HIERARCHICAL DEPENDENCY CONTEXT MODEL BASED ARITHMETIC CODING FOR DCT VIDEO COMPRESSION

Min Gao, Debin Zhao, Qiang Wang and Wen Gao

Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{mgao, dbzhao}@hit.edu.cn, {qwang, wgao}@jdl.ac.cn

ABSTRACT

This paper presents an arithmetic coding scheme for DCT coefficients in video compression, in which the number of non-zero coefficients, significant map and level information for a DCT block are used as coding elements. To exploit the statistical correlations, an hierarchical dependency context model (HDCM) is proposed, where the number of non-zero coefficients and scanned position are used to capture the magnitude varying tendency of DCT coefficients. Then a new binary arithmetic coding using HDCM (HDCMBAC) is proposed to code the coding elements. Experimental results demonstrate that HDCMBAC can achieve the similar coding performance as CABAC at low and high QPs. Meanwhile the context modeling and arithmetic decoding in HDCMBAC can be carried out in parallel, since the context dependency only exists among different parts of basic coding elements in HDCM.

Index Terms— Arithmetic coding, context modeling, video compression, DCT coefficients

1. INTRODUCTION

The motion-compensated hybrid coding framework is used in video compression. In such framework, Discrete Cosine Transform (DCT) has been widely applied to the prediction error after intra and inter prediction. In DCT domain, the statistical correlation of prediction error could be easily removed by entropy coding.

It is well known that arithmetic coding [1] allows us to compress the sequentially observed symbols as well as possible for a given model of the process that generated these symbols. For enabling the arithmetic coding to obtain the maximum coding efficiency, the key part is to design an accurate context model for the probability of occurrence of each symbol.

For video compression, DCT coefficients' statistical behaviors are usually used to design context model. For example, along the *zig-zag* scan path of DCT block, non-zero coefficients show a statistical decreasing tendency in magnitude and the run-length of successive zero coefficients shows a statistical increasing tendency. This priori domain knowledge is used to drive the context modeling in Context-Based Adaptive Binary Arithmetic Coding (CABAC) [2] for H.264/AVC [3] and Context-based Binary Arithmetic coding(CBAC) [4] for AVS [5].

During the standardization of entropy coding for HEVC [6], DCT coefficients' statistical behaviors are also used in the context modeling for transform coefficients coding [7]. In addition, the data throughput of entropy coding has been taken into account. Various techniques to improve the data throughput of entropy coding in HEVC have been reviewed in [8]. These techniques include reducing the number of context coded bins, grouping bypass coding bins, grouping the bins with the same context, reducing the context dependency, and reducing total bins.

In this paper, an alternative context model for DCT coefficients namely hierarchical dependency context model (HDCM) is proposed to exploit the statistical correlations. Then a binary arithmetic coding scheme based on HDCM (HDCMBAC) is presented, in which the number of non-zero coefficients, significant map and level information for a DCT block are used as coding elements. Since the context dependency in HDCM only exists among different coding elements, the context modeling and arithmetic decoding in HDCMBAC can be carried out in parallel. Experimental results demonstrate that HDCMBAC can achieve the similar coding efficiency as CABAC at low and high QPs.

The paper is organized as follows. Section 2 describes the context modeling process for DCT coefficients in CABAC. Section 3 presents the description of HDCM. Section 4 gives the implementation of HDCMBAC. The experimental results are presented in Section 5. Section 6 concludes the paper.

2. CONTEXT MODEL OF DCT COEFFICIENTS IN CABAC

The coding elements used by CABAC in H.264/AVC consist of *significant_flag*, *last_significant_flag* and level information. The *significant_flag* is a binary-valued symbol to indicate the coefficient at each scanned position is whether non-zero coefficient. If *significant_flag* is one, it means that a non-zero coefficient exists at this scanned position, a binary-valued symbol *last_significant_flag* is transmitted to indicate the current non-zero coefficient is whether the last non-zero coefficient. For each non-zero coefficient, its value (*level*) is encoded by level information, which consists of *coeff_abs_level_minus1* (representing the

absolute value of *level* minus 1) and *coeff_sign_flag* (representing the sign of *level*). The *coeff_abs_level_minus1* is mapped into a string of bins by UEG0 [2] before context modeling, since it is a non-binary valued symbol.

For a coefficient *coeff[i]* scanned at the *i*th position, the context model for *significant_flag* and *last_significant_flag* is described as follows:

$$C_{SIG}(coeff[i]) = C_{LAST}(coeff[i]) = i \quad (1)$$

where $C_{SIG}(\bullet)$ and $C_{LAST}(\bullet)$ are the context indexes of *significant_flag* and *last_significant_flag*, respectively.

The context model for *coeff_abs_level_minus1* consists of two parts. One is used for the first bin with bin index equal to 0, denoted as *bin0*. The other is used for the bins with bin index between 1 and 13, denoted as *bin13*. The context index $C_{abscoeff}(i, bin0)$ for *bin0* is determined as follows:

$$C_{abscoeff}(i, bin0) = \begin{cases} 4, & if\ NumLgt1(i) > 0 \\ \max(3, NumT1(i)), & otherwise \end{cases} \quad (2)$$

The context index $C_{abscoeff}(i, bin13)$ for *bin13* is determined as follows:

$$C_{abscoeff}(i, bin13) = 5 + \max(4, NumLgt1(i)) \quad (3)$$

In the two context models described above, *i* is the scanned position of the current *coeff_abs_level_minus1*; *NumLgt1(i)* is the accumulated number of the previously encoded non-zero coefficients with absolute value greater than 1; *NumT1(i)* is the accumulated number of the previously encoded non-zero coefficients with absolute value equal to 1.

3. HIERARCHICAL DEPENDENCY CONTEXT MODEL FOR DCT COEFFICIENTS

In this section, *significant_flag*, *bin0* and *bin13* are still used as coding elements, which are defined in Section 2.

The context modeling for *significant_flag* is defined as follows:

$$C_{SIG}(P, N) = P + (N-1) \times B \quad (4)$$

where *P* is the scanned position, *N* is the number of non-zero coefficients in DCT block and *B* is a constant denoting the DCT block size, e.g., $B = 16$ for a 4x4 DCT block.

The context modeling for *bin0* is defined as follows:

$$C_{bin0\_of\_abscoeff}(P, N) = P + (N-1) \times B \quad (5)$$

where *P*, *N* and *B* are the same as that in (4).

For *bin13*, the original context modeling in CABAC is used, which is shown in (3). Because there is no context dependency between *bin13*s of different *level*s, if *bin0* is coded in the reverse scan order.

Context modeling in (4) and (5) are inspired by the statistical behavior of DCT coefficients. The variances of probability distribution for *level* in different sub-bands are different. So it is reasonable to utilize sub-band position *P* to model the *level* [9]. However, it is not accurate enough. When the number of non-zero coefficients *N* in a DCT block is different, even in the same sub-band position, the *level* may still has diverse distributions, which is shown in Fig. 1. Actually, *N* partially reflects the influence to *level*'s distributions due to some important factors, such as quantization step size and local texture complexity. So *P* and *N* are adopted as the contexts.

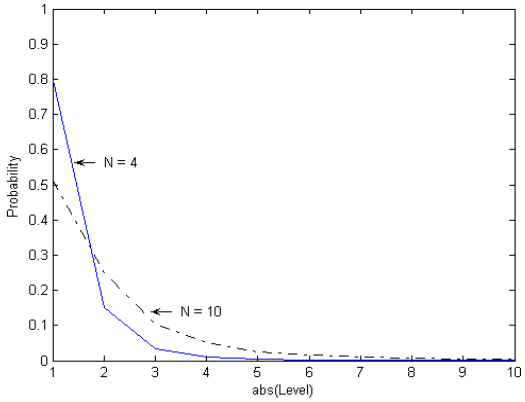

Fig. 1. Probability distributions of *level*'s magnitude at *P=0* when $N = 4$ and $N = 10$ in *Mobile&Calendar* CIF video.

The context dependency in this context model only exists among different coding elements, which is shown in Fig. 2. That is *significant_flag* depends on *N*, *bin0* depends on *N* and *significant_flag*, and *bin13* depends on *bin0*. So we call this type of context model as *hierarchical dependency context model* (HDCM).

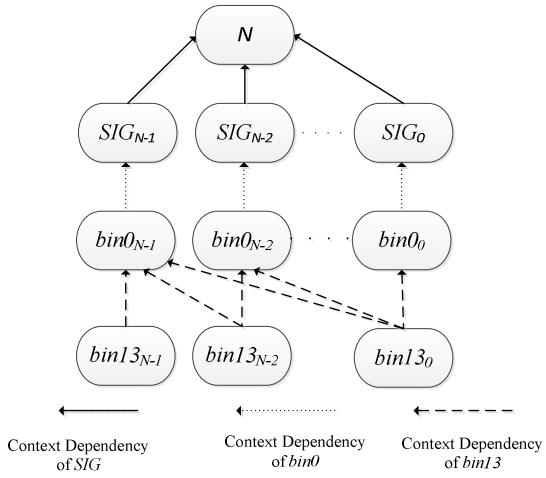

Fig. 2. Context dependency in the HDCM.

4. BINARY ARITHMETIC CODING USING HDCM

4.1. Implementation of binary arithmetic coding

The number of non-zero coefficients (*N*) is used for separating the coding of *significant_flag* from coding of *last_significant_flag*. Therefore, the coding elements consist of *N*, *significant_flag*, *bin0* and *bin13*, and the coding order is *N*→*significant_flag*→*bin0*→*bin13* due to the context dependency in HDCM.

Since *N* is non-binary valued symbol, it is mapped into a string of bins by unary codes. The contexts used for each

bin of N consist of its bin index and prediction $\hat{N}$ of N, which is defined as $\hat{N} = (N_U + N_L)/2$, where N_U and N_L are the numbers of non-zero coefficients of its two neighboring blocks on the top and on the left of the current DCT block to be coded.

To code these coding elements, binary arithmetic coding based on HDCM (HDCMBAC) is designed, in which the binary arithmetic coding engine M-coder of H.264/AVC [2] is adopted. In M-coder, the probability of a symbol to be coded is represented by (p_{LPS}, V_{MPS}), in which p_{LPS} is the probability of the least probable symbol (*LPS*) and V_{MPS} is the value of the most probable symbol (*MPS*); and the probability range associated with *LPS* is projected into a set of representative values $S_p = \{p_0, p_1, ..., p_{63}\}$. Each representative probability p_σ, $0 \le \sigma \le 63$, is implicitly addressed by its index σ in M-coder.

Therefore, the context modeling in (4) and (5) is actually a mapping:

$$C_X : (P, N) \rightarrow \{(\sigma, V_{MPS}) \mid 0 \le \sigma \le 63, V_{MPS} \in \{0,1\}\} \qquad (6)$$

where C_X represents the context modeling for *significant_flag* or *bin0*.

The number of contexts for each coding element is up to B^2 for a DCT block with block size equal to B according to the context model. Such as there are 256 different contexts for each coding element in a 4x4 DCT block. So many contexts will increase the model cost and may cause the context dilution problem.

To solve this problem, the contexts for each coding element are quantized by combining some contexts with similar statistics. Take 4x4 DCT block as an example, the context set for N is partitioned into four intervals: [0, 2), [2, 4), [4, 8) and [8, 16] by $\hat{N}$; for each interval, the context index is determined by the bin index. The context set for *significant_flag* is partitioned into four intervals: [0, 3), [3, 5), [5, 10) and [10, 16] by N; for each interval, the context index is determined by the scanned position. The number of contexts used for *bin0* is reduced to four, which is $\{(P, N)\} \rightarrow \{0, 1, 2, 3\}$.

4.2. Potential parallelism of HDCMBAC

The complete description for encoding a DCT block using HDCMBAC is stated in Table 1. For the parallel processing between context modeling and arithmetic decoding, context index of *significant_flag* is always input to the arithmetic decoder before its decoding is finished. If the decoded bin indicates the decoding of *significant_flag* is finished, the arithmetic decoder can be reset and context model the level information at the same time.

Fig. 3 illustrates the parallel organization between context modeling and arithmetic decoding in HDCMBAC, in which *R&D* module executes reading code stream and decoding a bin, *Undo* module executes resetting the arithmetic decoder and C_X denotes the context modeling for *significant_flag*, *bin0* and *bin13*. It can be seen that *R&D* and C_X can be carried out in parallel. For example, when *R&D* circled by red square is executed for *bin0* of *coeff_abs_level_minus1[1]*, context modeling circled by green square for *bin0* of *coeff_abs_level_minus1[0]* can be carried out at the same time because the prerequisite parameters like N and P_0 are available at that moment.

Table 1. Description for Encoding a DCT block by HDCMBAC

Input: *N*, *significant_flag*, *bin0* and *bin13* in a DCT block

```
Initialization: Num_sig_coeff = 0;
Encode N;
For(i = 0;i < B;i++){
 Encode significant_flag[i];
 If(significant_flag[i]==1)
     Num_sig_coeff += 1;
 If(Num_sig_coeff==N)
    break; }
For(i = B – 1; i >= 0;i++){
   If(significant_flag[i]==1)
      Encode bin0 of coeff_abs_level_minus1[i];}
For(i = B – 1; i >= 0;i++){
   If(coeff_abs_level_minus1[i] > 0)
      Encode bin13 of coeff_abs_level_minus1[i];}
```

5. EXPERIMENTAL RESULTS

The following experiments adopt H.264/AVC reference software Jm15.1 as the test platform, and the 4x4 DCT defined in H.264/AVC is used. The coding configurations for CABAC and HDCMBAC consist of 5 reference frames, 1/4-pixel motion vector resolution, +/-32 pixel motion search range, and R-D optimization enabled. The test videos include *Mobile*, *Foreman*, and *Paris* in CIF; *City*, *Crew* and *Ice* in 4CIF; and *City*, *Bigships* and *Cyclist* in 720p; *Cactus*, *BasketballDrive* and *BQTerrace* in 1920x1080; *Traffic* and *PeopleOnStreet* in 2560x1600. The *QP*s values 16, 20, 24, 28, 32 and 36 are used.

5.1. Coding performance of HDCMBAC

To evaluate the coding performance, the experiments are conducted under three configurations, which include all intra, IPPP and IBBP coding structures. The *BD-Rate* under each configuration is shown in Table 2, in which *BD-Rate* in *Low QP* column is calculated at QPs 16, 20 24 and 28, while *BD-Rate* in *High QP* column is calculated at QPs 24, 28, 32 and 36. It can be seen that HDCMBAC can achieve the similar coding performance as CABAC.

5.2. Analysis of parallelism in HDCMBAC

Speedup is used to analyze the parallelism in HDCMBAC. Since HDCMBAC and CABAC can be implemented using the basic instructions in assembly language (e.g. *Add*, *Move*, *Shift*, *Subtract*, *Compare*, etc.), the number of clock cycles consumed by the basic instructions in assembly language is used to measure the execution time.

$$S_p = \frac{T_{CABAC}}{T_{HDCMBAC}} \qquad (7)$$

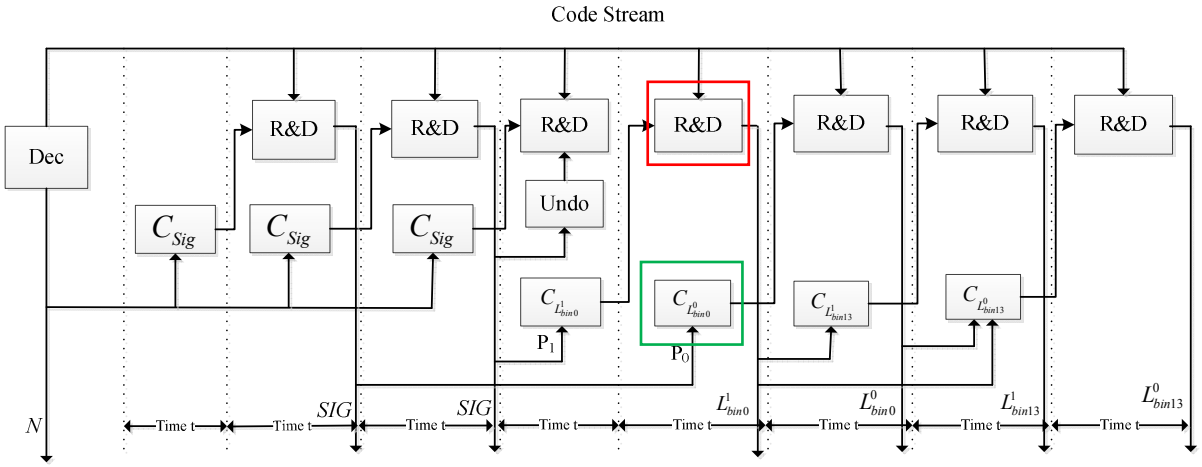

Fig. 3. Parallel organization of context modeling and entropy decoding in HDCMBAC.

The speedup of HDCMBAC relative to CABAC is defined in (7), in which p is the number of processors; T_{CABAC} and $T_{HDCMBAC}$ are the number of clock cycles consumed by CABAC and HDCMBAC, respectively.

The context modeling and arithmetic decoding for *significant_flag* and *last_significant_flag* in CABAC must be carried out serially, since they are interleaved. So the execution time in CABAC is the sum of context modeling and arithmetic decoding. According to the parallel organization applied to HDCMBAC in Section 4, two processors (p = 2) are used to achieve the parallelism between context modeling and arithmetic decoding. So the execution time in HDCMBAC is the maximum one between context modeling and arithmetic decoding.

Table 3 presents the speedup of HDCMBAC relative to CABAC under IBBP coding structure. The speedup is not very large at different QPs, because the arithmetic decoding dominates the decoding process of HDCMBAC and CABAC. If the execution time of arithmetic decoding could be reduced, the speedup would become larger.

6. CONCLUSION

We proposed an hierarchical dependency context model (HDCM) to exploit the statistical correlations of DCT coefficients, in which the number of non-zero coefficients and scanned position are used as the contexts. Then a binary arithmetic coding scheme based on HDCM (HDCMBAC) is presented, in which the number of non-zero coefficients, significant map and level information for a DCT block are used as coding elements. Experimental results demonstrate that HDCMBAC can achieve the similar coding performance as CABAC. Meanwhile the context modeling and arithmetic decoding in HDCMBAC can be carried out in parallel due to the hierarchical context dependency in HDCM.

7. ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation of China (NSFC) under grants 61272386 and 61390513, the Program for New Century Excellent Talents in University (NCET) of China (NCET-11-0797), and the Fundamental Research Funds for the Central Universities (Grant No. HIT.BRETIII.201221).

Table 2. BD-Rate increasing [%] of HDCMBAC Relative to CABAC

Sequences	BD-Rate[%]					
	Low QP			High QP		
	All intra	IPPP	IBBP	All intra	IPPP	IBBP
Mobile	-0.5	0.2	0.1	-0.2	0.3	0.4
Paris	-1.2	-0.7	-0.8	-0.7	0.1	0.0
Foreman	-0.2	-0.1	0.0	0.0	-0.2	0.2
City@4CIF	-0.2	0.4	0.4	-0.1	-0.2	0.1
Crew	-0.3	0.2	0.3	-0.5	-0.2	0.0
Ice	-1.1	-0.7	-0.5	-0.1	0.0	0.1
City@720p	-0.1	0.4	0.4	0.1	0.0	0.4
Bigships	-0.1	0.7	0.6	0.1	0.2	0.2
Cyclist	0.0	-0.1	0.1	0.0	-0.3	0.1
BasketballDrive	0.2	0.3	0.5	0.0	-0.1	0.0
BQTerrace	-0.5	0.9	0.9	-0.7	0.3	0.2
Cactus	0.0	0.8	0.7	0.0	0.1	-0.1
PeopleOnStreet	-0.2	-0.3	-0.3	-0.1	-0.1	0.0
Traffic	-0.6	-0.5	-0.5	-0.2	0.1	0.0
Average	**-0.34**	**0.11**	**0.14**	**-0.17**	**0.00**	**0.11**

Table 3. Speedup of HDCMBAC Relative to CABAC

QP / Sequence	16	20	24	28	32	36
Mobile	1.17	1.18	1.18	1.18	1.18	1.18
Paris	1.17	1.17	1.17	1.15	1.18	1.13
Foreman	1.16	1.16	1.18	1.19	1.17	1.18
City@4CIF	1.19	1.19	1.18	1.17	1.17	1.17
Crew	1.19	1.18	1.16	1.15	1.14	1.13
Ice	1.19	1.17	1.17	1.17	1.17	1.16
City@720p	1.19	1.19	1.18	1.17	1.16	1.15
Bigships	1.19	1.18	1.18	1.17	1.17	1.15
Cyclist	1.19	1.17	1.16	1.15	1.16	1.13
BasketballDrive	1.19	1.18	1.18	1.16	1.16	1.16
BQTerrace	1.18	1.19	1.19	1.19	1.17	1.16
Cactus	1.19	1.19	1.18	1.16	1.16	1.15
PeopleOnStreet	1.18	1.15	1.12	1.13	1.12	1.15
Traffic	1.19	1.18	1.18	1.17	1.16	1.15
Average	**1.18**	**1.18**	**1.17**	**1.17**	**1.16**	**1.15**

8. REFERENCES

[1] A. Moffat, R. Neal, and I. Witten, "Arithmetic coding revisited," IEEE International Conference on Data Compression Conference, pp. 202-211, Mar 1995.

[2] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 620-636, July 2003.

[3] T. Wiegand, G. J. Sullivan, G. Bjontegaard, 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, July 2003.

[4] L. Zhang, Q. Wang, N. Zhang, D. Zhao, X. Wu and W. Gao, "Context based entropy coding in AVS standard," Signal Processing: Image Communication, vol. 24, no. 4, pp. 263-276, April 2009.

[5] L. Yu, S. Chen, and J. Wang, "Overview of AVS-video coding standards," Signal Processing: Image Communication, vol. 24, no. 4, pp. 247-262, April 2009.

[6] J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1649-1668, Dec 2012.

[7] J. Sole, R. Joshi, N. Nguyen, T. Ji, M. Karczewicz, G. Clare, F. Henry, and A. Duenas, "Transform Coefficient Coding in HEVC," IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1765-1777, Dec. 2012.

[8] V. Sze and M. Budagavi, "High throughput CABAC entropy coding in HEVC," IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1778-1791, Dec 2012.

[9] E. Y. Lam and J. W. Goodman, "A mathematical analysis of the DCT coefficient distributions of images," IEEE Trans. Image Process, vol. 9, no. 10, pp. 1661-1666, Oct. 2000.