CONTEXT-ADAPTIVE HYBRID VARIABLE LENGTH CODING IN H.264/AVC

Junlin Li[1], *Ghassan AlRegib*[2], *Dihong Tian*[1], *Wen H. Chen*[1], *and Pi Sheng Chang*[3]

[1]Cisco Systems, Inc., San Jose, CA, USA
[2]Georgia Institute of Technology, Atlanta, GA, USA
[3]Plantronics, Inc., Santa Cruz, CA, USA
{junlli, dtian, wenc}@cisco.com, gregib@ece.gatech.edu, pschang@ieee.org

ABSTRACT

It is well observed that the nonzero transform coefficients for block-based hybrid video coding is more clustered in the low frequency region while more scattered in the high frequency region. Based on this observation, hybrid variable length coding (HVLC) was recently proposed, which divides each transform block into low frequency (LF) region and high frequency (HF) region by a pre-defined breakpoint and codes the clustered low frequency coefficients with new coding schemes, such as two dimensional position and one-dimensional amplitude coding (2DP1DA) or three-dimensional position and amplitude coding (3DPA). However, the transition between clustered and scattered nonzero coefficients varies with different blocks and sequences. In this paper, we propose a context-adaptive hybrid variable length coding (CAHVLC) scheme, where multiple code tables are used and the code table is adaptively chosen based on the derivable context information from coded neighboring blocks or from the coded portion of the currently being coded block. The experimental results show that CAHVLC achieves about $6.5\% \sim 8\%$ bit rate reduction for a wide range of quantization parameters (QP) compared with CAVLC in H.264.

Index Terms— Video coding, H.264, VLC, context-adaptive.

1. INTRODUCTION

Variable length coding (VLC) is widely deployed for entropy coding in the existing video coding standards, such as MPEG-2/4, H.263, H.264/AVC and its extension of scalable video coding (SVC). In H.263, run-level based VLC is adopted, referred to as RL-VLC, where "run" indicates the number of zeros preceding a nonzero coefficient, "level" indicates the magnitude of the nonzero coefficient, and the "run" and "level" are jointly coded by a two-dimensional code table. In H.264/AVC, context-adaptive VLC, referred to as CAVLC, is used, where "run" and "level" are separately coded by a code table adaptively chosen from multiple code tables. Multiple tables based RL-VLC with context-adaptive table switch is also studied in [1, 2, 3]. Run-level based coding schemes, such as RL-VLC in H.263 and CAVLC in H.264, are efficient to code scattered nonzero coefficients; however, they are inefficient in coding clustered nonzero coefficients, due to the fact that n separate codes are required to represent n consecutive nonzero coefficients, each of which has a run equal to zero.

To overcome the inefficiency of the conventional run-level coding scheme in coding consecutive nonzero transform coefficients, hybrid variable length coding (HVLC) [4] was recently proposed. In HVLC, a breakpoint, which is a coefficient index along the coefficient scan path, is first defined to divide the transform coefficient block into low-frequency (LF) and high-frequency (HF) regions, where the nonzero coefficients are statistically more clustered in LF region while more scattered in HF region. In HVLC, the conventional RL-VLC or an equivalent scheme is used for coding the HF coefficients, while several new coding schemes were proposed to exploit the clustered nature of nonzero coefficients in LF region, such as two-dimensional position and one dimensional amplitude coding (2DP1DA) scheme [4] and three-dimensional position and amplitude coding (3DPA) scheme [5]. It is noted that a fixed breakpoint is used in HVLC, which is not necessarily optimal since the transition between clustered and scattered nonzero coefficients varies with different blocks and sequences. In this paper, we further design a context-adaptive hybrid variable length coding scheme (CAHVLC), where multiple code tables are used to code both position and amplitude information and the code table is adaptively chosen based on the derivable context information, which includes both the information from the coded neighboring blocks and the information from the coded portion of the currently being coded block.

2. HYBRID THREE-DIMENSIONAL POSITION AND AMPLITUDE CODE

To efficiently code a sequence of quantized transform coefficients, there are essentially two types of information to be coded: the positions of the nonzero coefficients and their corresponding amplitudes. In RL-VLC scheme, the position and amplitude of each nonzero coefficient are jointly coded as a (run, level) pair. In 2DP1DA scheme [4], the positions of consecutive nonzero coefficients are jointly coded as a two-dimensional position code (2DP), i.e., the run of zeros that precede the nonzero cluster and the run of nonzero coefficients within the cluster, while the amplitudes of the nonzero coefficients are coded separately from the position information. In 3DPA [5], the 2D position information of each nonzero cluster and the number of trailing ones in the nonzero cluster are jointly coded as a triple using a three-dimensional table to further improve the coding efficiency. In all the previously proposed HVLC algorithms [4, 5], two different coding schemes are adopted to code the clustered low-frequency coefficients and the scattered high-frequency coefficients, respectively. In this paper, based on the 3DPA scheme [5], we design a hybrid three-dimensional position and amplitude code to code both the clustered and scattered coefficients, and multiple code tables are constructed to efficiently code different coefficients.

Denote the run of the zero-valued coefficients that precede the nonzero cluster as R_z, denote the run of the consecutive nonzero-valued coefficients as R_n, denote the amplitude information of the nonzero cluster as A, then the proposed hybrid scheme codes R_z, R_n and T_1 as a triple by a three-dimensional code table. Since a run

Table 1. An Example of Hybrid Three-Dimensional Position and Amplitude Coding

Index:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	···
Coefficient:	9	-5	3	-2	1	0	0	2	1	1	0	0	0	-1	1	0	0	0	2	0	···
Coding Stream:	$C_{3D}(0,5,1)$ $C_A(9)$ $C_A(5)$ $C_A(3)$ $C_A(1)$ $C_{3D}(1,3,2)$ $C_A(1)$ $C_{3D}(2,2,2)$ $C_{3D}(2,1,2)$																				

of nonzero-valued coefficients implies that the following coefficient is a zero-valued coefficient (otherwise it would have been counted into the nonzero cluster), it can be skipped in the coding process to save bits, that is to say, each run of zero-valued coefficients can be reduced by 1 before it is coded, with the exception of the first run at the beginning of a block. Therefore, the code symbol has the following form:

$$\begin{cases} (R_z, R_n, A), & \text{for the first symbol in a block,} \\ (R_z - 1, R_n, A), & \text{otherwise,} \end{cases}$$

where the amplitude information A is defined as follows:

$$A = \begin{cases} T_1, & \text{if } R_n > 1, \\ L, & \text{if } R_n = 1, \end{cases}$$

where T_1 denotes the number of trailing coefficients with amplitude +/-1 in the nonzero cluster, and L denotes the amplitude level of an isolated nonzero coefficient. It is noted that the proposed hybrid three-dimensional code symbol is equivalent to run-level pair when $R_n = 1$. Furthermore, a binary information can be incorporated into the proposed coding scheme to indicate whether this is the last nonzero coefficient in the block as in [5]. An additional one-dimensional amplitude coding is also needed to code the nonzero coefficients except the trailing T_1 ones in the nonzero cluster with more than one nonzero coefficients. The amplitudes of all the remaining nonzero coefficients except the one immediately preceding the trailing ones are coded directly by a 1D VLC, while the amplitude of the coefficient immediately preceding the trailing ones minus 1 is coded by a 1D VLC to shorten the codeword since it must be larger than 1.

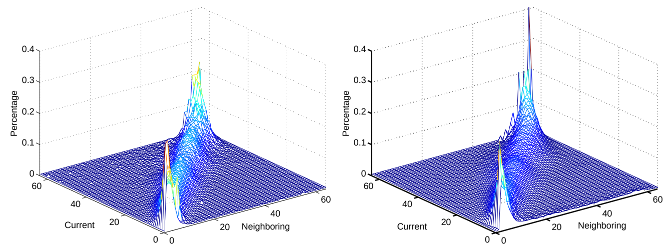
To illustrate the proposed hybrid 3DPA coding scheme, consider an example shown in Table 1, where all the remaining coefficients in the sequences are assumed to be zeros. In the coding stream, C_{3D} denotes the codeword of the hybrid 3DPA coding, C_A denotes the codeword of the one-dimensional amplitude coding. Note that the last information and sign information are omitted in Table 1.

3. CONTEXT-ADAPTIVE CODE TABLE SWITCH

Since the distribution of transform coefficients varies with different blocks and sequences, multiple code tables representing different possibility distribution and context-adaptive code table switch are desirable to further improve the coding efficiency. In the section, we discuss the context-adaptive code table switch for both hybrid 3DPA coding and one-dimensional amplitude coding. Context information derived from both the coded neighboring blocks and the coded portion of the current block in the coding process are utilized.

3.1. Adaptive 3DPA Coding for the First Nonzero Cluster

As shown in Fig. 1, the number of nonzero coefficients in the neighboring blocks are highly correlated. Thus to efficiently code the hybrid 3DPA information of the first nonzero cluster in the current block, we use the average number of nonzero coefficients in the left and above neighboring blocks, denoted as N_n, of the current block as the context information to choose the code table. In our experiments, there are six different VLC tables (T_0 to T_5) to choose from.



(*a*) (*b*)

Fig. 1. The distribution of the total number of nonzero coefficients in the current block given the total number of nonzero coefficients in the neighboring blocks for QP=13: (a) INTRA, (b) INTER.

T_0 is biased towards the blocks with less nonzero coefficients; T_1 is biased towards the blocks with slightly more nonzero coefficients and so on. The code table is chosen based on the range of N_n as shown in Table 2.

Table 2. Code Table Switch for the First Nonzero Cluster

N_n	$[1,4]$	$[5,8]$	$[9,16]$	$[17,24]$	$[25,36]$	$[37,64]$
Table	T_0	T_1	T_2	T_3	T_4	T_5

3.2. Adaptive 3DPA Coding for Remaining Nonzero Clusters

It is well observed that the quantized transform coefficients after zigzag scan is more clustered in the LF region, while it is more scattered in the HF region [4]. Also, from LF to HF region, the nonzero run tends to decrease, while the zero run tends to increase. Fig. 2 shows the correlation of the nonzero runs and zero runs between neighboring clusters within the same block. In this paper, we exploit this correlation to code all the remaining clusters except the first cluster in the block. More specifically, the nonzero run-length and zero run-length of the previously coded clusters serve as the context information to adaptively choose the code table for coding the following cluster. In our experiments, there are thirty-six different VCL tables (T_{00} to T_{55}) to choose from as shown in Table 3. T_{00} is biased towards the clusters with larger nonzero run-length and smaller zero run-length; T_{11} is biased towards the clusters with slightly smaller nonzero run-length and slightly larger zero run-length and so on. Assume the previous cluster with nonzero run-length R_n and zero run-length R_z is coded using the code table T_{ij} $(0 \le i, j \le 4)$, then the code table for coding the current cluster is chosen as follows:

$$\mathrm{T} = \begin{cases} \mathrm{T}_{ij}, & \text{if } R_z \le z_i,\ R_n > n_j \\ \mathrm{T}_{(i+1)j}, & \text{if } R_z > z_i,\ R_n > n_j \\ \mathrm{T}_{i(j+1)}, & \text{if } R_z \le z_i,\ R_n \le n_j \\ \mathrm{T}_{(i+1)(j+1)}, & \text{if } R_z > z_i,\ R_n \le n_j \end{cases}$$

where, $z_0 = n_4 = 1, z_1 = n_3 = 2, z_2 = n_2 = 4, z_3 = n_1 = 7, z_4 = n_0 = 10$. It is noted that the code table switch is restrained in a monochrome direction from the top-left to the bottom-right. In other words, a table switch occurs when either R_z increases or

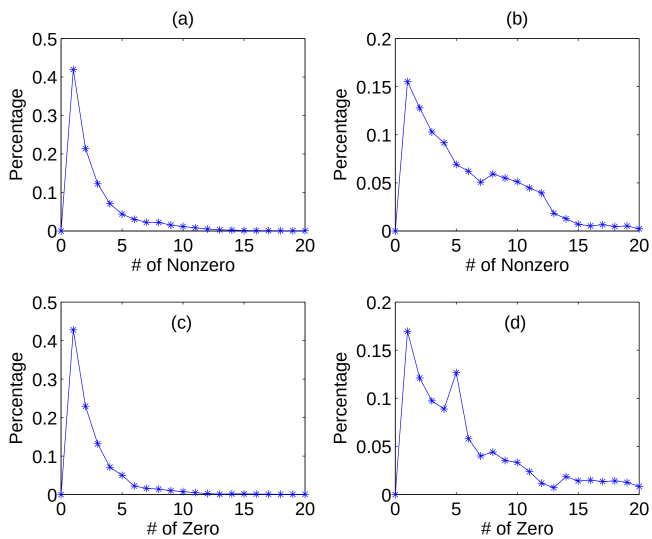


Fig. 2. Correlation between the neighboring clusters within the same block (INTER-coded with QP=13): (a-b) the distribution of the nonzero run-length of the current cluster given the nonzero run-length of the previously coded cluster as 5 and 15 respectively; (c-d) the distribution of the zero run-length preceding the current cluster given the zero run-length preceding the previously coded cluster as 5 and 15 respectively.

R_n decreases and the value falls into a new range according to the thresholds. Furthermore, the two-dimensional code table switch is restrained at most one-step from left to right each time based on the nonzero run-length parameter while multiple steps from top to bottom each time is allowed based on the zero run-length parameter.

Table 3. Code Table Switch for Remaining Nonzero Clusters

Table		R_n^{th}					
		n_0	n_1	n_2	n_3	n_4	N/A
R_z^{th}	z_0	T_{00}	T_{01}	T_{02}	T_{03}	T_{04}	T_{05}
	z_1	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}
	z_2	T_{20}	T_{21}	T_{22}	T_{23}	T_{24}	T_{25}
	z_3	T_{30}	T_{31}	T_{32}	T_{33}	T_{34}	T_{35}
	z_4	T_{40}	T_{41}	T_{42}	T_{43}	T_{44}	T_{45}
	N/A	T_{50}	T_{51}	T_{52}	T_{53}	T_{54}	T_{55}

3.3. Adaptive Amplitude Coding

As mentioned in Section 2, an additional one-dimensional amplitude coding is needed to code the amplitude of the nonzero coefficients except the trailing ones in a nonzero cluster with more than one nonzero coefficients. Within each nonzero cluster, the one-dimensional amplitude coding is performed in a reverse scan order, i.e., from high frequency coefficients to low frequency coefficients. It is well-known that the amplitude of the nonzero coefficients tends to increase from the high frequency to low frequency. In this paper, we use the previously coded amplitude as the context information to adaptively code the amplitude of the next to-be-coded nonzero coefficient. In our experiments, there are seven code tables (T_0^A to T_6^A) to choose from as shown in Table 4. T_0^A is biased towards smaller amplitudes, T_1^A is biased towards slightly bigger amplitudes and so on. Assume the previous nonzero coefficient with amplitude A_p is coded with code table T_i^A $(0 \leq i \leq 5)$, then the code table for coding the current nonzero coefficient is chosen as follows:

$$T = \begin{cases} T_i^A, & \text{if } A_p \leq a_i \\ T_{i+1}^A, & \text{if } A_p > a_i \end{cases}$$

where $a_0 = 0, a_1 = 3, a_2 = 6, a_3 = 12, a_4 = 24, a_5 = 48$. Furthermore, multiple-step switch from left to right each time is also allowed based on the amplitude parameter.

Table 4. Code Table Switch for Amplitude Coding

A_{th}	a_0	a_1	a_2	a_3	a_4	a_5	N/A
Table	T_0^A	T_1^A	T_2^A	T_3^A	T_4^A	T_5^A	T_6^A

4. EXPERIMENTAL RESULTS

In this section, we report the preliminary test results of the proposed CAHVLC in H.264/AVC. Since the proposed CAHVLC algorithm is targeted at coding high resolution video sequences with 8×8 transform block size, H.264/AVC FRExt with fixed 8×8 transform is used in the simulation. The test database includes ten 4CIF standard test sequences. Each sequence has 210 frames with a frame rate of 25 frames per second (fps). The GOP length is 15 frames, where one is INTRA coded and the others are INTER coded. In our experiments, we test three different quantization parameters (QP = 5, 25, 37) as test examples for small, medium, and large QPs respectively. For comparison purpose, HVLC with 3DPA [5] and CAVLC in H.264/AVC are also tested. For 3DPA, a constant breakpoint is optimally chosen for a given quantization parameter (QP) based on the statistics from the training sequences. All the experiments are conducted using H.264/AVC FRExt with fixed 8×8 transform. Note that the complexity of HVLC with 3DPA algorithm and CAVLC in H.264 is much less than the complexity of CABAC in H.264 [6], therefore, CABAC is not included in the comparison.

Table 5, 6 and 7 show the bit-rate results for coding the test sequences with QP = 5, 25, and 37, respectively. All the bit-rate results are in kbits/sec. A positive percentage in the tables indicates the bit-rate reduction achieved by HVLC or CAHVLC compared with H.264 with CAVLC, while a negative percentage indicates that the bit rate is increased by HVLC or CAHVLC. From the results, we can see that compared with CAVLC in H.264, HVLC with 3DPA can reduce the bit rate considerably for a wide range of QP, i.e., 4.08% for QP=5, 4.46% for QP=25 and 5.06% for QP=37. Furthermore, with the context-adaptation, CAHVLC further reduces the bit rate significantly, i.e., 6.54% for QP=5, 6.89% for QP=25 and 7.78% for QP=37 bit rate reduction compared with CAVLC in H.264. It is noted that only the bit rate used for coding Luma component is included in the test results.

5. CONCLUSIONS

In this paper, we proposed an efficient context-adaptive hybrid variable length coding scheme (CAHVLC) based on hybrid three dimensional position and amplitude coding and one-dimensional amplitude coding, where multiple code tables representing different possibility distributions are used and context-adaptive code table switch schemes are also developed to optimally choose the code table for both position and amplitude coding. The experimental results show that significant improvement on coding efficiency can be achieved comparing to HVLC and CAVLC in H.264.

Table 5. Bit-Rate Results for QP = 5 (kb/s)

Sequence	INTRA			INTER			Overall		
	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC
Crew	2732.13	0.62%	3.82%	34970.49	0.87%	3.42%	37702.62	0.85%	3.45%
Harbour	3614.30	3.68%	6.72%	41947.97	1.47%	3.11%	45562.27	1.65%	3.40%
Soccer	3034.17	3.42%	5.92%	34502.42	2.22%	4.93%	37536.59	2.32%	5.01%
Barcelona	4622.69	-0.39%	4.92%	51346.84	5.74%	7.39%	55969.53	5.23%	7.18%
Fries	3118.23	1.25%	4.49%	39193.07	1.09%	4.34%	42311.30	1.10%	4.35%
Mobile	4782.22	1.05%	6.99%	48810.12	4.43%	7.24%	53592.34	4.13%	7.22%
Music	4268.96	1.62%	6.22%	51790.34	3.82%	6.31%	56059.30	3.66%	6.30%
Race	4123.26	4.90%	7.87%	51458.21	4.73%	7.52%	55581.47	4.74%	7.54%
Rower	3788.56	12.38%	12.92%	43435.11	7.41%	9.51%	47223.67	7.81%	9.79%
Rugby	4030.01	9.39%	10.76%	50057.02	7.05%	9.06%	54087.03	7.22%	9.19%
Total	38114.53	3.79%	7.19%	447511.60	4.11%	6.49%	485626.12	4.08%	6.54%

Table 6. Bit-Rate Results for QP = 25 (kb/s)

Sequence	INTRA			INTER			Overall		
	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC
Crew	462.76	2.35%	4.73%	3094.34	1.68%	4.26%	3557.11	1.77%	4.32%
Harbour	983.21	6.45%	8.70%	6048.15	3.09%	4.80%	7031.35	3.56%	5.35%
Soccer	720.02	3.12%	5.17%	3734.98	2.92%	5.70%	4455.00	2.96%	5.62%
Barcelona	1549.78	-1.65%	3.19%	8673.26	5.32%	6.59%	10223.05	4.27%	6.07%
Fries	482.39	2.83%	5.87%	3301.83	1.65%	4.46%	3784.22	1.80%	4.64%
Mobile	1696.40	-1.72%	2.80%	8926.48	4.09%	5.62%	10622.88	3.17%	5.17%
Music	1094.63	-1.95%	3.05%	7289.75	1.68%	4.46%	8384.38	1.20%	4.28%
Race	1244.15	4.69%	6.95%	12208.19	5.42%	7.11%	13452.34	5.36%	7.10%
Rower	1048.61	12.18%	14.58%	8279.52	7.66%	10.87%	9328.13	8.17%	11.28%
Rugby	1189.16	9.52%	11.81%	11417.31	6.51%	9.82%	12606.46	6.80%	10.01%
Total	10471.11	3.19%	6.52%	72973.80	4.65%	6.95%	83444.91	4.46%	6.89%

Table 7. Bit-Rate Results for QP = 37 (kb/s)

Sequence	INTRA			INTER			Overall		
	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC	H.264	HVLC	CAHVLC
Crew	94.51	4.46%	4.39%	290.34	4.77%	7.63%	384.85	4.70%	6.84%
Harbour	297.58	6.95%	7.41%	594.38	3.85%	7.49%	891.96	4.88%	7.46%
Soccer	151.47	3.04%	4.08%	415.39	4.37%	6.58%	566.86	4.02%	5.91%
Barcelona	421.56	1.37%	3.07%	647.01	5.28%	5.41%	1068.57	3.74%	4.48%
Fries	116.79	5.82%	6.50%	492.02	4.57%	6.32%	608.81	4.81%	6.35%
Mobile	572.93	-0.59%	1.66%	1147.02	5.40%	5.43%	1719.95	3.41%	4.17%
Music	343.30	-1.15%	1.64%	1238.60	2.91%	3.52%	1581.90	2.03%	3.11%
Race	359.75	4.14%	6.68%	2259.25	4.78%	8.01%	2619.00	4.69%	7.83%
Rower	300.80	8.64%	12.98%	1559.73	6.20%	11.37%	1860.53	6.60%	11.63%
Rugby	342.76	8.36%	11.93%	2335.01	8.11%	12.47%	2677.78	8.15%	12.40%
Total	3001.45	3.47%	5.73%	10978.75	5.50%	8.34%	13980.20	5.06%	7.78%

6. REFERENCES

[1] Q. Wang, D.-B. Zhao, and W. Gao, "Context-based 2D-VLC entropy coder in AVS video coding standard," *J. Comput. Sci. Technol.*, vol. 21, no. 3, pp. 315–322, 2006.

[2] G. Lakhani, "Optimal huffman coding for DCT blocks," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 14, no. 4, pp. 522–527, Apr. 2004.

[3] C. Tu, J. Liang, and T. Tran, "Adaptive runlength coding," *IEEE Signal Processing Letters*, vol. 10, no. 3, pp. 61–64, Mar. 2003.

[4] D. Tian, W. H. Chen, P. S. Chang, G. AlRegib, and R. Mersereau, "Hybrid variable length coding for image and video compression," in *Proc. of the Intl. Conf. on Acoustics, Speech, and Signal Processing*, Honolulu, HI, Apr. 15-20 2007, vol. 1, pp. 1133–1136.

[5] J. Li, G. AlRegib, D. Tian, P. S. Chang, and W. H. Chen, "Three-dimensional position and amplitude VLC coding in H.264/AVC," in *Proc. of IEEE International Conference on Image Processing*, San Diego, CA, Oct. 12-15 2008.

[6] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits and Systems for Video Tehnology*, vol. 13, no. 7, pp. 620–636, Jul. 2003.