MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Complexity and Memory Efficient GOP Structures Supporting VCR Functionalities in H.264/AVC

Jian Lou, Shan Liu, Anthony Vetro, and Ming-Ting Sun

TR2008-041 May 2008

Abstract

Supporting digital Video Cassette Recording (VCR) trick-play functionalities (e.g. random access, fast-forward play, fast-reverse play) is desirable for compressed video streams. However, due to strong inter-frame dependencies introduced by motion compensated prediction (MCP), the computational complexity and memory requirement is drastically increased. Trade offs between coding efficiency and decoding complexity can be made with different Group of Pictures (GOP) structures. In this paper, we investigate two flexible GOP structures, named G-Group and Binary Reference GOP Structure (BRGS), which can achieve trick-play functionalities while keeping low decoder complexity and memory requirement for the state-of-the art H.264/AVC video coding standard. The schemes are drift-free since they utilize the compression and memory management tools adopted in H.264/AVC. Our analysis and experimental results show that they can greatly reduce the decoder complexity and buffer size while introducing only about 4.0%-7.6% bit rate increase on average. The computational complexity saving for the worst case and average case can be up to 77.8% and 62.5%, respectively. The memory buffer for the fast-reverse play mode can be reduced to 33.3% compared to the conventional scheme. moreover, the schemes are flexible and can be easily adapted to achieve a good trade off between compression performance and complexity saving for the trick-play modes.

IEEE International Symposium on Circuits and Systems (ISCAS)

Complexity and Memory Efficient GOP Structures Supporting VCR Functionalities in H.264/AVC

Jian Lou*[a], Shan Liu[b], Anthony Vetro[b] and Ming-Ting Sun[a]
[a]Department of Electrical Engineering, University of Washington, Seattle, WA 98195, USA
{louj, sun}@ee.washington.edu
[b]Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, USA
{liu, avetro}@merl.com

***Abstract*—Supporting digital Video Cassette Recording (VCR) trick-play functionalities (e.g. random access, fast-forward play, fast-reverse play) is desirable for compressed video streams. However, due to strong inter-frame dependencies introduced by motion compensated prediction (MCP), the computational complexity and memory requirement is drastically increased. Tradeoffs between coding efficiency and decoding complexity can be made with different Group of Pictures (GOP) structures. In this paper, we investigate two flexible GOP structures, named G-Group and Binary Reference GOP Structure (BRGS), which can achieve trick-play functionalities while keeping low decoder complexity and memory requirement for the state-of-the-art H.264/AVC video coding standard. The schemes are drift-free since they utilize the compression and memory management tools adopted in H.264/AVC. Our analysis and experimental results show that they can greatly reduce the decoder complexity and buffer size while introducing only about 4.0%-7.6% bitrate increase on average. The computational complexity saving for the worst case and average case can be up to 77.8% and 62.5%, respectively. The memory buffer for the fast-reverse play mode can be reduced to 33.3% compared to the conventional scheme. Moreover, the schemes are flexible and can be easily adapted to achieve a good tradeoff between compression performance and complexity saving for the trick-play modes.**

I. INTRODUCTION

Digital video compression techniques have played a very important role in the world of multimedia systems where bandwidth and storage are valuable commodities. Nowadays, most of the video contents for consumer applications are encoded using various video coding standards, since compressed digital videos are more preferable than the traditional video cassette in terms of storage and transmission efficiency. Due to the quick and user-friendly browsing of video content, it is highly desirable for multimedia systems to support Video Cassette Recording (VCR) functionalities, such as random access, fast-forward, fast-reverse play, etc. While it is straightforward to implement the trick-play modes for the video cassettes, it is not a trivial task for compressed video streams. Most of the video coding standards are based on the hybrid motion compensated prediction (MCP) framework, where correlation between successive video frames is utilized to achieve higher compression ratios. However, MCP introduces inter-frame dependencies that are unfriendly for VCR functionalities. That is, in order to display one video frame in a trick-play mode, not only does the target frame need to be decoded, but also any reference frames that the target frame is predicted from need to be decoded as well. Therefore, the computational and memory complexity is dramatically increased compared to the normal decoding process.

Several previous schemes on MPEG-2 or MPEG-4 have been proposed addressing the implementation of video trick-play modes. Several approaches utilizing transcoding between different frame types have been proposed [1]-[3]. However, extra complexity and higher storage cost are required to perform the transcoding. The approach proposed in [3] also causes drift due to the motion vector approximation. In [4], a scheme that stores both the forward-encoded and backward-encoded bitstreams in the server is proposed to reduce the reverse-play complexity while maintaining a low bandwidth. However, this doubles the storage requirement of the server and cannot be used for applications that require real-time or low-delay encoding. Center-biased motion vector distribution characteristics of video sequences are utilized in [5] for MPEG-2 video reverse play. However, the bandwidth savings are highly dependent on the statistics of the coded sequence; thus, it is not always efficient for high motion sequences. Furthermore, the scheme cannot be applied in the context of H.264/AVC due to the different semantics of the standards.

H.264/AVC offers 2-3 times improvement in compression efficiency over MPEG-2, and is becoming the dominant compressed video format for a wide range of applications [6]. In H.264/AVC, multiple reference pictures could be organized in List 0 and List 1 and used for prediction of P and B pictures. Another important feature of H.264/AVC is the decoupling of referencing order from display order, which allows the encoder to choose the ordering of pictures for referencing and display purposes with a high degree of flexibility [7]. With reference picture list reordering commands and Memory Management Control Operations (MMCO), encoders can flexibly choose any short-term and long-term reference picture as the reference picture to be used for prediction.

In this paper, we analyze two complexity and memory efficient Group of Pictures (GOP) structures in the context of

This work was performed during Jian Lou's internship at Mitsubishi Electric Research Laboratories.

the H.264/AVC standard to support VCR functionalities. This paper is organized as follows. Firstly, the analysis on the impact of VCR functionalities on H.264/AVC decoder complexity and memory requirement is explained. Secondly the two GOP structures are presented and compared. Then, the performances of the schemes are given. Finally, some concluding remarks and future work are provided.

II. Complexity and Memory Analysis of VCR Functionalities in Decoder

In this section, we provide an analysis of the computational complexity for H.264/AVC decoders when applying VCR functionalities. Here, we denote the index of the first I-frame as 0, and define N as the number of frames in one GOP and M as the inter frame distance between every two successive I/P-frames, which we assume fixed throughout the whole GOP. Also, we will use R_P, R_{BL0}, and R_{BL1} as the number of reference frames used for P-frames, B-frame List 0, and B-frame List 1, respectively. The bitstreams are encoded with a conventional prediction structure where every P-frame is predicted from the nearest forward I/P-frame and every B-frame uses the nearest I/P-frame as the references. Forward reference frames are in List 0 and backward reference frames are in List 1.

A. Random Access

When random access to a specific frame is required, the frame to be displayed and any frames that it depends on need to be decoded. The complexity to access the *j-th* frame in one GOP[1] is denoted as $C_{RA}(j)$ which indicates the number of frames to be decoded.

$$C_{RA}(j) = \begin{cases} 1 & \text{I - frame} \\ j/M + 1 & \text{P - frame} \\ \min(\lceil j/M \rceil + R_{BL1}), N/M + 1) & \text{B - frame} \end{cases} \quad (1)$$

Within one GOP, the decoding complexity for a P-frame is determined by its index j, since large j means more previously encoded P-frames need to be decoded due to MCP reference dependencies.

B. Fast-Forward Play

In fast-forward play, we can jump to the next I-frame as the starting point. Denote the speed-up factor as s. After s/g GOPs, the frame to be displayed will again be an I-frame, where $g = \gcd(s, N)$ stands for the greatest common divisor of s and N. Therefore, our analysis is based on $N*s/g$ frames starting from an I-frame. Moreover, we assume N is larger than s, which is usually the case in practice.

In general, fast-forward play can be regarded as accessing the frames with indices of 0, s, $2s$, …$(N/g-1)*s$. In these frames, only the first frame is an I-frame and there are $N*s/(g*h)$-1 P-frames and the remaining are B-frames, where $h = \mathrm{lcm}(s, M)$ stands for the least common multiple of s and M. If all the decoded frames for displaying the current frame are discarded although some of the decoded frames could be reused for decoding future frames to be displayed, the average number of frames that need to be decoded for fast-forward play is

$$\overline{C_{FF}} = \frac{1 + \sum_i (i/\mathrm{M} + 1) + \sum_j \min(\lceil j/M \rceil + R_{BL1}), N/M + 1)}{N/g} \quad (2)$$

The three parts in the numerator indicate the total decoding complexity for I-frame, P-frames, and B-frames, respectively.

If we store those decoded frames which could be used for future display, the decoding complexity and bandwidth requirement will be reduced, but the buffer memory will be increased. The decoding complexity defined in (2) will be reduced by the number of necessary frames buffered in the memory, and the memory size will be increased by the maximum number of frames buffered for future use in addition to $R_{BL0}+R_{BL1}$ in the unit of frame size.

C. Fast-Reverse Play

In the fast-reverse play mode, the problem is similar to that for fast-forward play assuming the ending point is an I-frame. If the decoder performs the same as for fast-forward play when dealing with the decoded frames needed for the future use, i.e. discarding them or not, the average complexity for fast-reverse play is exactly the same as that for fast-forward. The reason is that both fast-forward and fast-reverse play modes follow the same periodicity which indicates that all the frames to be displayed are the same for the two modes.

If we choose to buffer those decoded frames needed for future use, then the maximum decoding speed and maximum buffer size will be greater for fast-reverse play compared to fast-forward play due to the forward encoding nature in video compression.

III. Complexity and Memory Efficient GOP Structures

A straightforward scheme to reduce the computational complexity and memory requirement is to assign the I-frame as the only reference for all the P-frames in the same GOP and every B-frame uses the nearest 2 I/P-frames as the references. We refer to this scheme as "All P Ref I". The dependencies between P-frames have been eliminated in this scheme, so the complexity for accessing one P or B-frame is minimal. The obvious drawback of this method is that the performance of MCP will be greatly degraded when the distance between I/P-frames becomes too long.

A. G-Group GOP Structure

In order to reduce the inter-frame dependencies without sacrificing too much coding efficiency, we introduce a scheme called G-Group GOP Structure where several successive P-frames are grouped together which only use the last P-frame in the previous group or the I-frame as the reference. Here, G stands for the number of P-frames in one group.

The index of the reference frame, RI_P, for the current P-frame with index j is

$$RI_P = \lfloor (j/M - 1)/G \rfloor MG \quad (3)$$

In addition, every B-frame still uses the nearest 2 I/P-frames as the references in order to keep the low inter-frame dependencies. Fig. 1 provides an example with G=4.

[1] If j is larger than N, we can use $\mathrm{mod}(j, N)$ instead.

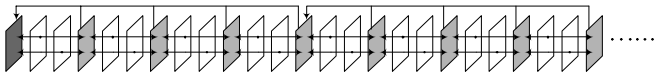

Figure 1. G-Group GOP Structure (M=3, G=4)

The Longest Forward Prediction Distance (*LFPD*), the Average Forward Prediction Distance (*AFPD*), the Random Access Worst Complexity (*RAWC*) and the Random Access Average Complexity (*RAAC*) are

$$LFPD_{G-Group} = MG \tag{4}$$

$$AFPD_{G-Group} = (M+G)/2 \tag{5}$$

$$RAWC_{G-Group} = \lceil N/G \rceil + 1 \tag{6}$$

$$RAAC_{G-Group} = (1 + G\sum_{i=1}^{\frac{N-1}{MG}}(i+1) + (M-1)\sum_{j=1}^{\frac{N-1}{MG}}((j+3)G-1))/N$$
$$= (7MNG + N^2 - 2MN - 4NG - 5MG + 2M + 4G - 1)/2MNG \tag{7}$$

LFPD and *AFPD* are indicators of coding efficiency, while *RAWC* and *RAAC* can be derived from trick-play complexity constraints. According to (4)-(7), when N and M are fixed, G can be served as the parameter to control the coding efficiency and complexity. Generally, the larger the G, the worse the coding efficiency, but the better the trick-play performance.

B. Binary Reference GOP Structure

Notice that the G-Group scheme linearly increases prediction distance and reduces the complexity. It is possible to use a logarithmic scheme which we refer to as Binary Reference GOP Structure (BRGS). The scheme uses an index derived from the binary code of the number, j/M, where j is the frame index of a P-frame. Let L represent the number of bits for the binary code. The index of the reference frame for the current P-frame with index j is

$$RI_P = \mathrm{mod}(j/M, 2^L) - 2^k + \lfloor (j/M - 1)/2^L \rfloor 2^L \tag{8}$$

where k stands for the position of last 1 in the binary code of $\mathrm{mod}(j/M, 2^L)$. Fig. 2 provides examples of BRGS with L=3.

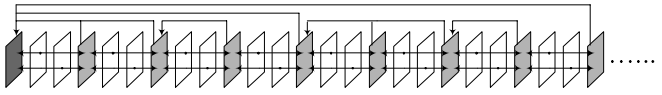

Figure 2. Binary Reference GOP Structure (M=3, L=3)

It can be shown that:

$$LFPD_{BRGS} = M2^L \tag{9}$$

$$AFPD_{BRGS} = (M + L + 1)/2 \tag{10}$$

$$RAWC_{BRGS} = (N-1)/M2^L + L \tag{11}$$

$$RAAC_{BRGS} = (1 + \sum_{i=0}^{\frac{N-1}{M2^L}-1}(2^L + L2^{L-1} + 1 + (M-1)(3\cdot 2^L + L2^{L-1}) + iM2^L)))/N$$
$$= ((5MN + MNL - 4N - 3M - ML + 4)2^L + N^2 - 1)/2MN2^L \tag{12}$$

When L is larger than 3, which corresponds to G larger than 8, BRGS can achieve lower complexity than the G-Group scheme with same LFPDs. L can be utilized to control the coding efficiency and complexity.

The average number of frames that need to be decoded for fast-forward/reverse play can be derived by replacing the three components in (2) with the counterparts in the two schemes. The maximum buffer size required for fast-reverse play will be reduced, due to the fewer frames buffered to be used for future display.

IV. Performance Evaluation

In our experiments, we set N to 30 frames and M to 3. The number of reference frames in every list is set to 1. The necessary frames are stored in the memory for future display.

TABLE I. Average No. of Frames to be Decoded for Random Access

	Conv.	All P Ref I	G-Group (G=2)	G-Group (G=4)	BRGS (L=3)
LFPD	3	27	6	12	24
AFPD	1.97	5.69	2.38	3.21	3.21
RAWC	12	4	8	6	6
RAAC	6.83	3.23	4.83	3.83	3.83

Table I compares the average number of frames to be decoded for random access which indicates that both G-Group and BRGS can achieve similar complexity as the "All P Ref I" scheme. Fig. 3 shows the decoding speed needed for the fast-forward and fast-reverse play modes in the worst case and Fig. 4 shows the decoding speed on average. These two figures indicate that the "Conventional" structure requires the decoder to be 12 times faster while the "All P Ref I" structure only needs to be 4 times faster. The decoding speeds needed for the G-Group and BRGS are between those of the two reference structures. The complexity saving for the worst case and the average case can be up to 77.8% and 62.5%, respectively.

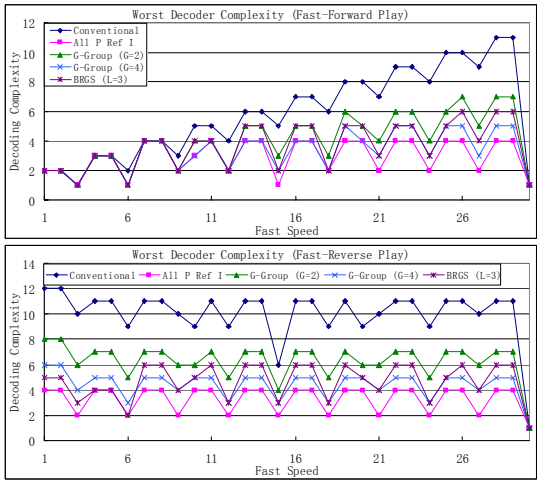

Figure 3. Worst Decoder Complexity

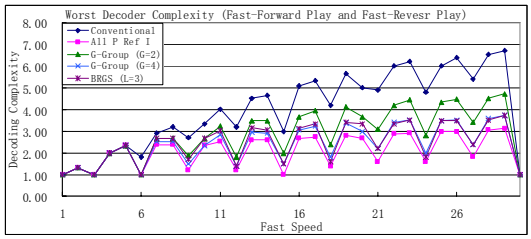

Figure 4. Average Decoder Complexity

The maximum buffer sizes needed for the different schemes are presented in Fig 5. For fast-forward, there is no difference between difference schemes, however the memory buffer for the fast-reverse play mode which is of more importance [1]-[5] can be reduced to 33.3%.

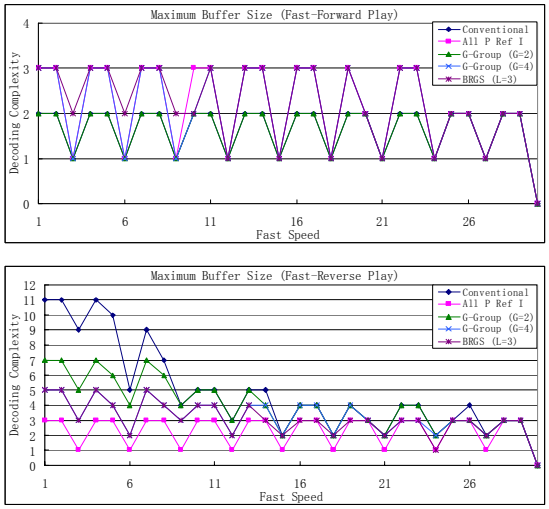

Figure 5. Max Buffer Size

The coding efficiency of different schemes is verified using the H.264/AVC reference software, JM12.3 [8]. We choose four 1280*720 format sequences, "City", "Cyclists", "Horses", and "Night". High complexity RD optimization is enabled. Table II lists the PSNR loss or equivalent bitrate increase compared with the "Conventional" approach when applying different GOP structures by using the method proposed in [9]. Even though "All P ref I" structure gives the best trick-play performance, it introduces 17.27% bitrate increase. The average bitrate increases from G-Group and BRGS schemes are between 3.97% and 7.6%. As can be seen from the sample R-D curves for "Cyclists" in Fig. 6, the G-Group and BRGS are much better than the "All P ref I" structure. Also, when video is encoded at higher quality, e.g. 38dB, the performance loss from the proposed schemes becomes negligible.

TABLE II. COMPRESSION PERFORMANCE FOR DIFFERENT GOP STRUCTURES COMPARED WITH THE "BENCHMARK" SCHEME

Seq.		All P Ref I	G-Group (G=2)	G-Group (G=4)	BRGS (L=3)
City	ΔPSNR	-0.68dB	-0.15dB	-0.27dB	-0.31dB
	ΔBitrate	23.21%	5.09%	9.21%	10.43%
Cycl.	ΔPSNR	-0.46dB	-0.09dB	-0.19dB	-0.19dB
	ΔBitrate	17.43%	3.44%	7.22%	7.37%
Horses	ΔPSNR	-0.41dB	-0.1dB	-0.2dB	-0.2dB
	ΔBitrate	16.24%	3.87%	8.01%	8.07%
Night	ΔPSNR	-0.39dB	-0.11dB	-0.14dB	-0.15dB
	ΔBitrate	12.18%	3.48%	4.17%	4.55%
Ave.	ΔPSNR	-0.49dB	-0.11dB	-0.19dB	-0.21dB
	ΔBitrate	17.27%	3.97%	7.15%	7.6%

These experimental results indicate that good tradeoff between compression performance, complexity, and memory saving for the trick-play modes can be made with different GOP structures, which favors user preference. With appropriate parameters, G-Group and BRGS can greatly reduce the computational complexity and memory requirement for VCR functionalities in H.264/AVC without much loss in coding efficiency.

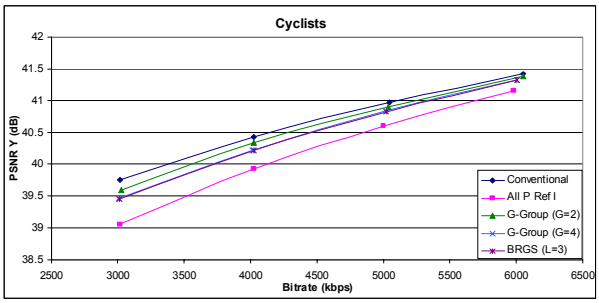

Figure 6. R-D Performance for Different GOP Structures

V. CONCLUSION AND FUTURE WORK

Due to the inter-frame dependencies introduced by modern video coding standards, the computational complexity and memory requirement for decoders are drastically increased when applying VCR functionalities. In this paper, we first analyze the impact of trick-play modes on decoder complexity and buffer size. Then, we analyze two drift-free schemes called G-Group GOP Structure and Binary Reference GOP Structure for H.264/AVC which can reduce the computational complexity and buffer size when applying VCR functionalities relative to conventional GOP structures. Theses schemes favor trick-play mode with only 4.0%-7.6% bitrate increase while requiring much less complexity and smaller buffer size. Users can choose appropriate parameters to control the GOP structure to satisfy their constraints on coding efficiency, decoder complexity and memory requirement. Our future work will focus on jointly optimizing the coding efficiency and trick-play performance.

REFERENCES

[1] M.S. Chen and D.D. Kandlur, "Downloading and Stream Conversion: Supporting Interactive Playout of Videos in a Client Station", IEEE Proc. of International Conference on Multimedia Computing and Systems, pp. 73–80, 1995.

[2] S.J. Wee, "Reversing Motion Vector Fields", IEEE Proc. of International Conference on Image Processing, pp. 209–212, Oct. 1998.

[3] S. J. Wee and B. Vasudev, "Compressed-domain Reverse Play of MPEG Video Streams," SPIE Proc. of Conference on Multimedia Systems and Applications, pp. 237–248, Nov. 1998.

[4] C. W. Lin, J. Zhou, J. Youn, and M. T. Sun, "MPEG Video Streaming with VCR Functionality", IEEE Trans. Circuits and Systems for Video Technology, vol. 11, no. 3, pp. 415–425, Mar. 2001.

[5] C. H. Fu, Y. L. Chan, T. P. Ip and W. C. Siu, "New Architecture for MPEG Video Streaming System With Backward Playback Support", IEEE Trans. Image Proc., vol. 16, no. 8, pp. 2169–2183, Sept. 2007.

[6] ITU-T Recommendation H.264 & ISO/IEC 14496-10, "Advanced Video Coding for Generic Audiovisual Services", Version 4, 2005.

[7] T. Wiegand, G. J. Sullivan, G. Bjøntegaard and A. Luthra, "Overview of the H.264/AVC Video Coding Standard", IEEE Trans. Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 560–576, July 2003.

[8] Joint Video Team Reference Software, Version 12.3 (JM12.3), http://iphome.hhi.de/suehring/tml/download/

[9] G. Bjøntegaard, "Calculation of Average PSNR Differences Between RD-curves", Video Coding Experts Group, VCEG-M33, Mar. 2001.