LETTER

Real-Time Complexity Control for H.264 Video Encoding by Coding Gain Maximization*

Ming-Chen CHIEN[†], *Nonmember and* **Pao-Chi CHANG**[†a)], *Member*

SUMMARY This research proposes a Coding-Gain-Based (CGB) complexity control method for real-time H.264 video encoding in complexity-constrained systems such as wireless handsets. By allocating more complexity to the encoding tools which have higher coding efficiency, the CGB method is able to maximize the overall coding efficiency of the encoder.
key words: *video coding, complexity control, power control*

1. Introduction

Applications of real-time video encoding are common in modern wireless handsets. Since the computation capability of handsets is generally limited, the allowable computational complexity of encoding a video frame, C_{FC}, is also limited. To achieve high compression ratio, an H.264 video encoder utilizes a number of encoding tools [1]. However, the full-scale encoder is associated with large computational complexity and hence may not be allowed for a complexity-constrained system [2]. Therefore, a mechanism which controls the encoding complexity of each frame under the complexity limit while achieving optimal Rate-Distortion (R-D) performance is necessary. The complexity control problem can be modeled as

$$\begin{aligned} &\min J = \min\{D + \lambda R\} \\ &s.t. \qquad c_F \le C_{FC} \end{aligned} \tag{1}$$

where D denotes the distortion, R denotes the bit rate, λ denotes the Lagrange multiplier, J denotes the R-D cost, and c_F denotes the complexity involved in encoding a frame.

Several studies on complexity control have been conducted [3]–[5]. The first C-R-D model which formulates the complexity allocation among encoding tools was proposed by He et al. [3]. Based on the model, the optimal complexity of Motion Estimation (ME), DCT, and entropy coding can be obtained. However, its optimization formula is too complicated to be derived to a closed-form solution and finding the solution by conducting a global search requires a large computational overhead. Kannangara et al. developed a complexity control algorithm by adjusting the proportion of *SKIP* mode [4]. However, skipping a Macro-block (MB) could sacrifice contributions from the coding efficiency of ME, DCT, and entropy coding in that MB. Su et al. proposed a virtual leaky bucket model to prevent encoding a frame from consuming too much computational complexity or too little [5]. Frame-layer control, however, focuses only on ME without considering other tools.

This work proposes a Coding-Gain-Based (CGB) complexity control method for complexity-constrained H.264 video encoding. It describes the procedure to obtain the coding efficiency of each encoding tool and presents a utilization strategy for these tools. By using the utilization strategy and allocating more complexity to the encoding tools which have higher coding efficiency, the CGB method is able to maximize the overall coding efficiency of the encoder. Though the coding efficiency of encoding tools may vary with different platforms, this research provides a design procedure which allows designers to develop a high efficient complexity control algorithm on different platforms.

In Sect. 2, we analyze the coding efficiency of encoding tools in the H.264 video encoder. In Sect. 3, We propose the CGB complexity control algorithm. We present experimental results in Sect. 4, and draws conclusions in Sect. 5.

2. Coding Efficiency Analysis of Encoding Tools

A metric of Coding Gain (CG) which represents coding efficiency was proposed [7] as follows:

$$CG = \Delta J / \Delta C \tag{2}$$

where ΔC represents the increase in complexity when an encoding tool is adopted and ΔJ represents the decrease in R-D cost. With rate control being utilized to maintain a target rate, the metric of CG can be simplified as

$$CG = \Delta PSNR / \Delta C \tag{3}$$

where $\Delta PSNR$ represents the increase in PSNR, and ΔC can be measured by the number of CPU clock cycles spent on running an encoding tool.

This section conducts CG analysis with the encoding options shown in Table 1. To reflect the condition in real applications, a software optimized H.264 code, ×264, is used [6]. This analysis adopts five parameters s1, s2, s3, s4, and s5 to obtain the CG of various encoding tools in the H.264 encoder. Their meanings are described as the following:

s1: available block sizes for ME (3 - all; 2 - 16 × 16, 16 × 8, 8 × 16, 8 × 8; 1 - 16 × 16, 8 × 8; 0 - 16 × 16);

Manuscript received December 18, 2009.
Manuscript revised November 23, 2010.
[†]The authors are with National Central University, Taiwan. Ming-Chen Chien is also with Asia Pacific Institute of Creativity, Taiwan.
*This work was supported in part by the National Science Council of R.O.C. under Grant NSC 99-2221-E-008-003.
a) E-mail: pcchang@ce.ncu.edu.tw
DOI: 10.1587/transcom.E94.B.2181

Table 1 Options for coding gain analysis.

Video source	Akiyo, Mobile, Suzie, Carphone, Foreman, Stefan QCIF
Fast ME	Diamond
Target rate (bps)	Foreman: 150k, 240k; Akiyo: 60k; Carphone & Suzie: 150k; Stefan: 300k; Mobile: 600k
Frame rate	30
Number of ref frames	1
GOP type	IPPPP
CPU	Intel Pentium 4 2.66 GHz
RAM	512M bytes
MMX tech.	On for SAD computation

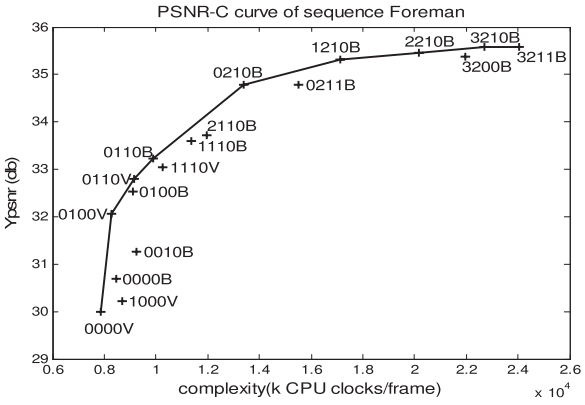

Fig. 1 Coding efficiency analysis of various encoding tools for H.264 video encoders: Foreman 150 kbps.

Table 2 Coding gain of each encoding tool. The unit of *CG* is db/kclks.

Mode transition	Encoding tool	*CG*
0000V->0100V	Half pixel ME	6.1e-3
0100V->0110V	Deblocking filter	5.1e-4
0110V->0110B	CABAC (compare to CAVLC)	4.9e-4
0110B->0210B	Quarter pixel ME	4.4e-4
0210B->1210B	8×8 partition mode	2.0e-4
1210B->2210B	16×8 and 8×16 partition mode	6.7e-5
2210B->3210B	Sub 8×8 partition mode	3.1e-5
3210B->3211B	4×4 Intra	5.9e-6

s2: pixel accuracy of ME (2 - quarter; 1 - half; 0 - full);
s3: use of deblocking filter (1 - use; 0 - not use);
s4: use of intra 4×4 (1 - use; 0 - not use);
s5: type of entropy coding (B - CABAC; V - CAVLC).

Analysis results for Foreman sequence at the rate of 150 kbps are plotted in Fig. 1, where the symbol on the chart represents the combination of parameters (s1 s2 s3 s4 s5). The line marks the optimal path for the encoding tool selection with the highest coding efficiency. Experiments with Foreman sequence at the rate of 240 kbps and other sequences including Carphone and Stefan yield the same optimal path. Experiments with Akiyo, Suzie, and Mobile sequences yield similar optimal path. The only difference is '0100B', instead of '0110 V', on the optimal path indicating that CABAC has higher *CG* than deblocking filter for these sequences. Table 2 presents the average *CG* of each encoding tool in descending order for the six sequences.

To minimize distortion, the encoding tool with greater *CG* should be used earlier. As Table 2 shows, half pixel ME has the highest *CG* and should be utilized first. Deblocking filter should be utilized after half pixel ME and before CABAC. H.264 standard provides 'disable_deblocking_filter_idc' to disable deblocking filter [1]. Because the deblocking filter is applied to the whole frame once it is activated, the use of deblocking filter should be frame-based. In the previous work [8], deblocking filter is always utilized. As a result, the overall *CG* of the encoder is not optimal when C_{FC} is small. This work suggests utilizing a complexity threshold TH_{df} to determine the use of deblocking filter. Only if C_{FC} is greater than TH_{df}, would deblocking filter be used. TH_{df} can be regarded as the complexity of '0110 V' in Fig. 1. Let F_{mv} be defined as

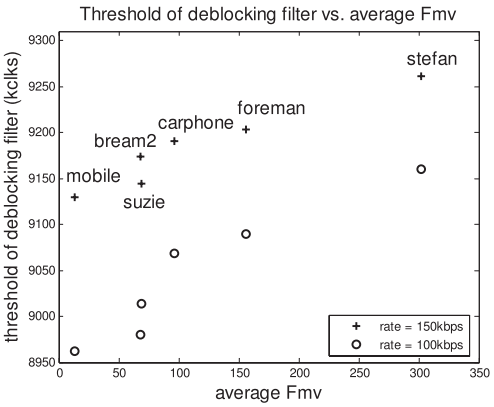

Fig. 2 Threshold of deblocking filter vs. average Fmv.

$$F_{mv} = \sum_{i=1}^{M} (|mvx_i| + |mvy_i|) \tag{4}$$

This represents the motion of a video frame, where (mvx_i, mvy_i) denotes the motion vector difference of ith MB in 16×16 block size and M denotes the number of MBs in a frame. According to our experiments with various video sequences, TH_{df} is proportional to the average value of F_{mv} at a given rate as Fig. 2 shows. Hence it can be modeled as

$$TH_{df} = a \cdot F_{mv} + b \tag{5}$$

where a and b are model constants, which can be obtained statistically by running test video sequences in advance.

CABAC should be utilized after deblocking filter and before quarter pixel ME as suggested in Table 2. If CABAC is utilized when C_{FC} is small [8], the overall *CG* of the encoder is not optimal because higher *CG* tools may not be utilized. As proposed previously [3], the complexity of entropy coding is proportional to the bit rate. Accordingly, this work suggests a threshold TH_{cb} for CABAC utilization which can be obtained by

$$TH_{cb} = TH_{df} + g \cdot R \tag{6}$$

where g is a model constant.

The other encoding tools listed in Table 2 belong to ME. We will suggest their utilization strategy in the next

section.

3. CGB Complexity Control Algorithm

Our algorithm first allocates the allowable complexity C_{FC} from frame layer to MB layer, and then to each encoding tool. The complexity allocation should be performed before the first MB is encoded. In the frame layer, the video encoder performs initialization, encoding slices, deblocking filtering if it is utilized, and updating references. Deblocking filter should be utilized only if C_{FC} exceeds TH_{df}. Let C_{Finit} denote the complexity of the initialization, C_{SLs} denote the complexity of encoding all slices, and C_{Fother} denote the complexity of deblocking filtering if it is utilized and updating references. C_{Finit} can be easily measured. C_{Fother} is much smaller than C_{SLs} as we previously proposed [8], and it does not vary greatly. C_{Fother} can be regarded as a constant and can be estimated from the previous frame. Therefore, by measuring C_{Finit} and reserving C_{Fother}, C_{SLs} can be allocated by

$$C_{SLs} = C_{FC} - C_{Finit} - C_{Fother} \tag{7}$$

The operation of the slice layer is very simple and its complexity is very low. Therefore, the complexity of encoding all MBs in a frame, C_{MBs}, is approximately equal to C_{SLs}.

Each MB can be encoded by ME, DCT, Quantization (Q) and entropy coding. Typically, ME consumes most of the total complexity [8]. Therefore, the complexity of ME should be well controlled. Instead of CAVLC, CABAC should be used if C_{FC} exceeds TH_{cb}. DCT, Q, Q^{-1}, IDCT were collectively denoted by PRECODING [3]. Since H.264 significantly simplifies DCT operation [1], PRECODING has high coding gain and is destined to be adopted. Let $C_{MBother}$ denote the complexity reserved for PRECODING and entropy coding for an MB. As we have proposed [8], $C_{MBother}$ is much smaller than C_{ME}, the complexity of ME for an MB. Therefore, $C_{MBother}$ can be treated as a constant and can be estimated statistically by running test video sequences in advance. As described above, the complexity budget of ME for all MBs, C_{MEs}, can be allocated using

$$C_{MEs} = C_{MBs} - C_{MBother} \times M \tag{8}$$

Since each MB has different motion and texture from each other, allocating C_{MEs} among MBs is critical. Our approach is based on allocating more complexity to the MB where ME yields higher *CG*. The *CG* of ME for an MB can be measured by (2), where ΔC represents the complexity of ME and ΔJ represents the decrease in R-D cost as

$$\Delta J = COST0 - COSTF \tag{9}$$

Here *COST0* represents the cost of ME with zero MV in 16×16 partition mode and *COSTF* represents the final cost of ME. The actual *CG* of ME, however, can only be obtained after ME but the complexity allocation should be performed before ME. This work proposes an approach to predict the

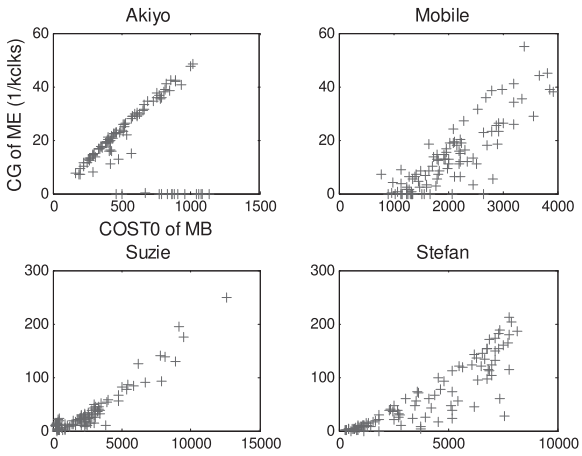

Fig. 3 *CG* of ME vs. *COST0* in 50th frame in various sequences when QP = 28 under little allowable complexity.

CG of ME by *COST0*, which can be easily obtained before ME. To analyze the relationship between the *CG* and *COST0* under different constraints, we simulate two cases: 1) little allowable complexity; 2) relatively sufficient allowable complexity. In the first case, each MB is inter predicted and intra predicted using only 16×16 block size. The experiment environment is shown in Table 1 but QP instead of rate is given. As Fig. 3 shows, the *CG* of ME is roughly proportional to *COST0* for various sequences when QP is 28. Experiments with QP set to 30 yield similar results. In the second case, each MB is inter predicted using 16×16 block size, 8×8 block size, and intra predicted using 16×16 block size. The *CG* of ME is still roughly proportional to *COST0* for various sequences though the *CG* is smaller. Therefore, we propose an efficient allocation as

$$C_{ME}(i) = C_{MEs} \times \frac{COST0_i}{\sum_{j=1}^{M} COST0_j}, \quad i = 1, 2, \cdots, M \tag{10}$$

where $C_{ME}(i)$ means the complexity budget of ME for ith MB.

This work suggests the flow of ME process should be arranged in decreasing *CG* order and divided into several levels as Fig. 4 shows. After an ME level is complete, the used complexity C_{MEused} is examined. If C_{MEused} exceeds $C_{ME}(i)$, the ME process is terminated. Otherwise, the ME process continues. This design ensures the complexity budget is allocated to encoding tools with higher *CG*. In the previous work [8], C_{MEused} is checked after searching a point. If C_{MEused} exceeds the budget, ME process is terminated. That design may make the motion search with the latest partition mode, 8×8 for example, incomplete and waste. In this work, we solve this problem by checking C_{MEused} after a level is complete.

4. Numerical Results and Discussions

This section examines the performance of the proposed complexity control algorithm. The experiment environment

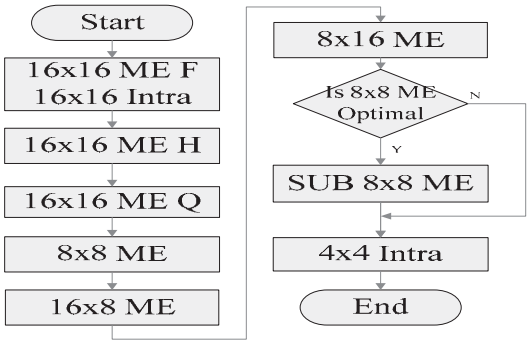

Fig. 4 ME flow in decreasing CG of encoding tools. F, H, Q denotes full pixel, half pixel, and quarter pixel accuracy of ME respectively. The KxN ME without F, H, and Q means using all the three resolutions.

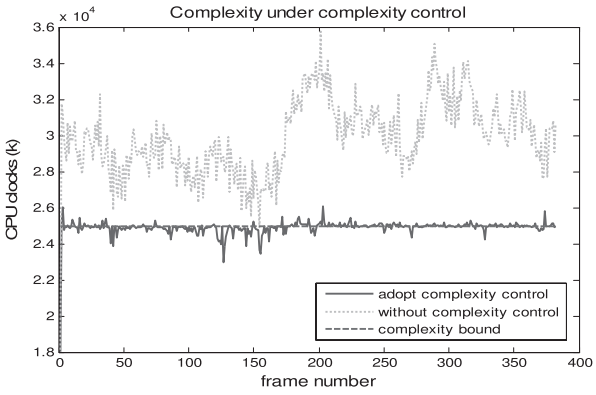

Fig. 5 Computational complexity with and without complexity control when C_{FC} is set to 70% of C_{FM}.

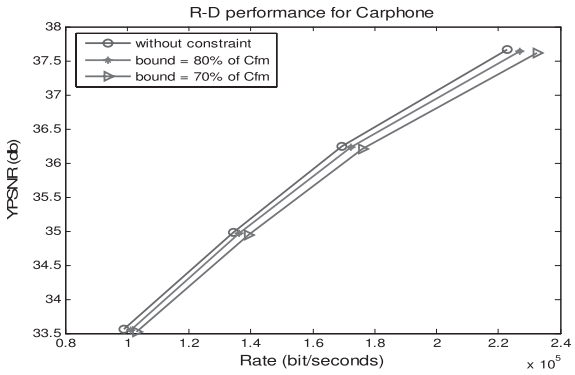

Fig. 6 R-D performance with and without complexity control.

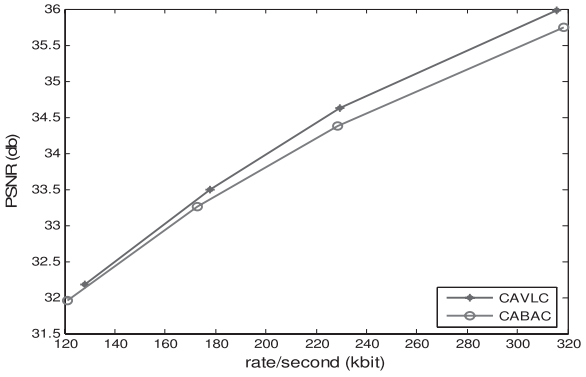

Fig. 7 R-D performance with CAVLC and CABAC when C_{FC} (10 MHz/frame) is less than TH_{cb} for Foreman.

is described in Table 1. The complexity limit C_{FC} is set to 70% ∼ 80% of C_{FM}, the maximum complexity of a frame in a sequence without complexity limit. Figure 5 and Fig. 6 show the results with Carphone sequence. As Fig. 5 shows, our method can well control the complexity. When C_{FC} is set to 80% of C_{FM}, which means the system suffers from a serious insufficiency of allowable complexity, this algorithm yields less than 0.2db YPSNR degradation at any rate as Fig. 6 shows. Experiments with other video sources, including Suzie, Foreman, and Stefan, yield similar results. These results reveal this new method is superior to the method in the previous work [8] because of better complexity control of ME. This section also compares the encoding with our new utilization strategy of entropy coding to the previous strategy [8]. When C_{FC} is less than threshold TH_{cb}, the new strategy which selects CAVLC yields an average gain of 0.2 db to the previous strategy which selects CABAC as Fig. 7 shows.

5. Conclusion

This work proposes an efficient method for controlling the complexity of encoding a frame under a given limit while maintaining excellent R-D performance. The proposed method, which has very low overhead, is excellent for practical use.

References

[1] ISO/IEC ITU-T Rec. H264: Advanced Video Coding for Generic Audiovisual Services, Joint Video Team (JVT) of ISO-IEC MPEG & ITU-T VCEG, Int. Standard, May 2003.
[2] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: Tools, performance, and complexity," IEEE Circuits Syst. Mag., vol.4, no.1, pp.7–28, April 2004.
[3] Z.H. He, Y.F. Liang, L.L. Chen, I. Ahmad, and D.P. Wu "Power-Rate-Distortion analysis for wireless video communication under energy constraints," IEEE Trans. Circuits Syst. Video Technol., vol.15, no.5, pp.645–658, May 2005.
[4] S. Kannangara, I.E.G. Richardson, and A.J. Miller, "Computational complexity management of a real-time H.264/AVC encoder," IEEE Trans. Circuits Syst. Video Technol., vol.18, no.9, pp.1191–1200, Sept. 2008.
[5] L. Su, Y. Lu, F. Wu, S. Li, and W. Gao, "Complexity-constrained H.264 video encoding," IEEE Trans. Circuits Syst. Video Technol., vol.19, no.4, pp.1–14, April 2009.
[6] ×264, Available: http://developers.videolan.org/x264.html
[7] C. Kim and J. Xin, "Hierarchical complexity control of motion estimation for H.264/AVC," Mitsubishi Electric Research Laboratories, TR2006-004, Dec. 2006. Available: http://www.merl.com
[8] M.C. Chien, Z.Y. Chen, and P.C. Chang, "Coding-gain-based complexity control for H.264 video encoder," ICIP 2008, San Diego, US, Oct. 2008.