

MOTION COMPENSATION COMPLEXITY MODEL FOR DECODER-FRIENDLY H.264 SYSTEM DESIGN

Szu-Wei Lee and C.-C. Jay Kuo

Ming Hsieh Department of Electrical Engineering and Signal and Image Processing Institute
University of Southern California, Los Angeles, CA 90089-2564, USA

E-mails: szuweile@usc.edu and cckuo@sipi.usc.edu

ABSTRACT

An improved motion compensation decoding complexity model is proposed and its application to H.264/AVC decoding complexity reduction is examined in this work. This complexity model considers a rich set of inter prediction modes of H.264 as well as the relationship between motion vectors and frame sizes, which turn out to be highly related to the cache management efficiency. An H.264 encoder equipped with the complexity model can estimate the decoding complexity and choose the best inter prediction mode to meet the decoding complexity constraint of the target receiver platform. The performance of the proposed complexity model and its application to video decoding complexity reduction are demonstrated experimentally.

Index Terms— H.264/AVC, motion compensation, decoding complexity, rate-distortion optimization (RDO)

I. INTRODUCTION

H.264/AVC [1] is the latest video coding standard proposed by ITU-T and ISO/IEC. It has been selected as the video coding tool in HD-DVD and Blue-ray specifications. Since H.264 provides various inter prediction modes to improve the coding gain, its encoder may search all possible inter prediction modes and choose the best one that minimizes the rate-distortion (RD) cost. Due to the use of a larger set of inter prediction modes, the H.264 decoding complexity is about 2.1 to 2.9 times more than the H.263 decoder [5]. For some applications, the coded video bit stream will be decoded in portable consumer electronics devices. Under such a scenario, the reduction in decoding complexity so as to save the power becomes a critical issue.

To achieve power saving purpose of the H.264 decoder, one possible solution is that the H.264 encoder can generate a decoder-friendly bit stream. That is, the H.264 encoder has a target decoding platform in mind and can generate a bit stream that is easy to be decoded in that platform. This motivates us to study the decoding complexity model and its applications. This model can be used by the H.264 encoder to estimate the decoding complexity for various inter prediction modes and select the best one to balance the rate-distortion (RD) and decoding complexity tradeoff. Since a general decoding complexity model is too broad to cover, this work mainly considers the complexity model for motion compensation (MCP, the decoding process for inter prediction) operations in H.264.

The decoding complexity models have been studied in the past [2]-[4]. A MPEG-4 video complexity verifier (VCV) model was described in [2], where the numbers of boundary macro-blocks (MBs) and non-boundary MBs decoded per second are estimated and the decoding complexity can be modeled as a function of these two numbers. However, the decoding complexities of the MBs

coded by different inter prediction modes in MPEG-4 can be different so that the VCV model is not very accurate. To address this shortcoming, Valentim *et al.* [3] proposed an enhanced MCP decoding complexity model for MPEG-4 video. By considering the fact that MBs coded with different inter prediction modes have different decoding complexities, they use the maximal decoding time to measure the decoding complexities of MBs coded by different inter prediction modes individually. Then, the total decoding complexity of this bit stream is the sum of each individual MB's complexity. Recently, it was reported in [4] that the MCP decoding complexity model can be simplified and the decoding complexity of MB is proportional to the number of motion vectors (MVs). In other words, the MB with more MVs should have higher decoding complexity than that with fewer MVs.

The above decoding complexity models are however not suitable for H.264 for two reasons. First, H.264 provides a richer set of inter prediction modes that cannot be well handled by the existing models. Second, these models do not take the relationship of MVs into account, which is related to the efficiency of cache management and therefore plays an important role in CPU performance and decoding complexity. More recently, an MCP decoding complexity model for H.264/AVC was proposed to solve the above issues with cache models [6]. Here, we improve the complexity model in [6] and examine its application to H.264 video coding, which allows encoders to generate decoder-friendly bit streams.

II. PROPOSED IMPROVED COMPLEXITY MODEL

A. Existing MCP decoding complexity model

H.264 provides various block sizes for inter prediction to improve the coding gain. One 16x16 MB can be partitioned into one 16x16 block, two 16x8 or 8x16 blocks or four 8x8 blocks. Each 8x8 blocks can be further partitioned into two 8x4 or 4x8 blocks or four 4x4 blocks. Moreover, each block whose size is larger than 8x8 can be predicted using different reference frames. In addition, H.264 provides fractional samples of motion estimation (ME). The MV can be of half- or quarter-pel resolution. The half-pel value is obtained by applying a one-dimension 6-tap FIR interpolation filter horizontally (the x-direction) or vertically (the y-direction). The quarter-pel value is obtained by the average of two nearest half-pel values. Furthermore, H.264 provides the so-called P_Skip mode to code an MB in a large area with slow motion. When one MB is coded as the P_Skip mode, neither MV nor residual error data are transmitted. The MV of the P_Skip mode MB is decided by those MVs of its neighboring MBs. This flexibility makes the decoding complexity modeling more challenging as compared with that in [3] and [4].

Recently, an MCP decoding complexity model for H.264/AVC was proposed in [6] to overcome the difficulties described above. The

decoding complexity is modeled as a function of the number of cache misses N_c , the number of y-direction interpolation filters N_y , the number of x-direction interpolation filters N_x and the number of MVs per MB N_v . Mathematically, it can be written as

$$\alpha \cdot N_c + \beta \cdot N_y + \gamma \cdot N_x + \mu \cdot N_v, \quad (1)$$

where $\alpha, \beta, \gamma, \mu$ are weight coefficients of these four decoding complexity coefficients, respectively. As done in [4], the decoding complexity model includes the number of MVs per MB. The numbers of the x-direction and the y-direction interpolation filters are used to model the decoding complexity of interpolation filter if the MV is of half- or quarter-pel accuracy. Since the decoder may have different implementations of interpolation filters along the x- and y-directions, two different terms are used in the decoding complexity model.

The number of cache misses is included to deal with two cases shown in Fig. 1, where two MVs point to two reference blocks with a different spatial relationship. The two reference blocks are closer to each other in Case I but far away in Case II. Intuitively, the number of cache misses in Case I is fewer than that in Case II since most data required for block B in Case I could be obtained from the CPU cache or internal registers after the decoding of block A is done. Thus, the decoding complexity for block B in Case I should be lower than that in Case II.

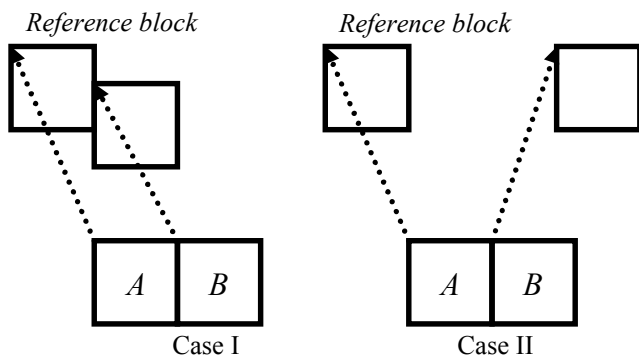


Figure 1. Inter prediction of two successive blocks

The weight coefficients in the model can be obtained by the following steps. First, several pre-encoded bit streams are selected and Intel Vtune performance analyzer is used to measure the number of clock ticks spent by the MCP. Second, the numbers of the four decoding complexity coefficients are counted individually for those pre-encoded bit streams. Finally, the constrained least square method is used to find the best fitting of the weight coefficients.

The method to count the number of these four decoding complexity coefficients was reported in [6] and briefly summarized below. A simple cache model is used to count the number of cache misses. This cache is assumed to have T entries and each entry has L bytes. Since the MCP operation can be seen as a 2-D memory access copying the required data from the reference frame to the decoded frame, the cache examines the address and length of memory access and the number of cache misses is added by one if the required data for MCP cannot be found in the cache. The address of memory access is determined by MV and the reference frame index while the length of memory access is decided by the inter prediction mode and the resolution of MV (i.e. integer- or sub-pel). The number of

MVs is decided by the inter prediction model while the number of the x- or y-direction interpolation filters are determined by the inter prediction mode and the MV resolution.

B. Improved MCP decoding complexity model

The relationship between decoding complexity and frame size is examined in this sub-section. The decoding platform is assumed to have two-level cache management. Usually, the size of the first-level (L1) cache is small such that only a few data of reference frame can be cached, but the second-level (L2) cache is larger and it might be able to store the whole reference frame. Consider the scenario that the frame size is small such that the reference frame can be stored in the L2 cache. Under such a scenario, the cache miss penalty, which can be seen as the weight coefficient of cache miss, of the L1 cache during the MCP operation is equal to the access time of the L2 cache. In contrast, the cache miss penalty of the L1 cache is equal to that of the L2 cache if frame size is larger such that the required data for MCP operation cannot be obtained from the L2 cache. Therefore, the cache miss penalty of the L1 cache highly depends on the hit rate of L2 cache [7] and the hit rate of the L2 cache is related to frame size. In other words, frame sizes play an important role in the cache miss penalty of the L1 cache and therefore the decoding complexity.

Two experiments are conducted on Pentium 1.7 GHz mobile CPU platform to verify this conjecture. First, we selected Foreman and Mobile CIF (352x288) sequences as training sequences, and pre-encoded 70 bit streams to train the weight coefficients. Then, the weight coefficients are used to estimate the decoding complexities of Mobile D1 (720x480) 1024 Kbps and Stefan CIF 256 Kbps bit streams. The actual MCP decoding complexities for Mobile D1 1024 Kbps and Stefan CIF 256 Kbps bit streams are 247.10 ms and 159.35 ms while the estimated MCP decoding complexities are 205.44 ms and 158.79 ms, respectively. The weight coefficients trained by CIF bit streams can provide a good estimation result for CIF bit streams but cannot for D1 bit streams. Second, we re-train the weight coefficients from 14 D1 bit streams generated by Football and Susie D1 sequences. It is observed that the weight coefficient of cache miss trained by CIF bit streams is different from that trained by D1 bit streams but the other three weight coefficients are quite similar. This clearly demonstrates that frame sizes play an important role in cache miss penalty of L1 cache and therefore the weight coefficient of cache miss in our model. The weight coefficient of cache miss varies among frame sizes and therefore it must be trained according to different frame sizes of bit streams.

III. APPLICATIONS OF PROPOSED COMPLEXITY MODEL

The application of the proposed complexity model is briefly discussed in this section. Consider the scenario that an H.264 encoder generates a single bit stream for different decoding platforms without the use of any decoding complexity model. In contrast, the encoder may generate several bit streams for different decoding platforms separately according to their computational powers so that the resultant bit stream is easy to be decoded for a particular platform. For the latter case, the decoding complexity models have to be integrated into the H.264 encoder so that the encoder can estimate the possible decoding complexity and then generate decoder-friendly bit streams.

A. RDO process with decoding complexity model

In the conventional H.264 encoder, there are two rate-distortion optimization (RDO) processes. The first RDO process decides the

optimal inter prediction mode among the P8x8, P8x4, P4x8 and P4x4 modes for one 8x8 block while the second RDO process determines the optimal inter prediction mode for one 16x16 MB among the P_Skip, P16x16, P16x8, P8x16 modes and four 8x8 blocks whose optimal modes have been decided by the first RDO process. Both of the two RDO processes consist of three steps as shown in Fig. 2 (a). First, since different inter prediction modes have a different number of MVs, the RDO process performs the motion estimation (ME) to find the best MV for a specific mode. Second, the RDO process performs the encoding (*e.g.* the spatial domain transform, quantization and entropy encoding) and then the decoding processes to get reconstructed video frame so that the distortion and the bit rate can be obtained. Finally, the RDO process finds the best mode that yields the minimal RD cost.

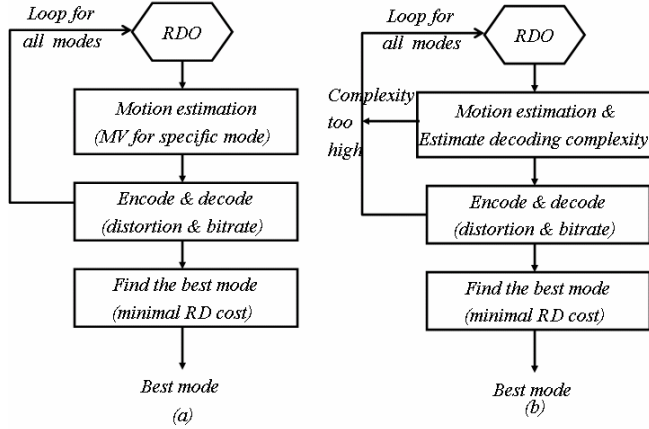


Figure 2. RDO process and RDO process with integrated decoding complexity model.

Since the proposed decoding complexity model requires MV only for a specific inter prediction mode, the estimated complexity can be computed if the weight coefficients are given. The proposed complexity model can be integrated in the ME stage as shown in Fig. 2 (b). For a specific inter prediction mode, the ME process in the conventional H.264 encoder generates integer- and sub-pel MVs for all reference frames and then decides the best MV that minimizes the RD cost. The proposed decoding complexity model estimates decoding complexities from a set of MVs of different pixel resolutions and reference frames, and then eliminates those MVs whose decoding complexities are higher than that can allowed by the target decoding platform. In other words, the RDO process shown in Fig. 2 (b) searches the best MV and mode among a smaller set of MVs and inter prediction modes whose decoding complexities meet the decoding complexity constraint. As a result, the RDO process with the decoding complexity model can select the best MV and inter prediction mode to minimize the RD cost and meet the decoding complexity constraint for the target platform. Since the MV of the P_Skip mode MB is decided by its neighboring MBs, the proposed model can also estimate the required decoding complexity of the P_Skip mode. Another advantage for the RDO process shown in Fig. 2 (b) is that the encoding complexity can be saved since the encoder does not have to perform the encoding and the decoding processes for all inter prediction modes. Please note that the weight coefficients are different among different platforms and frame sizes. Therefore, they have to be measured for the target decoding platform first so that the RDO process with the decoding complexity model can estimate the decoding complexity for a given inter prediction mode and MV.

B. Decoding complexity control

The proposed algorithm to allocate decoding complexities for frame, MB and block is presented in this sub-section. In order to get smooth playback quality, each frame has to be decoded within the same time duration. The time duration to complete frame decoding can be referred to as the frame decoding complexity over the computational power of target decoding platform. Therefore, it is desirable that the same decoding complexity is assigned to each frame. In our algorithm, the target decoding complexity for the k -th MB is given by:

$$C_{MB,k} = C_{MB,average} \cdot k - \sum_{i=1}^{k-1} C_{MB,i}, \quad (2)$$

where $C_{MB,average}$ is the average decoding complexity per MB, which is equal to the frame decoding complexity over the number of MBs.

Since the actual MB decoding complexity cannot exceed the target MB decoding complexity given in (2), the remaining decoding complexities from previous MBs can be used in current MB and the sum of all MBs' decoding complexities is the frame decoding complexity. Here, the same decoding complexity is assigned to all blocks within the same MB. For example, the target decoding complexities for two 16x8 blocks are half of the target MB decoding complexity. As a result, once target decoding complexity of MB and that of its all blocks are determined, the 8x8 block and 16x16 MB RDO processes with integrated decoding complexity model can estimate required decoding complexity for a given MV and inter prediction mode. Then, those MVs whose decoding complexities are higher than the target MB (or block) decoding complexities are eliminated by the 16x16 MB (or 8x8 block) RDO processes. Finally, the two RDO processes decide the optimal MV and inter prediction mode which can minimize RD cost function and also satisfy the given decoding complexity constraint. Please note that the MV and inter prediction mode with the least decoding complexity are selected if all MVs and modes have MB (or block) decoding complexities higher than the constraints. The experimental results for the H.264 encoder with integrated decoding complexity model and complexity control scheme will be shown in the next section.

IV. EXPERIMENTAL MODEL VERIFICATION

We conducted experiments to verify the proposed decoding complexity model given in (1) and the decoding complexity control scheme given by (2) on the PC platform. The CPU was Pentium mobile 1.7 GHz CPU with 512 Mb RAM and the operating system was Windows XP. The reference JM9.4 decoder was optimized by the Intel MMX technology. We selected Foreman and Mobile CIF sequences as training sequences and pre-encoded 70 training bit streams. Each bit stream file contained 270 frames. Among them, 42 bit streams were coded with the intpel ME mode while 28 bit streams were coded with the subpel ME mode. Football D1, Susie D1, Blue HD (1920x1080) and Toy HD sequences were also selected as training sequences. There were 28 D1 and 28 HD training bit streams coded with integer-pel ME mode. Each D1 and HD bit streams contain 68 frames and 14 frames, respectively. The Intel Vtune performance analyzer 8.0 was used to measure the MCP decoding complexities for all pre-encoded bit streams.

Since memory usages for the MCP operation may be different among inter prediction modes, it is desirable that different cache models are used to compute the number of cache misses. In our experiments, there were three different cache models for all inter

prediction modes. The three cache models were used to count the number of cache misses for bit streams with different frames sizes. For the P_Skip mode, a larger cache that has 64 cache entries of size 32 bytes was used to collect to the number of cache misses between two successive MBs. Since no entropy decoding and no inverse spatial transform are required by the P_Skip mode, it is very likely that data of the current MB can be cached into the CPU cache or internal registers and then the cached data can be used in the next MB. For P4x8 and P4x4 blocks, the number of cache entries was 64 and the size of cache entry was 8 bytes. The number of cache misses for an MxN block rather than P4x8 and P8x4 was simply M or M+5 if the y-direction MVs are of subpel accuracy. This setting can result in good estimation results for pre-encoded bit streams as well as others. The number of clock-ticks spent by the MCP operation was measured by the Intel Vtune and then the number of clock-ticks was divided by $1.7 \cdot 10^7$ to get the decoding time of MCP in milli-seconds. The proposed complexity model counted the number of these four decoding complexity coefficients for those pre-encoded bit streams. Finally, the information was used to train the weight coefficients, i.e. α , β , γ , and μ . The constrained least square method was used to determine weight coefficients. The weight coefficients for CIF sequences are:

$$\alpha = 2.412 \times 10^{-5}, \beta = 4.861 \times 10^{-6},$$

$$\gamma = 2.675 \times 10^{-7}, \mu = 9.656 \times 10^{-5}.$$

Then, 28 D1 and 28 HD bit streams coded with integer-pel ME mode were used to train the weight coefficient of cache miss for D1 and HD sequences. The weight coefficients of cache miss for D1 and HD sequences are $\alpha = 3.9556 \cdot 10^{-5}$ and $\alpha = 4.293 \cdot 10^{-5}$, respectively.

Next, these weight coefficients were integrated into our complexity model to estimate the decoding complexities of the following test bit streams: three CIF video sequences, three D1 video sequences and two HD sequences. Tables I shows the comparison between the estimated decoding complexity based on the proposed complexity model and the actual decoding complexity measured by the Intel Vtune. We see that the proposed complexity model provides good estimation results for these test bit streams. The errors are within 10%. Tables II shows the experimental results that the H.264 encoder with a decoding complexity model and a complexity control scheme generates bit streams that meet different decoding complexity constraints.

As compared with sequences without decoding complexity control, the CIF Stefan sequence with the target MCP complexity at 110 ms loses 0.67 dB in PSNR but saves 28.24% in decoding complexity. The D1 Ship sequence with the target MCP complexity at 60 ms loses 0.39 dB in PSNR but saves 35.38% in decoding complexity. Finally, the HD Toy sequence with the target MCP complexity at 160 ms loses 0.32 dB in PSNR but saves 22.26% in decoding complexity. These experimental results clearly demonstrate that the H.264 encoder with the proposed decoding complexity model and the decoding complexity control scheme can generate bit streams to meet different decoding complexity constraints. The errors between actual decoding complexities and target decoding complexity constraints are all less than 10%. Thus, the H.264 encoder with the decoding complexity control scheme can generate bit streams to meet target decoding complexity constraints well. Besides, the resultant bit streams can save a significant amount of decoding complexity at the cost of some PSNR loss. This is particular

interesting in a mobile broadcasting environment, where multiple mobile devices will get broadcast/streaming video in real time.

TABLE I. DECODING COMPEXITY (DECODING TIME IN MILLI-SECONDS) COMPARSION FOR CIF, D1 AND HD VIDEO SEQUENCES

Sequence	Actual complexity (ms)	Estimated complexity (ms)	Error (%)	PSNR (dB)
Flower 384k (CIF)	127.52	125.12	1.88	32.07
Akiyo 320k (CIF)	73.78	76.22	3.31	46.66
Stefan 256k (CIF)	159.35	158.80	0.35	32.08
Mobile 1280k (D1)	263.04	249.27	5.23	30.25
Ship 1024k (D1)	98.13	96.83	1.33	38.62
Susie 768k (D1)	205.03	205.43	0.20	43.12
Toy 5.12M (HD)	215.75	211.10	2.16	38.70
Flower 2.56M (HD)	403.04	412.42	2.33	41.72

TABLE II. DECODING COMPEXITY CONTROL FOR VARIANT SEQUENCES

Sequence	Actual complexity (ms)	Error (%)	Complexity saving (%)	PSNR loss (dB)
Stefan CIF 256k (130 ms)	134.41	3.33	15.65	0.43
Stefan CIF 256k (110 ms)	114.36	3.88	28.24	0.67
Ship D1 1024k (60 ms)	80.87	1.20	17.49	0.17
Ship D1 1024k (60 ms)	63.41	5.38	35.38	0.39
Toy HD 5.12M 180 (ms)	185.27	2.62	14.13	0.17
Toy HD 5.12M 160 (ms)	167.72	4.22	22.26	0.32

V. CONCLUSION

An improved motion compensation decoding complexity model and its application to H.264/AVC encoding were presented in this work. This model helps the encoder select proper motion vector and inter prediction modes, and then generates a video bit stream that is most suitable for a receiver platform with some hardware constraint. As a result, the coded bit stream can balance the tradeoff between the RD requirement as well as the computational power of the decoding platform. The proposed decoding complexity model was verified experimentally. We showed that the model provides fairly good estimation results for various test bit streams. The performance of the H.264 codec with the proposed decoding complexity model and the proposed decoding complexity control scheme was presented. It was shown that the H.264 encoder can generate bit streams according to different decoding complexity constraints accurately. The resultant bit streams can be decoded at a much lower complexity at the cost of small PSNR loss.

REFERENCES

- [1] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC coding standard," IEEE Trans. Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 560-576, July 2003.
- [2] "Information technology – coding of audiovisual objects – Part 2: Visual," Dec. 1999.
- [3] J. Valentim, P. Nunes and F. Pereira, "Evaluating MPEG-4 video decoding complexity for an alternative video complexity verifier model," IEEE Trans. on Circuits and Systems for Video Technology, pp. 1034-1044, vol. 12, no. 11, Dec. 2002.
- [4] M. van der Schaar and Y. Andreopoulos, "Rate-distortion-complexity modeling for network and receiver aware adaptation," IEEE Trans. on Multimedia, pp. 471-479, vol. 7, no. 3, June 2005.
- [5] M. Horowitz, A. Joch, F. Kossentini and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," IEEE Trans. Circuits and Systems for Video Technology, vol. 13, no. 7, pp.704-716, July 2003.
- [6] S. W. Lee and C.-C. Jay Kuo, "Complexity modeling for motion compensation in H.264/AVC decoder," accepted by IEEE Int. Conf. on Image Processing 2007.
- [7] J. Hennessy and D. A. Patterson, "Computer architecture: A quantitative approach 2nd edition," Chapter 5, pp. 417, 1996.