in electrical engineering from Kwangwoon University, Seoul, Korea, in 2004 and the M.S. degree in information and communication engineering from the Gwangju Institute of Science and Technology (GIST), Gwangju, Korea, in 2006. He is currently pursuing the Ph.D. degree from the Department of Information and Communications at GIST, Korea.

His research interests include digital image and video coding, multiview video coding, and future video coding technologies.

Seung-Hwan Kim received the B.S. degree in electronic engineering from Ajou University, Suwon, Korea, in 2001, and the M.S. and Ph.D. degrees in information and communication engineering from the Gwangju Institute of Science and Technology, Gwangju, Korea, in 2003 and 2008, respectively.

He is currently a Postdoctoral Research Fellow in the Department of Electrical Engineering, University of Southern California, Los Angeles. His research interests include digital image and video coding, scalable video coding, multiview video coding, and digital video broadcasting.

Yo-Sung Ho (SM'06) received the B.S. and M.S. degrees in electronic engineering from Seoul National University, Seoul, Korea, in 1981 and 1983, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, Santa Barbara, in 1990.

He joined the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea, in 1983. From 1990 to 1993, he was with Philips Laboratories, Briarcliff Manor, NY, where he was involved in development of the advanced digital high-definition television system. In 1993, he rejoined the Technical Staff of ETRI and was involved in development of the Korea direct broadcast satellite digital television and high-definition television systems. Since 1995, he has been with the Gwangju Institute of Science and Technology, Gwangju, Korea, where he is currently a Professor in the Department of Information and Communications. His research interests include digital image and video coding, image analysis and image restoration, advanced coding techniques, digital video and audio broadcasting, 3-D television, and realistic broadcasting.