Context-Based 2D-VLC for Video Coding

Qiang Wang[1], Debin Zhao[1,2], Siwei Ma[2], Yan Lu[1], Qingming Huang[2], Wen Gao[2]
[1]*Department of Computer Science, Harbin Institute of Technology, Harbin 150001, China*
[2]*Institute of Computer Technology, Chinese Academy of Science, Beijing 100080, China*
{qwang, dbzhao, wgao, qmhuang, ylu, swma}@jdl.ac.cn

Abstract

It is well observed that usually the run-length of successive zero coefficients becomes longer and the magnitude of non-zero coefficients gets smaller while the DCT subband frequency increases. So the probability distributions of level/run combinations should be different at different DCT positions. Based on this observation, an efficient context based 2D-VLC entropy coder, especially designed for 8x8 DCT, is proposed in this paper to exploit this context characteristic. The key element of the proposed coder is to design multiple 2D-VLC tables and each can be used context-adaptively to better match different level/run combination probability distributions. Another key element is the usage of Exponential-Golomb codes, which keep the multiple tables with low memory requirement. In terms of coding efficiency, the experimental results show that the proposed method can gain up to 0.23dB when compared to one-table-for-one-block coding method.

1. Introduction

Block-based hybrid video coding scheme has been widely testified in real applications as an efficient coding solution and has been used successfully as the basic coding framework in H.26x and MPEG families. On entropy coding part, the basic concept is mapping from video signal after prediction and transform to a variable length coded bitstream, generally referring to two entropy coding methods, either variable length coding (VLC) or arithmetic coding. For the request of higher coding efficiency, context-based adaptive entropy coding technique is developed and favored by current coding standards, which led to the need of tracking the symbol probability variation to utilize higher order entropy.

Early in H.263 [1], an optional arithmetic entropy coder is specified in normative part, which contains multiple probability distribution models selected automatically in coding process. It expresses a simple context adaptive mechanism but certainly not a true ability of tracking probability variation. After that, a sophisticatedly designed arithmetic coder, named Context-Based Adaptive Binary Arithmetic Coding (CABAC) [2] was proposed and adopted by H.264 [3]. CABAC combines an adaptive binary arithmetic coder with many well-designed context models to fulfill adaptation to symbol statistical behavior and results in significant coding efficiency improvement. But for the increased computational complexity, although good at context adaptation, arithmetic-based entropy coders are still restricted for some applications [2]. As the counterpart, VLC methods also import some adaptability. MPEG-2/4 [4][5] uses different VLC tables for intra- and inter-predicted blocks. H.263 adds optional advanced INTRA coding mode and alternative INTER VLC mode [1] to gain some adaptation. But all these methods use one single VLC table to code a DCT block, and neglect the redundancy or context relationship between the block coefficients. This context information is not fully exploited for further compression.

In DCT block, it is a well observation that the magnitude of non-zero coefficients in low frequency subbands is usually bigger than that in high frequency subbands. Hence, when DCT block is mapped into one-dimensional level/run combination sequence by zig-zag scan, the sequence exhibits decreasing tendency for level's magnitude and increasing tendency for run. This indicates a kind of very useful context information for further compression. Context-based adaptive variable length coding (CAVLC) [3] designed for 4x4 DCT makes use of this context to reduce inter-coefficients redundancy and gets good performance. The method codes level and run separately using corresponding multiple VLC tables and make table selection based on already coded information. The aim of the proposed coding method in this paper is also to exploit this context information, but realization means is different with CAVLC. Our scheme utilizes joint probability of level/run combination by two dimensional VLC which is especially designed for AVS (China audio-video coding standard targeting at HD applications), and so at the same time avoid relative high table memory requirement if CAVLC is directly realized on 8x8 block which is the transform matrix size in AVS. For exploiting context information, we use multiple conditional-trained 2D-VLC tables to better match different level/run combinations' probability distributions at different coding phase by automatic table switch.

The remainder of the paper is organized as follows. In Section 2, we give a detailed description of context-based 2D-VLC. Section 3 provides the experimental results and

0-7803-8603-5/04/$20.00 ©2004 IEEE.

performance comparison. Finally, section 4 concludes this paper.

2. Context-based 2D-VLC

This section gives a detailed description of context-based 2D-VLC, and also provides information on the underlying idea. The outline of the proposed coder is given first, and then the key elements are discussed separately.

2.1. Coder discription

Transformed coefficients are classified into three categories, inter_luma, intra_luma and chroma (both inter and intra), according to macroblock prediction type and video data component type. For each upper three categories, we trained multiple 2D-VLC tables on typical HD sequences and common video quality range. Figure 1 shows the block diagram of context-based 2D-VLC when coding a DCT block. First, table category to be used is determined. Second, an (level, run) instance is coded in a reverse zig-zag scan order using current table denoted by the variable *tablenum*, except that the (level, run) is out of table's range and escape coding method is used. Third, dependent on the last coded level's magnitude, the table for coding the next (level, run) is selected. Then upper last two procedures are carried on iteratively until EOB (End of Block) occurs. Specially, the reason for coding in the reverse zig-zag scan order is that generally the magnitude of the last non-zero coefficient is equal to 1 which presents a constant context characteristic.

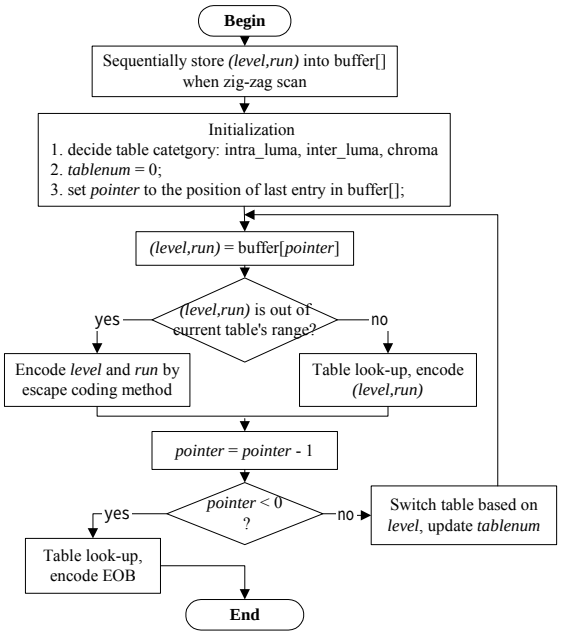

Figure 1. **Flow diagram of context-based 2D-VLC**

In summary, our proposed coder contains following main technical points:

- 2D-VLC for level/run combinations;
- three groups of multiple tables for inter_luma, intra_luma and chroma coefficients, respectively;
- coding in a reverse zig-zag scan order;
- table switch based on last coded level;
- escape coding;
- Exponential-Golomb codes for all elements.

2.2. Exponential-Golomb codes

For every 2D-VLC table, codewords are constructed based on Exponential-Golomb codes [6]. Specifically, k^{th} order Exp-Golomb codes for k equal to 0, 1, 2 and 3 are used. Table 1 lists part of the codewords (Exp-Golomb codes) when k values 0 and 1 as a sample. We can see that a codeword has regular code structure, which is a concatenation of a prefix and a suffix code. Given a codenumber N and specific order k, the prefix part consists of l zeros followed by one 1 and the suffix part is the binarization representation of value $N-2^k(2^l-1)$. l is given by

$$l = \min\{0, \lceil \log_2((N+1)/2^{k+1}+1/2) \rceil\} \quad (1).$$

Table 1. **0^{th} and 1^{st} order Exp-Golomb codes**

Codenumber (N)	Codeword			
	k=0		k=1	
	prefix	suffix	prefix	suffix
0	1	-	1	0
1	01	0	1	1
2	01	1	01	00
3	001	00	01	01
4	001	01	01	10
5	001	10	01	11
6	001	11	001	000
…	…	…	…	…

Owe to the regular codeword structure, Exp-Golomb codes can be real-time constructed in coding process without involving high computation complexity. So the entries stored in VLC tables could be mapping relationships (codenumbers) from level/run combinations to codewords instead of real codes like Huffman codes in MPEG-2. It is a valuable feature that resolves the problem of high memory requirement for multiple 2D-VLC tables. Therefore, although multiple tables are used, memory requirement is still kept low.

2.3. 2D-VLC tables

2.3.1. Table design. It has been stated previously that when zig-zag scan translates a DCT block into one sequence of level/run combination the sequence always demonstrates a phenomenon that, tracked in reverse order, the magnitude of level increases while that of run decreases. Looking into this further, we can find if select a (level, run) in middle of the sequence as a partition point, then the (level, run)s following it usually has bigger size of level and smaller size of run than that of preceding (level, run)s. So the probability distributions of (level, run) before and after that point are basically different. This forms a kind of context information,

which will be very helpful for further compression. Based on upper analysis, a simple and straightforward idea is to find out these typical probability distributions and design corresponding different VLC tables to better match them, which will result in further efficiency. Intuitively an occurrence of increase of the magnitude of level, here increase defined as bigger than ever (the meaning of 'increase' in following paragraphs is the same as this definition), can serve as an indicator of probability distribution's significant change. So depending on level increase, we track the probability distributions and find the typical ones.

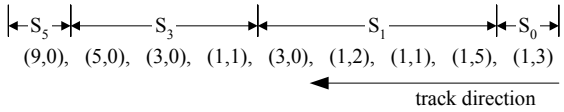

Figure 2. **A sample of tracking to build sets S_T**

Refer to a (level, run) sequence as (l_n, r_n), …, (l_0, r_0), which is indexed in reverse zig-zag scan order, and define sets S_T as follows,

$$S_0 = \{(l_i,r_i) \mid i=0 \} \quad (2),$$

$$S_T = \{(l_i,r_i) \mid k\le i\le k+m,\ m\ge 0;\ (l_{k+m},r_{k+m}), \ldots, (l_{k+1},r_{k+1}), (l_k,r_k),\ \mathrm{abs}(l_{k-1})=T,\ \mathrm{abs}(l_{k+m})>T,\ \mathrm{abs}(l_p)\le T,\ k\le p\le k+m-1 \},\ T>0 \quad (3).$$

At the same time, we suppose sets S_T have functionality to count the occurrence of level/run instance in it. So, when tracking the sequence from the end to the head, we can place any (level, run) into one and only one of upper sets. By the definition we can see the sets S_T contain the (level, run)s locating between two successive increase of the level including the (level, run) bringing the second increase. Figure 2 demonstrates how a (level, run) sequence is placed into sets S_T.

To obtain typical probability distributions robustly, it is on a set of test sequences representing typical material of HD application and at a range of acceptable visual quality of about 30 to 40dB that we involve a two-step process.

Step1. Do statistic to get valuable distributions.

Tracking all (level, run) sequences produced in coding process in reverse zig-zag scan order to build sets S_T, we obtain sets S_0, S_1, …, S_{19}, S_{upper}. S_{upper} is equal to $S_{20} \cup S_{21} \cup \ldots$. Then twenty-one probability distributions of (level, run), referred to as pmf_i, $i=0\sim20$, can be computed corresponding to these sets since these sets can memorize the (level, run) occurrence count. We think these *pmf*s are valuable distributions based on the hypothesis that an increase of level indicates a significant distribution change. For the use of step 2, the optimal k^{th} order Exp-Golomb codes are selected for each upper *pmf*.

Step2. Merge to get typical distributions.

It is not economical and necessary to design twenty-one VLC tables corresponding to each upper *pmf*, because many of them are similar. So a merging process is used to combine similar sets or *pmf*s into a typical one. The rule for the merging is: the *pmf*s or sets to be merged a) have the same k^{th} order Exp-Golomb codes, b) have similar sorting result in terms of (level, run)'s probability, and c) are adjacent to each other.

Through step 1 and 2, we get the typical probability distributions, which depict the occurring behavior of (level, run) symbol at different conditions ("condition" is based on the maximal magnitude of previous levels or level increase) when reverse tracked. For each of these typical distributions, we design its VLC table by selecting an optimal k^{th} order Exp-Golomb codes as the table codewords, and then we eventually finish building multiple 2D-VLC tables, which tell how a 2D (level, run) symbol is mapped to an Exp-Golomb code. Figure 3 shows the different covering range of (level, run) symbols by different multiple 2D-VLC tables, which brings improved coding performance. Specifically, in our design of chroma tables, we get five merged sets as follows, S_0, S_1, S_2, $S_3 \cup S_4$ and $S_5 \cup \ldots \cup S_{upper}$. Corresponding to these merged sets, we accordingly have five 2D-VLC tables for chroma coefficients coding.

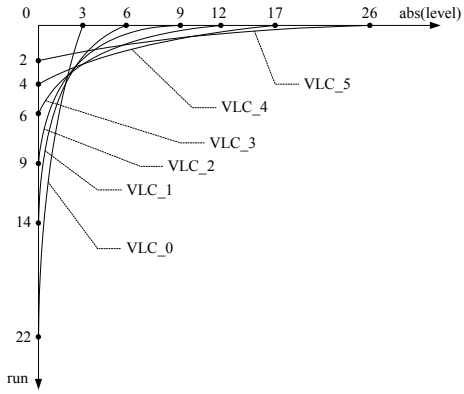

Figure 3. **(level, run) ranges covered by different VLC tables**

2.3.2. Table switch. We know that if a VLC table can better match current coding symbol's probability then further efficiency will be achieved. So the table switch method is consistent with how the table is obtained in statistical process. Therefore, table switch is based on the magnitude of last coded level information just as specific set S_T is selected to adopt current (level, run) symbol by observing whether there is a level increase in statistical process. For instance of chroma coefficients coding, table switch or table selection method is as follows.

```
int T[] = {0, 1, 2, 4, 0xffff};
tablenum = 0;  //initialize
for(totalcoeff; totalcoeff>0; totalcoeff--){  //code DCT block

   Encode_symbol(tablenum, totalcoeff, level, run);
                                          //code a level/run
   if(abs(level) > T[tablenum]) { //table switch
      if(abs(level)<=2)
         tablenum = abs(level);
      else if(abs(level)<=4)
         tablenum = 3;
      else
         tablenum = 4;
   }
}
```

We can see the specific VLC table for coding next (level, run) is selected when a coded level's magnitude exceeds the thresholds 0, 1, 2, 4 in array T[]. These values correspond to the bounds of the merged sets presented previously.

2.4. Escape coding method

The most commonly occurring level/run combinations are coded by the 2D-VLC tables. For other (level, run)s, an escape coding method like MPEG-2 is used. But the difference is an Escape VLC is followed by two variable length codes representing level and run respectively, not two fixed length codes in MPEG-2.

3. Experimental results

To evaluate the coding efficiency, our experiments compare the performance of the proposed coder with that of so-called one-table-for-one-block method, which means only one table is used for coding one DCT block. Specifically, the 2D-VLC tables for 8x8 block depicted in [3] for ABT [7] serve as our method's counterpart. These two methods are both integrated into RM4.0 platform, which is developed by AVS as reference software. The coding condition is two GOPs per second, and in one GOP the first frame is I-frame coded and the other frames are coded in the order of two B-frames and one P-frame. The two test sequences, city and spin&calendar, are all 720p format progressive sequences with 60Hz.

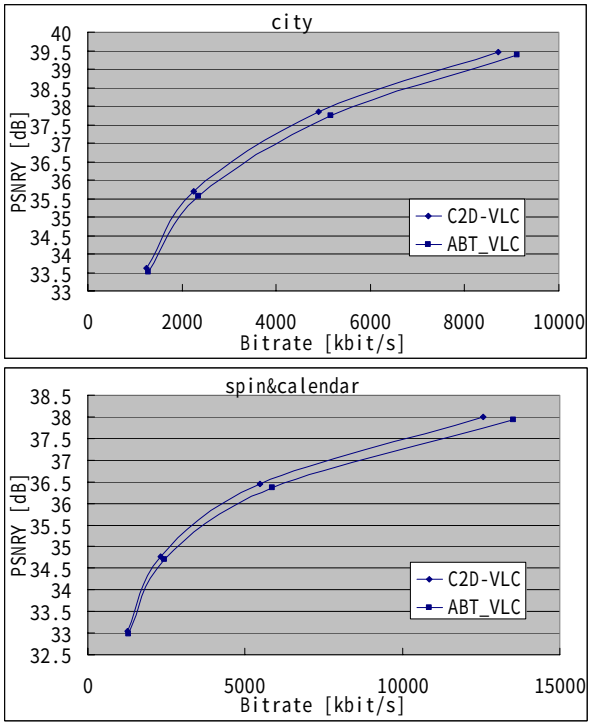

Figure 4. **Rate-distortion performance curves for city and spincalendar**

Figure 4 shows the proposed method outperforms ABT 8x8 tables and for city sequence we can gain up to 0.23dB. Since ABT also adopts Exp-Golomb codes, so all of the performance gain comes from context-based adaptive multiple tables coding.

4. Conclusion

This paper has presented an efficient context-based 2D-VLC entropy coder for transformed coefficients coding. The experimental results have shown that better coding efficiency can be achieved comparing to one-table-for-one-block coding method. The performance improvement comes from two aspects, one of which is multiple tables better match the different probability distributions of level/run combination at different DCT subbands, and another is automatic context-based table selection avoiding transmitting side-information. Exp-Golomb codes also serve as a key element in the proposed coder to keep low memory requirement when multiple tables are used.

5. Acknowledgements

This work has been partially supported by National Science Foundation of China under contract No.60333020, and National Hi-Tech Development Program of China under contract No.2002AA119010.

6. References

[1] "Video Coding for Low Bit Rate Communications", ITU-T, ITU-T Recommendation H.263 version 1, 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, pp. 620-636, July 2003.

[3] T. Wiegand, "Joint Final Committee Draft (JFCD) of Joint Video Specification", in Joint Video Team Doc. JVT-D157, Klagenfurt, Austria, July 2002.

[4] "Generic Coding of Moving Pictures and Associated Audio Information – Part 2: Video", Tech. Rep., ISO/IEC 13818-2 (MPEG-2), 1994.

[5] "Coding of audio-visual objects – Part 2: Visual", ISO/IEC 14496-2 (MPEG-4), 1999.

[6] J. Teuhola, "A Compression Method for Clustered Bit-Vectors", *Information Processing Letters*, Vol. 7, pp. 308-311, Oct. 1978.

[7] M. Wien, "Variable Block-Size Transforms for H.264/AVC", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 604-613, July 2003.