PERFORMANCE-COMPLEXITY ANALYSIS OF HIGH RESOLUTION VIDEO ENCODER AND ITS MEMORY ORGANIZATION FOR DSP IMPLEMENTATION

Zhigang Yang[1], Wen Gao[1,2], Yan Liu[1]

[1]Department of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2]Institute of Computing Technology, Chinese Academy of Science, Beijing 100080, China
{zgyang, wgao, liuyan}@jdl.ac.cn

ABSTRACT

This paper first analyses the relationship between performance and complexity of several state-of-the-art coding algorithms for high resolution videos. Based on the coding efficiency comparison under different config parameters and the intra mode usage in P/B frame, this paper presents a practical scheme to improve the coding speed with slight quality loss. And a DSP-oriented two-level internal memory organization is also proposed to keep pipeline processing. In such organization, block correlation caused by motion vector predictions is lightened while keeping almost the same performance as the original[1].

1. INTRODUCTION

Due to the increasing interests in digital TV and other multimedia applications, video compression has greatly progressed. The state-of-the-art video coding standards, like H.264 [1] and AVS [2], can approximately save 50% bit rate than the prior standards. However, this high coding efficiency is based on high coding complexity. As shown in Fig.1, the framework of H.264 and AVS is transform/ prediction hybrid based. In order to be more accurate, several direction intra predictions and joint motion compensation (MC), such as variable block-size MC, quarter-sample-accurate MC, and multiple reference picture MC, are adopted. And the mode decision rule is not only SAD, but also added with side information such as the bits for coding mode and motion vector. All these algorithms bring much quality gain as well as heavy computation. Performance-complexity analysis can help to find out a practical coding scheme for different video applications.

Obviously, high resolution video coding with these new standards contains tremendous computation, so a powerful platform is necessary. VLSI technology is a solution for such computationally intensive development, e.g. some VGA resolution (640x480) encoders have been implemented based on FPGA [3][4]. Under VLSI architecture, particular algorithms can be highly optimized since hardware is directly designed, but the disadvantage of the approach is the limited adaptability which always results in a redesign. Compared with VLSI, DSP is more flexible. More attention is paid on software optimization, so it's easy for DSP to reprogram and add new modules. Furthermore, special kinds of DSP address the needs of video processing [5], making DSP widely used in video application. Even though, it's still a hard work for real-time high resolution video coding either based on VLSI or DSP, because of tremendous optimizations in all levels.

Generally speaking, there are two aspects affecting the system most, one is computational complexity, and another is communication between processor and memory. M. Ravasi et al give an overview of high-level complexity analysis and memory architecture of multimedia algorithms [6]. If the amount of computation exceeds the maximum ability of processor, multi-processor or low complex algorithm should be considered. In this aspect, we present a tradeoff between coding efficiency and complexity. For the second aspect, it's a common situation that the speed of memory can't catch the speed of processor. Cache can balance the mismatch between them. So a two-level cache organization for DSP is designed to keep pipeline processing.

The rest of this paper is organized as follows: Section 2 lists the performance-complexity analysis of some time-consuming coding algorithms. Section 3 shows the L1/L2 memory hierarchy designed for DSP. Finally, Section 4 concludes the paper.

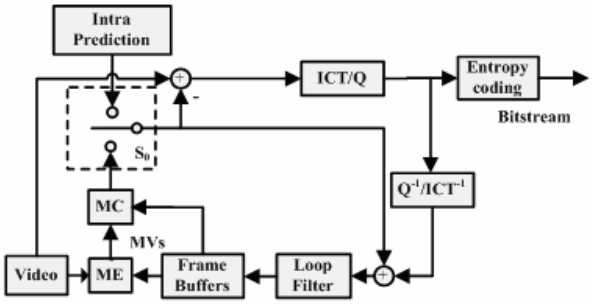

Fig.1. The block diagram of block-based video encoder

[1] This work is supported by National Natural Science Foundation of China No.60333020 and Natural Science Foundation of Beijing No. 4041003.

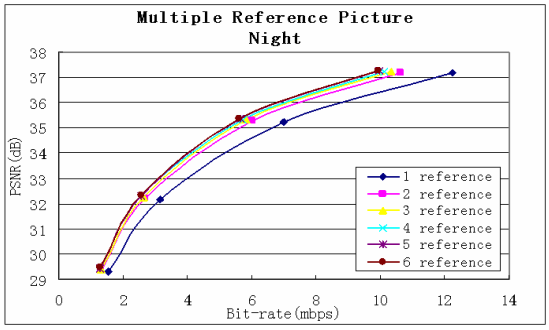

Fig.2. Multiple reference picture performance testing

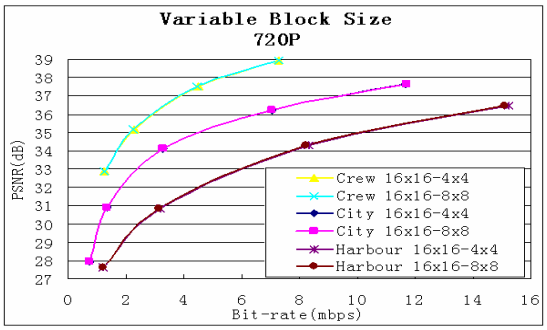

Fig.3. VBMC performance testing

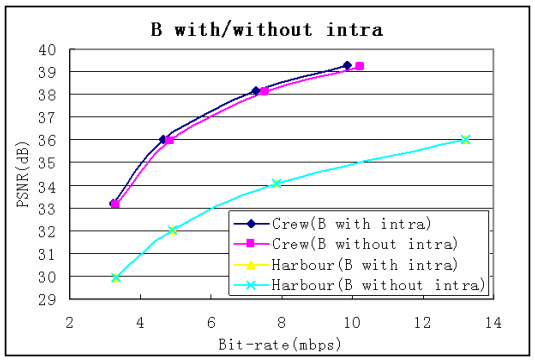

Fig.4. B frame with/without intra mode performance testing

Table 1. Time increment of multiple reference picture

Ref num	1	2	3	4	5	6
Time inc.(%)	-20.38	0	28.77	49.7	80.32	123.29

Note: based on the sequence "Night"

Table 2. Time decrement of variable block size

Seq.	Crew	City	Harbour
16x16-8x8 time dec.(%)	36.1	37.57	36.29
Avg. time dec.(%)		36.66	

Note: assume the time of 16x16-4x4 is 100%

Table 3. Average intra mode usage statistics

QP	Intra mode statistics for P Frame (%)				
	Seq.1	Seq.2	Seq.3	Seq.4	Seq.5
16	2.36	25.65	1.22	7.82	3.46
20	1.62	26.06	1.20	8.15	4.10
24	0.93	23.00	1.11	7.36	3.82
28	0.58	19.22	1.14	7.02	3.02
32	0.45	15.04	1.10	6.49	2.16
36	0.39	11.35	0.97	5.76	1.45
40	0.36	8.08	0.79	4.63	0.79
44	0.33	5.44	0.55	3.33	0.39
48	0.28	2.85	0.28	1.88	0.13
QP	Intra mode statistics for B Frame (%)				
	Seq.1	Seq.2	Seq.3	Seq.4	Seq.5
16	0.453	11.28	0.250	2.609	0.534
20	0.182	9.199	0.158	1.757	0.470
24	0.065	6.984	0.105	0.966	0.334
28	0.018	4.974	0.070	0.507	0.159
32	0.0064	3.803	0.045	0.306	0.064
36	0.0033	2.632	0.024	0.170	0.025
40	0.0014	1.759	0.011	0.078	0.0078
44	0.0008	1.118	0.0032	0.031	0.0015
48	0.0001	0.455	0.0002	0.0076	0.0004

Note: 1-City, 2-Crew, 3-Harbour, 4-Night, 5-Sailormen

2. PERFORMANCE-COMPLEXITY ANALYSIS

Coding efficiency can be achieved by large amount of computation, but the performance and complexity are not in linear ratio. It's meaningless for practical applications to increase much computation to get a little quality gain if the picture's quality has already reached a high level. The following parts are going to find out a reasonable tradeoff.

2.1. Multiple Reference Picture

Multiple reference picture can further improve coding efficiency [7]. In H.264, the number of reference picture can be up to 16. However, in general, two reference pictures usually give almost the best performance and more reference pictures won't bring significant performance improvement except for increasing the complexity heavily. Fig.2 shows the multiple reference picture performance comparison and Table 1 lists the compounding time increment proportion, they further prove the results. This is due to the high temporal redundancy between successive frames. The useful information for motion estimation won't increase much even if more reference frames are adopted.

Moreover, keeping reference pictures is the largest memory spending in video encoder, especially for fractional pixel pictures, so fewer reference pictures also help to reduce storage space and the manufacture cost.

2.2. Variable Block Size

Variable block size motion compensation (VBMC) is very efficient for prediction accuracy and picture quality improvement. In H.264, the block size can be varied from 16x16 to 4x4, each mode need motion estimation, bringing costly computation. Experimental results (Fig.3) illuminate that the block size less than 8x8 doesn't give much improvement for high resolution coding, and Table 2 shows that average 33.66% time is reduced. This is mainly because:

8x4, 4x8, 4x4 blocks are relative too small to carry complete information for objects in high resolution picture, so large blocks usually win during mode decision, resulting in seldom usage of small blocks.

From above analysis, 2 reference pictures and 16x16-8x8 block size are reasonable tradeoff between performance and complexity. Next part describes how much improvement the intra mode can contribute to the whole coding system.

2.3. Intra Mode in P and B Frame

Table 3 lists the intra mode distribution in P and B frame for five 720p sequences coding with AVS. Except for Crew, the intra mode proportion isn't large in P frame for most sequences, and the proportion for B frame is quite small. So intra mode plays an unimportant role in B frame. Moreover, if B frame doesn't process intra prediction, the advantages include not only cutting down intra mode itself, but also omitting the inverse quantization, inverse transform and reconstruction due to no other frames refer to B. Fig.4 shows the B frame with and without intra mode performance comparison, the sequences whose intra mode proportions are the largest (Crew) and the smallest (Harbour) in Table 3 are selected. The quality loss is slight.

But intra mode in P frame can't be omitted, because:

(1) P frame servers as reference frame for others, so no other modules (e.g. inverse quantization, inverse transform and reconstruction) can be omitted like B frame.

(2) When scene change occurs, temporal redundancy is little, inter search can't find good match, then intra coding is needed. If inter mode is still forced to use, bit rate rises and quality goes down until next I frame come.

Then, what happen if scene change occurs when B frame without intra mode:

(1) If scene change happens in P3 (Fig.5a), P3 is mainly coded with intra mode, so B1, B2 mostly refers P3.

(2) If scene change happens in B8 (Fig.5b), B7 mostly refers P6. And P9 is mainly coded with intra mode, so B8 mostly refers P9.

Wherever scene change occurs, the combination of P frame with intra mode and B frame without intra mode can work well together.

Fig.5. Scene change happening in P or B frame

3. MEMORY ORGANIZATION

In real-time coding system, one serious bottleneck is the speed mismatch between memory and processor. One solution is using cache efficiently. But only depending on cache replacement policy by hardware, the encoder can't reach the top coding speed. So we can analyze the data flow to help cache find out the best mapping position, or manage cache by ourselves. Compile-time data caching decisions have a large effect on the performance [8].

The following parts will introduce the two-level internal memory structure of DSP and the detailed memory organization design for motion estimation in turn.

3.1. Two-Level Internal Memory Structure

Fig.6 shows the L1/L2 memory hierarchy of DSP. The cache size of L1P, L1D and L2 is 16, 16 and 256KB respectively. Cache line size is 32, 64,128bytes, and cache miss penalty is 8, 6, 8 cycles respectively. The cache architecture allows pipelining read misses. Multiple parallel and consecutive misses consume only 2 cycles once pipelining is set up [9]. The useful routine "touch"[10] can load data into L1D with minimum cycle penalty by such read miss pipelining. L2 cache can also server as on-chip memory. So we are able to control the data flow in L1D (by touch) and L2 (by DMA) to get better DSP performance.

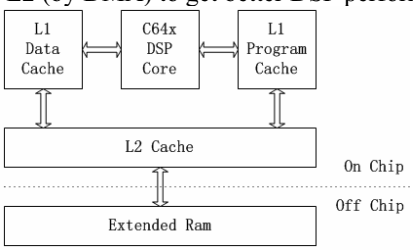

Fig.6. C64x two-level internal memory block diagram

3.2. Memory Organization of ME

Most of video coding efficiency is derived from motion estimation (ME). At the same time, ME contributes the heaviest computational burden for the whole video encoding, therefore different kind of fast search techniques appear. But whatever search method is used, the search area can be sent into cache ahead in order to improve performance, as long as the search range is fixed. For convenience, we assume that search range is ±32.

In H.264 and AVS, every block needs a MV prediction, calculated by four already known MVs of left, up, up-left, and up-right blocks. And MV prediction points the search center of each block. If variable block size is 16x16 to 8x8, there would be total nine different search areas. MV prediction causes great correlation among blocks, shown in Fig.7. The current search area can't be determined until prior blocks' ME is finished. The block correlation limits wide use of background transfer, therefore results in low DSP efficiency.

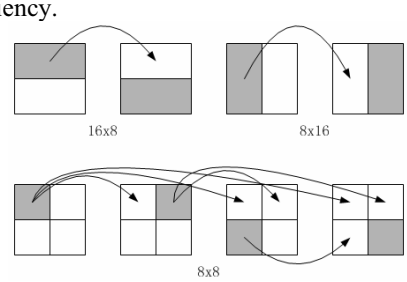

Fig.7. Block correlation caused by MV prediction

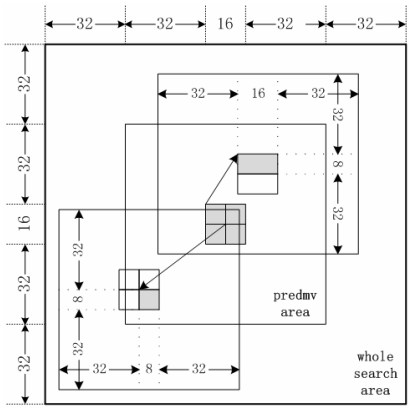

Fig.8. Whole search area for all blocks in one MB

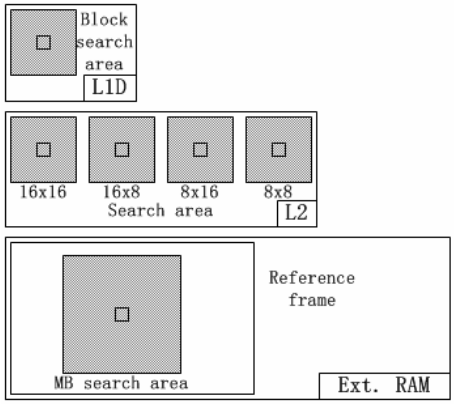

Fig.9. 3-level memory hierarchy for ME

Table 4. Simulation Results of Method B compared with original algorithm

Seq.	\ QP	20	24	28	32	36	40	44	48
Crew	PSNR gain	0.000	0.000	-0.002	-0.004	-0.001	-0.008	-0.009	-0.023
	Bitrate inc.(%)	0.140	0.270	0.620	1.040	1.260	1.930	2.260	2.240
Harbour	PSNR gain	0.000	0.000	0.000	-0.001	-0.002	-0.003	-0.002	-0.005
	Bitrate inc.(%)	0.020	0.020	0.060	0.120	0.340	0.560	0.580	0.710

Table 5. Cycle overlap for loading search area into L1D

Search Range	±32		±48	
Method	A	B	A	B
Area Size (Byte)	21025	26244	43681	51076
L1D Read Misses	329	412	683	800
L1D Stall Cycles	1974	840	4098	1616
Touch Execute Cycles	0	274	0	464
Total Cycle Overhead	1974	1112	4098	2080

Generally, the search center is limited in ±32 square, so all nine search areas are in a larger [-64,16+64] square, shown in Fig.8. The large square's size is $(64+1+16+64)^2$= 21025Bytes>16KB, L1D can't load all the data. Thus, during the period of ME, L1D read misses might occur, interrupting pipelines. [Method A]

In order to further improve the DSP efficiency, all the correlations in Fig.7 are removed. That is to say, blocks of same size use same MV prediction. Experimental results (Table 4) show that the improved method has similar coding performance with the original. Then the overall search area can be broken into four new areas. Each new area data size is $(32+1+16+32)^2$=6561Bytes<16KB, even if the search range is ±48, $(48+1+16+48)^2$=12769Bytes<16KB, the new area can be wholely put into L1D too. This improved method can increase searching speed by 3%~7% depending on different search algorithms. Fig.9 shows the logical diagram of 3-level memory organization for ME. [Method B]

Table 5 lists the cycle overlap for loading search area of one macroblock into L1D by the two mentioned methods.

4. CONCLUSIONS

This paper analyzes several time-consuming parts of high resolution video encoder, such as multiple reference picture, variable block size and intra mode in P/B frame. And then presents a high-quality and middle-complex coding scheme based on the experimental results and statistics. An efficient two-level internal memory organization for DSP is designed in detail as well. And this memory organization is independent of search algorithm. To design and implement 720p and 1080i resolution real-time encoder is our next goal.

5. REFERENCES

[1] T. Wiegand, et al, "Overview of the H.264/AVC Video Coding Standard", *IEEE Trans. CSVT*, Vol. 13, No. 7, pp. 560-576, July 2003.

[2] "Final draft of information technology – advanced coding of audio and video – part 2: video," in AVS workgroup Doc. N1214, Shanghai, China, Sep. 2005.

[3] Y.C. Lu, et al, "Performance-driven Optimization for Video Accelerator Design", in *Proc. ISCAS*, May 2005

[4] K. Denolf, et al, "Memory efficient design of an MPEG-4 video encoder for FPGAs", in *Proc. IEEE Int. Conf. Field Programmable Logic and Applications*,. 2005, pp. 391-396.

[5] TI TMS320DM642, http://www.ti.com

[6] M. Ravasi, et al, "High-Abstraction Level Complexity Analysis and Memory Architecture Simulations of Multimedia Algorithms", *IEEE Trans. CSVT*, Vol. 15, No. 5, pp. 673-684, May 2005.

[7] T. Wiegand, et al, "Long-Term Memory Motion-Compensated Prediction", *IEEE Trans. CSVT*, Vol. 9, No. 1, pp. 70-84, Feb. 1999.

[8] E.D. Greef, et al, "Memory Organization for Video Algorithms on Programmable Signal Processors", in *Proc. IEEE Int. Conf. Computer Design*, Oct. 1995, pp. 552-557.

[9] TMS320C64x DSP Two-Level Internal Memory Reference Guide, SPRU610B, http://www.ti.com/, Aug. 2004.

[10] TMS320C6000 DSP Cache User's Guide, SPRU656A, http://www.ti.com/, May 2003.