Macroblock-Based Algorithm for Dual-Bitstream MPEG Video Streaming with VCR Functionalities

Tak-Piu Ip, Yui-Lam Chan, Chang-Hong Fu and Wan-Chi Siu
Centre for Multimedia Signal Processing
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract— **Reverse playback is the most common video cassette recording (VCR) functions in many digital video players. However, the predictive processing techniques employed in MPEG severely complicate the reverse-play operation. One approach to achieve the reverse playback is to store an additional backward-encoded bitstream into the server. Once the client requests a backward-play operation, the server will select an appropriate frame for the client from either the forward or backward-encoded bitstream by considering the cost of network bandwidth and the decoder complexity. Unfortunately, the forward and backward-encoded bistreams are encoded separately. The frame that has previously decoded by the client may not exactly identical to the reference of the current selected frame and the drift problem is occurred. In this paper, we propose a macroblock-based approach to alleviate the drift problem with the minimal requirements on the network bandwidth and the decoder complexity. The novel macroblock-based techniques are used to manipulate the necessary macroblocks in the compressed-domain and the server then sends the processed macroblocks to the client machine. Experimental results show that, as compared to the conventional dual-bitstream system, the new streaming system enhances the quality of the reconstructed frame significantly.**

I. Introduction

With the rapid growth of online multimedia contents, it is also highly desirable that video streaming systems should have the capability of providing fast and effective browsing. A key technique that facilitates fast and user-friendly browsing of video content is to provide full video cassette recording (VCR) functionality. The set of effective VCR functionality consists of forward, backward, stop, pause, fast forward, fast backward, and random access.

Recently, some works on the implementation of backward playback for MPEG compressed video in streaming applications have been introduced [3]–[6]. Chen and Kandlur [3] suggested an approach of converting an incoming MPEG bitstream with I-B-P structure into a local bitstream with I-B structure by performing a P-to-I frame conversion at the client machine. This P-to-I frame conversion was used to break the inter-frame dependencies between the P-frames and the I-frames. After the frame conversion and frame re-ordering, the motion-vector reversing approach developed in [4] could be used for backward playback of the new I-B stream. However, this approach requires extra decoder complexity to perform the P-to-I conversion and higher storage cost to store the local bitstream in the client. Wee and Vasudev [5] described a reverse-play transcoder which is used to convert the I-P frames into another I-P bitstream with a reversed frame order. Lin et al [6] recently proposed to store the forward-encoded bitstream and the backward-encoded bitstream in the server. The idea behind is to switch frames between the forward-encoded bitstream and the backward-encoded bitstream based on a frame-selection scheme which is used to minimize the transmitted frames over the network for any speed-up factors. This dual-bitstream technique can alleviate the decoder complexity while maintaining the low network bandwidth requirement in the VCR operations. However, the frame-selection scheme employed by dual-bitstream technique does not work well when the speed-up factor is smaller than N/4, where N is the group-of-picture size [6], and the bitstream switching is occurred. In this case, it would cause the drift problem because the P-frames in the backward-encoded bitstream would be approximated by the P-frame in the forward-encoded bitstream and vice versa. Although the drift problem could be compensated by additional drift-compensated bitstreams, it is too complicated.

In this paper, we provide a computationally efficient solution to perform bitstream switching in the dual-bitstream MPEG streaming video system to reduce the drift problem arising from mismatch between the forward-encoded and backward-encoded bitstreams. Since the proposed scheme mainly operates on the compressed-domain, complete decoding and encoding are not required in the server. Thus, an additional processing requirement in the server can be minimized.

The organization of this paper is as follows. Section II of this paper presents an macroblock-based algorithm for performing bitstream switching in the dual-bitstream MPEG video streaming . Simulation results are presented in Section III. Finally, some concluding remarks are provided in Section IV.

II. Macroblock-based technique for bitstream switching in the dual-bitstream MPEG video streaming system

For dual-bitstream system, if the cost of current displayed frame (d_C) has the smallest value among all distances, then the bitstream switching is required and the drift problem are occurred. This is the case when the current VCR mode is forward play and then backward is requested. In this paper, we consider a macroblock-based solution to reduce the drift for a P-to-P approximation in which the switching between the FB and RB is at predictive frames. In the following,

0-7803-8834-8/05/$20.00 ©2005 IEEE.

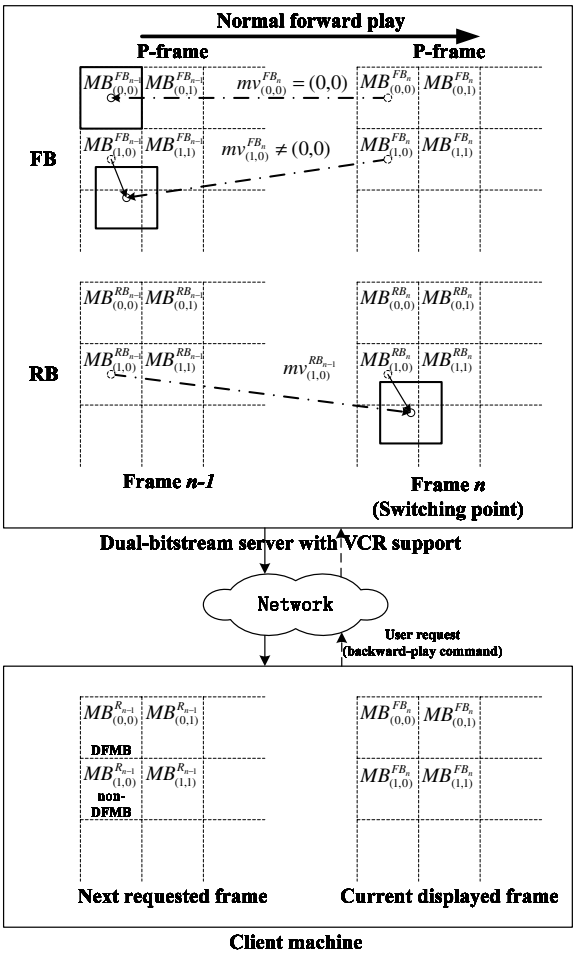

Fig. 1. Definition of the drift-free macroblock (DFMB) and non-drift-free macroblock (non-DFMB).

we provide a detailed description and formulation of the proposed macroblock-based algorithm. Since the process of the FB to RB switching is quite similar to that of the RB to FB switching, for the sake of simplicity, we only focus the discussion on the case that the FB is switched to the RB. To illustrate the proposed algorithm, we use the example where the current VCR mode is normal forward play and then backward-play request is launched. The situation in macroblock level is depicted in Fig. 1. In the server, $MB_{(k,l)}^{FB^n}$ and $MB_{(k,l)}^{RB^n}$ represent the macroblocks at the k^{th} row and l^{th} column of frame n in the FB and the RB respectively. In this example, for simplicity of the presentation, the MPEG video is coded in I- and P-frames only. The extension of our discussion to the case with the general I-B-P structure is straightforward.

A. Macroblock viewpoint of the dual-bitstream algorithm

In MPEG video coding standards, block motion-compensated prediction (MCP) is used to reduce the temporal redundancies in video sources [1], [2]. The prediction process is what gives the MPEG codecs the advantage over pure still-frame coding methods. In motion-compensated prediction, the previously transmitted and decoded frame serves as the prediction for the current frame. The difference between the prediction and the actual current frame is the prediction error. The prediction errors in the FB, $e_{(k,l)}^{FB_n}$, and the RB, $e_{(k,l)}^{RB_{n-1}}$, are given by

$$e_{(k,l)}^{FB_n} = MB_{(k,l)}^{FB_n} - MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n}) \quad (1)$$

$$e_{(k,l)}^{RB_{n-1}} = MB_{(k,l)}^{RB_{n-1}} - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}}) \quad (2)$$

where $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$ stands for the motion-compensated macroblock of $MB_{(k,l)}^{FB_n}$ which is translated by the motion vector $mv_{(k,l)}^{FB_n}$ in the previously reconstructed frame $n-1$ of the FB and $MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})$ represents the motion-compensated macroblock of $MB_{(k,l)}^{RB_{n-1}}$ which is translated by the motion vector $mv_{(k,l)}^{RB_{n-1}}$ in the previously reconstructed frame n of the RB. Noted that, in contrast to the FB, frame $n-1$ is predicted from frame n in the RB since this bitstream is generated by encoding the video frames in the reverse order. All these prediction errors are variable-length encoded and stored in the server.

Fig. 1 also shows the client side where a user requests a backward-play command at frame n, the next displayed frame is frame $n-1$. At that moment, frame n is stored in frame buffer at the client machine as it is used for decoding subsequent frame $n+1$ in the forward play operation. In other words, all $MB_{(k,l)}^{FB_n}$ are available at the decoder. When macroblocks of the requested frame (frame $n-1$) is requested, the distance d_C in the frame-selection scheme has the smallest value among all distances. Thus the current displayed frame of the FB (frame n) is selected as the reference to predict the requested frame (frame $n-1$) and the coded prediction error of frame $n-1$ in the RB is transmitted to the client machine. The client machine of the dual bit-stream system decodes the coded prediction error by using the value of the quantized DCT coefficients. These quantized coefficients are de-quantized and put through an inverse DCT. This process yields the residual signal $e_{(k,l)}^{RB_{n-1}}$ of frame $n-1$ in the RB. The requested macroblock of frame $n-1$, $MB_{(k,l)}^{R_{n-1}}$, can be reconstructed by adding the prediction which results from the previously decoded frame by applying motion compensation, as indicated below,

$$MB_{(k,l)}^{R_{n-1}} = MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) + e_{(k,l)}^{RB_{n-1}} \quad (3)$$

where $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}})$ is the motion-compensated macroblock of $MB_{(k,l)}^{R_{n-1}}$ which is translated by the motion vector $mv_{(k,l)}^{RB_{n-1}}$ in the previously reconstructed frame of the decoder. Note that, for $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}})$, the reference frame is come from the FB whereas the motion vector $mv_{(k,l)}^{RB_{n-1}}$ is extracted from the RB. Substitution of (2) into (3) yields

$$MB_{(k,l)}^{R_{n-1}} = MB_{(k,l)}^{RB_{n-1}} + [MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})] \quad (4)$$

This equation implies that the reconstructed macroblock in the client machine $MB_{(k,l)}^{R_{n-1}}$ is deviated from the $MB_{(k,l)}^{RB_{n-1}}$ of the RB in the server by $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})$. This term indicates the mismatch due to the use of the current displayed frame of the FB (frame n) as an approximation of frame n of the RB to predict the requested macroblocks in frame $n-1$ of the RB. It would lead to the drift errors. Furthermore, such errors could propagate and be accumulated in the subsequent P-frames.

B. Classification of Macroblocks at the switching point

In order to reduce such mismatch, in contrast to the frame-based scheme used in the conventional dual-bitstream architecture, a macroblock-based algorithm is proposed to use at the switching point. During the bitstream switching from the FB to the RB, motion vectors of the current displayed frame are extracted from the FB and these motion vectors are used by a macroblock classifier to identify the types of macroblocks. Two types of macroblocks are now defined. For illustration, we use the example in Fig. 1 again to give a clearer account for defining macroblocks in the requested frame $n-1$, $MB_{(k,l)}^{R_{n-1}}$. In this figure, a user requests a backward-play operation at frame n, the next frame to be displayed is frame $n-1$. $MB_{(k,l)}^{R_{n-1}}$ is defined as a drift-free macroblock (DFMB) if the macroblock in frame n of the FB, $MB_{(k,l)}^{FB_n}$, having the same spatial position of $MB_{(k,l)}^{R_{n-1}}$ is coded without motion compensation (non-MC macroblock). Otherwise, it is defined as a non-drift-free macroblock (non-DFMB). For example, in Fig. 1, since the motion vector of $MB_{(0,0)}^{FB_n}$, $mv_{(0,0)}^{FB_n}$, is zero, it means that $MB_{(0,0)}^{FB_n}$ is a non-MC macroblock and $MB_{(0,0)}^{FB_{n-1}}$ is classified as DFMB. On the other hand, since $MB_{(1,0)}^{FB_n}$ is coded with motion compensation (MC-macroblock), $MB_{(1,0)}^{R_{n-1}}$ is classified as non-DFMB. In this paper, our proposed algorithm works at the level of macroblocks. The server processes those non-DFMBs the same as the conventional dual-bitstream algorithm. The server gets the data from the RB and the drift problem mentioned in (4) cannot be avoided. As it will be described in detail, for DFMBs, the server will use only the MPEG data from the FB to avoid the drift errors. Note that the server will process each DFMB in the compressed-domain such that a complete decoding and encoding are not required at the server.

C. Sign inversion technique for DFMBs

Since frame n of the FB is stored in the frame buffer at the client machine when a user issues the backward-play operation at frame n. For each DFMB, we are interested in obtaining $MB_{(k,l)}^{R_{n-1}}$ from the FB. If this can be done, no mismatch will be occurred since both reference frame and the prediction errors are come from the FB. We will now present in detail how a DFMB can be reconstructed from the FB provided that the frame n of the FB is available at the decoder. Rearranging (1), we obtain an expression for $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$

$$MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n}) = MB_{(k,l)}^{FB_n} - e_{(k,l)}^{FB_n} \quad (5)$$

When the requested $MB_{(k,l)}^{R_{n-1}}$ is found to be a DFMB, its corresponding macroblock in frame n of the FB. $MB_{(k,l)}^{FB_n}$, is coded without motion compensation. It means that the spatial position of $MB_{(k,l)}^{FB_{n-1}}$ in the FB is the same as that of $MB_{(k,l)}^{FB_n}$. Hence, for this specific case, $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$ is equal to $MB_{(k,l)}^{FB_{n-1}}$, and (5) can be rewritten as

$$MB_{(k,l)}^{FB_{n-1}} = MB_{(k,l)}^{FB_n} + \widetilde{e}_{(k,l)}^{FB_n} \quad (6)$$

where $\widetilde{e}_{(k,l)}^{FB_n} = -e_{(k,l)}^{FB_n}$. Note that the client machine has the frame n when a user issues the backward-play request at frame n. In other words, pixels of $MB_{(k,l)}^{FB_n}$ are available at the decoder. To reconstruct $MB_{(k,l)}^{FB_{n-1}}$ in the backward-play operation, (6) indicates that, for a DFMB, the only data that the server needs to send is the quantized DCT coefficients of $\widetilde{e}_{(k,l)}^{FB_n}$. In the following discussions, we will describe how to compute these quantized DCT coefficients of $\widetilde{e}_{(k,l)}^{FB_n}$ from the existing MPEG video stream in the server. By applying the DCT to $\widetilde{e}_{(k,l)}^{FB_n}$ and considering that the DCT is an odd transform, we can find the DCT of $\widetilde{e}_{(k,l)}^{FB_n}$ in the DCT-domain, as indicated below,

$$DCT(\widetilde{e}_{(k,l)}^{FB_n}) = -DCT(e_{(k,l)}^{FB_n}) \quad (7)$$

Then the quantized DCT coefficients of $\widetilde{e}_{(k,l)}^{FB_n}$ are given by

$$Q[DCT(\widetilde{e}_{(k,l)}^{FB_n})] = -Q[DCT(e_{(k,l)}^{FB_n})] \quad (8)$$

From (8), $Q[DCT(\widetilde{e}_{(k,l)}^{FB_n})]$ can be obtained by inverting the sign of all DCT coefficients in $Q[DCT(e_{(k,l)}^{FB_n})]$, which can be directly extracted from the FB of the server. From the above derivation, we can conclude that the server only needs to invert the sign of all DCT coefficients for each DFMB and send them to the client. The client machine uses the previously reconstructed frame in the frame buffer of the decoder as the reference frame which adds the inverted DCT coefficients to reconstruct the macroblocks classified as DFMB. Since both of reference frame and inverted DCT coefficients are from the FB, the reconstructed pixels of the DFMB are identical to corresponding pixels of the macroblock in the FB. No drift errors would be introduced in the DFMB.

Furthermore, for the same backward playback example, the next frame to be displayed is frame $n-2$ after frame $n-1$ has been decoded and displayed in the client machine. Now, the reconstructed pixels of frame $n-1$ are stored in the frame buffer of the decoder. Therefore, the proposed technique can be processed in the recursive way to reduce the drift problem. However, if the macroblock has already defined as non-DFMB, in which the drift errors have already introduced in this reconstructed macroblock. It is useless to employ the technique of the sign inversion of DCT coefficients for the remaining macroblock that have same spatial position.

III. SIMULATION RESULTS

In this section, we present some simulation results. All the test sequences have a length of *148* frames and were encoded

TABLE I

OVERALL AVERAGE PSNR COMPARISON FOR ALL SWITCHING POINTS.

Sequences	Bitrate	Dual-bitstream algorithm	Macroblock-based algorithm	Gain
Salesman	1.5M	36.59	37.48	0.80
(352x288)	3M	39.51	40.70	1.19
Football	1.5M	28.94	29.39	0.40
(352x240)	3M	32.58	33.38	0.80
Foreman	64K	25.69	25.91	0.22
(176x144)	128K	27.60	28.12	0.52
Carphone	64K	28.73	29.06	0.33
(176x144)	128K	31.72	32.30	0.58
Claire	64K	34.10	34.77	0.67
(176x144)	128K	39.65	40.87	1.22

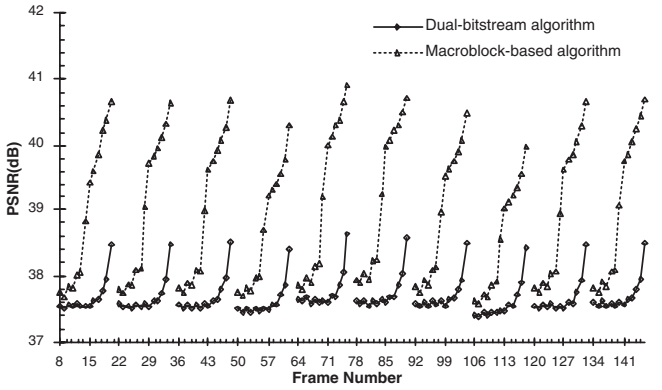

Fig. 2. The performance of the dual-bitstream algorithm and macroblock-based algorithm in the situation when the user requests a forward to backward-play operation at the end of each GOPs(RB) until received the next I-frame for the "Salesman" encoded at 3Mb/s.

at different bitrates. Each test sequence is encoded into two bitstreams, FB and RB, and I-frames in the RB are interleaved between I-frames in the FB. For all test sequences, the frame-rate of the video stream was *30* frames/s and the GOP length is *14* with an I-P structure. To evaluate the performance of the proposed macroblock-based algorithm, two simulations condition were carried out. First, we have simulated the situation from forward to backward-play operations on every possible switching points in which P-to-P frames bitstreams switching are occurred. Table I shows the PSNR performance of all requested frames $n-1$ where FB to RB bitstream switching at all switching points n are simulated. It is obvious that there is significant improvement for all video sequences, especially the slow motion sequences, such as "Salesman" and "Claire" sequences. For the high motion sequences such as "Football", the proposed algorithm achieves up to 0.8dB PSNR gain.

Second, we extend our simulations to the real MPEG video streaming condition. Actually, the drift errors will be introduced after performing bitstreams switching from FB to RB, and then these errors propagate and accumulate to the next displayed frame until the next I-frame. Fig. 2 demonstrate the performance of the dual-bitstream and macroblock-based algorithm in the situation that bitstream switching occurs at the switching point that have initiated longest duration of drift propagation. As shown in Fig. 2, the performance of macroblock-based algorithm outperforms the dual-bitstream algorithm without error compensation in all GOPs. However, the quality of the requested frames after bitstream switching drops gradually as the distance from the switching point increase. It is due to the fact that the number of non-DFMBs increases and the drift errors accumulate from frame to next requested frame, however, the quality would be better than the dual-bitstream algorithm.

IV. CONCLUSION

In this paper, we proposed an efficient macroblock-based selection techniques for the dual-bitstream system. The proposed techniques are motivated by the center-biased motion vector distribution characteristics of real-world video sequences. With the motion information, the video streaming server organizes the macroblocks in the requested frame into two categories - a drift-free macroblock (DFMB) and a non-drift-free macroblock (non-DFMB). Then it selects the necessary macroblocks adaptively, processes them in the compressed-domain and sends the processed macroblocks to the client machine. Since the re-encoding of DFMB is not required, the visual quality for DFMB during reverse playback will be exactly the same as that of forward playback. Simulation results show that, with our proposed scheme, the MPEG video streaming system with reverse-play functionality can alleviate the drift problem significantly.

ACKNOWLEDGMENT

The work described in this paper is partially supported by the Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, Hong Kong Polytechnic University and a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (PolyU 5216/03E). Chang-Hong Fu acknowledges the research studentships provided by the University.

REFERENCES

[1] ISO/IEC 11172-2, "*Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1,5 Mbit/s – Part 2: Video*", 1993

[2] ISO/IEC 13818-2, "*Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Video*", 1996.

[3] M. S. Chen and D. D. Kandlur, "*Downloading and stream conversion: supporting interactive playout of videos in a client station*", in Proc. 2nd Int. IEEE Conf. Multimedia Computing and Systems, pp. 73-80, 1995.

[4] S. J. Wee, "*Reversing motion vector fields*", in Proc. IEEE International on Conference Image Processing 1998 (ICIP98), pp. 209-212, October 1998.

[5] S. J. Wee and B. Vasudev, "*Compressed-domain reverse play of MPEG video streams*", in Proc. SPIE Conf. Multimedia Systems and Applications, pp. 237-248, November 1998.

[6] C. W. Lin, J. Zhou, J. Youn, and M. T. Sun, "*MPEG video streaming with VCR functionality*", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 3, pp. 415-425, March 2001.