Entropy Coding Based On Code-Value Compact Memory Structure for AVS Video Coding Standard*

Lejun Yu[1,2] **Feng Dai**[1,2] **Yongdong Zhang**[1] **Jintao Li**[1]

[1] Institute of Computing Technology, Chinese Academy of Sciences
Beijing 100080, P. R. China
[2] Graduate University of Chinese Academy of Sciences
Beijing China 100049, P. R. China
E-mail: {jlyu, fdai, zhyd, jtli}@ict.ac.cn

Abstract: Entropy coding based on k-th order Exp-Golomb (EGk) codes is a key part in the new AVS video coding standard issued by Audio Video Coding Standard Workgroup of China. An efficient design based on code-value compact memory structure (CVCMS) is proposed to reduce the computational complexity and memory requirement. Only 789 byte memory is required for variable length coding (VLC) tables in CVCMS, which is just about 5.92% compared with that of the reference software. Furthermore, code-value is stored in memory, which reduced the computational complexity of EGk coding. The simulation results show that the proposed entropy coding for AVS video coding standard reduces the computational cost by 26.48%.

Key words: entropy coding, AVS video coding standard，k-th order Exp-Golomb codes，variable length coding (VLC) tables，code-value compact memory structure(CVCMS)

I. INTRODUCTIONS

Most of the video coding standards, such as the H.26x[7] issued by International Telecommunication Union (ITU), MPEG-X[6] issued by International Organization for Standardization (ISO) and the new AVS video coding standard issued by Audio Video coding Standard (AVS) Workgroup of China, are built on the base of hybrid coding framework. In this framework, there are mainly three steps: motion estimation, transform and entropy coding. Entropy encoding is a very important coding tool, which compresses the bit stream based on the statistic probability distribution of syntax elements to be coded, such as macroblock mode information, transform coefficients and so on. Coefficient coding is key part for optimizing the realization of entropy doing. On one hand, the data structure of variable length coding (VLC) tables relates to the memory required. For example, the reference software [3] of AVS video coding standard requires about 14K bytes for VLC tables, which is too large for embedded system and DSP platform. On the other hand, the content in VLC tables relates to the computational complexity. For instance, code-number is stored in VLC tables, which should be converted to code-word with *k*-th Exp-Golomb (EGk) binarization scheme. The computational complexity of entropy coding is great if the video to be coded has large spatial size.

A new design for entropy coding of AVS video coding standard should be proposed, because there are many differences between the entropy coding of AVS video coding standard, MPEG-2 and H.264/AVC. The differences of Runs and Levels coding, which are run-length of zeros and non-zero quantified coefficient levels in Zig-Zag scan order, are detailed as follows. In MPEG-2, (Run, Level) pairs are coded with Huffman codes, and the signs of coefficient levels are suffixed directly. In H.264/AVC, Runs and Levels are coded separately with EGk. In AVS video coding standard, (Run, Level) pairs are coded with EGk, but the signs of Levels can not be simply suffixed as MPEG-2 does.

Entropy coding based on code-value compact memory structure (CVCMS) is proposed in this paper, which can save computational complexity and memory. The entropy coding of AVS video coding standard is introduced in section II. Entropy coding based on CVCMS is proposed in section III, where the method deriving code-value of negative Level from corresponding absolute value is proved based on the EGk codes characteristics in subsection A, and the CVCMS is detailed in subsection B. At last, simulation results on DSP are give in section IV, which confirms the efficiency of proposed CVCMS for entropy coding in AVS video coding standard.

II. ENTROPY CODING IN AVS VIDEO CODING STANDARD

There are four categories of entropy coding for different syntax elements in AVS video coding standard, which are shown in Table 1. ue(v), se(v) and me(v) are used to code unsigned integer (such as MB type), signed integer (such as motion vector) and coded block patter (CBP) information respectively, which are coded with 0-th Exp-Golomb coding. ce(v) is used to code coefficients, which is coded with context adaptive EGk coding.

Table 1. Categories of entropy coding

Categories	Description
ue(v)	Unsigned integer syntax element coding, such as MB type.
se(v)	Signed integer syntax elements coding, such as motion vector.
me(v)	Mapping elements syntax elements coding, such as coded block pattern (CBP).
ce(v)	Coefficients coding, such as (Run, Level) pairs.

The residual macroblock after intra or inter prediction is transformed, and then a series of (Run, Level) pairs and an end of block (EOB) flag are obtained after Zig-Zag scanning on the coefficients. These (Run, Level) pairs and EOB are coded context adaptively with EGk. The main highlights of entropy coding of AVS video coding standard include [2]:

(a) Coefficients of intra/inter luma and chroma blocks are coded with separate VLC tables. Because the statistic of intra luma block, inter luma block and chroma block are different, coefficient coding with separate VLC tables improves the coding efficiency.

(b) (Run, Level) pairs are coed context adaptively. Because there are correlation between the levels, the amplitude of a level is used as a condition for the VLC table selection of the next

* This work was supported by the National Nature Science Foundation of China (60302028 and 60473002).

Authorized licensed use limited to: INSTITUTE OF COMPUTING TECHNOLOGY CAS. Downloaded on January 14, 2009 at 02:21 from IEEE Xplore. Restrictions apply.

(Run, Level) pair, which can improve efficiency of coefficient coding also.

To sum up, there are 7 VLC tables for intra luma coefficients coding, 7 VLC tables for inter luma coefficients coding and 5 VLC tables for intra/inter chroma coefficients. With the storage scheme of VLC tables in reference software, about 14K byte memory is required. Furthermore, as the description about coefficient coding in AVS video coding standard, code-number is stored in the VLC tables of in the reference software, which make the process of binarization a code-number into a code-word consumes great time.

The improvement of coefficient coding is the key for improve entropy coding efficiency , because the coefficient coding takes the most part of the time for entropy coding, We propose to store code-value in the VLC tables of coefficient coding, and design a compact memory structure to save memory needed by the VLC tables, which is the CVCMS in section III.

III. ENTROPY CODING DESIGN

A. the method of deriving code-word

EGk coding is adopted in AVS video coding standard, and part of the EGk codes is illustrated in Table 2. We can see that a code-word is composed of leading "0"s and an end of leading flag "1", followed by payload "$x_i \cdots x_0$". The number of leading "0" is dependent on the order k and the number of bit(s) in the payload.

Table 2. k-th order Exp-Golomb codes

Order	Code-word	Code-number
$k = 0$	1	0
	0 1 x_0	1-2
	0 0 1 x_1 x_0	3-6
	…	…
$k = 1$	1 x_0	0-1
	0 1 x_1 x_0	2-5
	0 0 1 x_2 x_1 x_0	6-13
	…	…
$k = 2$	1 x_1 x_0	0-3
	0 1 x_2 x_1 x_0	4-11
	0 0 1 x_3 x_2 x_1 x_0	12-27
	…	…
$k = 3$	0 1 x_2 x_1 x_0	0-3
	0 0 1 x_3 x_2 x_1 x_0	4-11
	0 0 0 1 x_4 x_3 x_2 x_1 x_0	12-27
	…	…
…	…	…

When the code-word is looked as an unsigned integer G, a code-word can be denoted with (G, L) pair, where L is the length of code-word. If the code-number is C, there is a simple method to derive G from C [5]. Let the unsigned integer value of payload "$x_i \ldots x_0$" to be X, the corresponding code-word of k-th Exp-Golomb coding is ($G(X)$, $L(X, k)$), then $G(X)$ is the unsigned integer value of "1 $x_i \ldots x_0$", i.e.,

$$G(X) = 2^{i+1} + X\ . \qquad (1)$$

Payload "$x_{j-1} \ldots x_0$" with j bits can express 2^j code-numbers, so the EGk code-number corresponding to X is

$$C(X,\ k) = X + \sum_{j=k}^{i} 2^j = X + 2^{i+1} - 2^k\ . \qquad (2)$$

With Eq. (1) and (2), G can be expressed as the function about C and k, i.e.,

$$G(C,k) = 2^k + C\ , \qquad (3)$$

which indicates G can be derived from C with just simple shift and add operations.

The length L can be realized with lookup table or calculating method. For example, on the DSP of TI, the instruction of _lmbd can be used to find the first bit "1" from the most significant bit [4]. The length L can be calculated with

$$L(X,k) = ((32 - \_\mathrm{lmbd}(1, G(C,k))) << 1) + 1 - k\ , \quad (4)$$

where "<<" is left shift operator. Eq. (4) requires the $G(C, k)$ can be expressed as 32-bit integer, which can be satisfied in AVS video coding standard. This method is used to calculate L in the simulation following.

The entropy coding of ue(v), se(v) and me(u) can be realized simply. Let the value of the syntax element to be coded with ue(v) is Y_u, then C=Yu; let the value of syntax element to be coded with se(v) is Y_s, then $C = \begin{cases} 2 \times Y_s \text{ -1} & Y_s > 0 \\ -2 \times Y_s & Y_s \le 0 \end{cases}$; let the mapping value of CBP is Y_m, then C=Y_m. After that, code word (G, L) can be derived with Eq. (3) and (4). However, the coefficient coding with ce(v) is not so simple considering the signs coding of Levels, which is detailed in the following.

The coding scheme of ce(v) in reference software is illustrated in Figure 1. A value C_+ is obtained by looking up VLC tables with index (Run, abs(Level)). If Level is non-negative, then code-number C=C_+, otherwise, C=C_++1 is the entropy code of (Run, Level). Then code-word is derived with Eq. (3) and (4).

Step1. Lookup code-value C_+ in VLC tables with (Run, abs(Level));
Step2. Derive the code value C depending on the sign of Level $C = \begin{cases} C_+ & \text{Level} > 0 \\ C_+ + 1 & \text{Level} < 0 \end{cases}$;
Step3. Derive G and L with C and k with Eq. (3) and (4).

Fig. 1. coding scheme of ce(v) in reference software

To reduce the computational complexity entropy coding of ce(v), the code-value of positive Level G_+ is stored in VLC tables directly, which makes the calculation in (3) unnecessary. The code-value of negative Level, which is denoted as G-, can not be found in the tables, but it can be simply derived with

$$G_- = G_+ + 1\ , \qquad (5)$$

which is proved as follows. Replace C in (3) with C+t, i.e.,

$$G(C+t) = 2^k + C + t = G(C) + t\ . \qquad (6)$$

For a negative Level, the code-number of (Run, abs(Level)) is C_+, so the corresponding code value of C_+ is

$$G_+ = G(C_+) \qquad (7)$$

And the code-number of the negative Level C- = C_+ + 1, so

$$G_- = G(C_-) = G(C_+ + 1) \qquad (8)$$

With Eq. (6), (7) and (8), it can be derived that

$$G_- = G(C_+ + 1) = G(C_+) + 1 = G_+ + 1 \qquad (9)$$

In conclusion, when positive G_+ is stored in VLC tables, the code-value G of (Run, Level) can be derived with

$$G = \begin{cases} G_+ & \text{Level} > 0 \\ G_+ + 1 & \text{Level} < 0 \end{cases} \qquad (10)$$

The proposed ce(v) coding scheme is illustrated in Figure 2.

Authorized licensed use limited to: INSTITUTE OF COMPUTING TECHNOLOGY CAS. Downloaded on January 14, 2009 at 02:21 from IEEE Xplore. Restrictions apply.

Step1. Lookup code-value G+ in VLC tables with (Run, abs(Level));

Step2. Derive the code-value G

$$G = \begin{cases} G_+ & \text{Level} > 0 \\ G_+ + 1 & \text{Level} < 0 \end{cases}.$$

Step3. Calculate L with Eq. (4).

Fig. 2. coding scheme of ce() proposed

B. code-value compact memory structure(CVCMS)

In the VLC tables, not all of (Run, Level) pairs can be coded with EGk because it can be escaped. For example, a VLC table for chroma coefficients is shown in Table 3, where the elements are located in the upper-left corner, and "-" means (Run, Level) pair is escaped. The two-dimension memory structure for these VLC tables wastes too much memory. In fact, VLC tables can be looked as sparse matrixes. We can scan a VLC table for left to right and top to down, all the effective elements are stored in a one-dimension continual memory. To fast index the Run, an offset value offsetI (I=1…N) is stored to locate the first code value with each Run, which illustrated in Figure 3. Thus, we lookup the VLC tables in 3 steps:

step1. offsetI is found by index Run;

step2. *G*+ is found with offsetI and abs(Level);

step3. *G* is derived with Eq. (10).

Table 3. Example of VLC table

Run	Level > 0								
	1	2	3	4	5	6	7	8	9
0	0	3	7	13	17	27	35	43	55
1	5	11	21	33	51	-	-	-	-
2	9	23	37	57	-	-	-	-	-
3	15	29	47	-	-	-	-	-	-
4	19	41	-	-	-	-	-	-	-
5	25	49	-	-	-	-	-	-	-
6	31	-	-	-	-	-	-	-	-
7	39	-	-	-	-	-	-	-	-
8	45	-	-	-	-	-	-	-	-
9	53	-	-	-	-	-	-	-	-

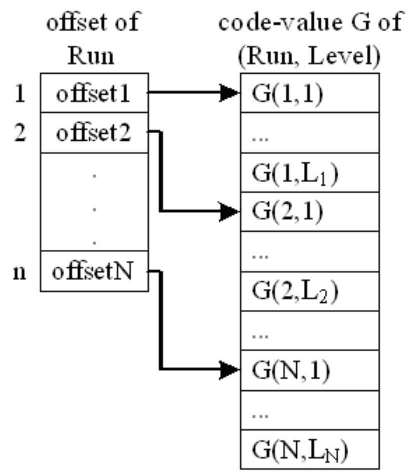

Fig. 3. CVCMS for coefficient coding in AVS video coding standard

The memory requirements of reference software (RS) and proposed CVCMS for VLC tables are compared, and the results are shown in Table 4. We can see that the reference software requires about 14K byte memory, but the CVCMS requires just 789 bytes for all VLC tables, which is just about 5.92% of RS.

Table 4. Memory requirement (byte)

VLC talbes	RS	CVCMS	Percentage
Intra luma	4914	275	5.60%
Inter luma	4914	295	6.00%
Chorma	3510	219	6.24%
Total	13338	789	5.92%

IV. SIMULATION RESULTS

The entropy coding scheme in reference software and proposed are realized on DSP TMS320C642. The environments of comparing simulation are the same, in which only the entropy coding parts are different. The clock cycles of writing every macroblock are profiled as the measure of entropy coding computational complexity, including the entropy coding of macroblock types, motion vectors and coefficients. The simulation results are given in Table 5, where sequences Boat and Piano are interlaced videos with size of 720×576, Crew and Harbor are progressive videos with size of 1280×720. It can be concluded that the coding time speedup is about 20%-30%, the average of which is about 26.48% on all the sequences.

Table 5. Entropy clock cycles per macroblock (cycle/MB)

Sequences	**RS**	**Proposed**	**Speedup(%)**
Boat	22891	16755	26.81%
Piano	37672	27148	27.94%
Crew	11508	8830	23.27%
Harbor	22839	16462	27.92%
Average			**26.48%**

V. CONCLUSIONS

An efficient design for entropy coding of AVS video coding standard is proposed. Based on the characteristics of EGk coding, we propose to store the code-value directly in the VLC table, which save the coding time. Furthermore, CVCMS saves the required memory for VLC tables, which takes up just about 5.92% that of reference software. And the simulation results show that the speedup of entropy coding is about 26.48% compared with the scheme in reference software.

REFERENCES

[1] AVS workgroup, "Advanced Audio Video System: Part two Video," Doc. N1276, Mar 2006.

[2] Q. Wang, D.B Zhao, "Improvement of context adaptive 2D-VLC variable length coding," version 1, 1995.

[3] AVS workgroup, "Reference software version rm5.1f"

[4] F. Li, F. Wang, "The theory and applications of TMS320c6000 series DSPs," Publishing. House of Electronic Industry, Beijing, Jan. 2003.

[5] Y. Zhong, Q.Wang, "MPEG-2 motion image compression standard and the new development," Tsinghua Univ. Press, Beijing, Mar. 2002

[6] D.Marpe, H. Schwarz and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard", IEEE Trans. Circuit Syst. Video Technol. Vol. 13, No.7, Jul. 2003

[7] JVT, "Draft ITU-T Recommendation H.264 and Draft ISO/IEC 14496-10 AVC", Doc JVT-50, Thailand, Mar. 2003

Authorized licensed use limited to: INSTITUTE OF COMPUTING TECHNOLOGY CAS. Downloaded on January 14, 2009 at 02:21 from IEEE Xplore. Restrictions apply.