New Architecture for MPEG Video Streaming System With Backward Playback Support

Chang-Hong Fu, Yui-Lam Chan, *Member, IEEE*, Tak-Piu Ip, and Wan-Chi Siu, *Senior Member, IEEE*

***Abstract*—MPEG digital video is becoming ubiquitous for video storage and communications. It is often desirable to perform various video cassette recording (VCR) functions such as backward playback in MPEG videos. However, the predictive processing techniques employed in MPEG severely complicate the backward-play operation. A straightforward implementation of backward playback is to transmit and decode the whole group-of-picture (GOP), store all the decoded frames in the decoder buffer, and play the decoded frames in reverse order. This approach requires a significant buffer in the decoder, which depends on the GOP size, to store the decoded frames. This approach could not be possible in a severely constrained memory requirement. Another alternative is to decode the GOP up to the current frame to be displayed, and then go back to decode the GOP again up to the next frame to be displayed. This approach does not need the huge buffer, but requires much higher bandwidth of the network and complexity of the decoder. In this paper, we propose a macroblock-based algorithm for an efficient implementation of the MPEG video streaming system to provide backward playback over a network with the minimal requirements on the network bandwidth and the decoder complexity. The proposed algorithm classifies macroblocks in the requested frame into backward macroblocks (BMBs) and forward/backward macroblocks (FBMBs). Two macroblock-based techniques are used to manipulate different types of macroblocks in the compressed domain and the server then sends the processed macroblocks to the client machine. For BMBs, a VLC-domain technique is adopted to reduce the number of macroblocks that need to be decoded by the decoder and the number of bits that need to be sent over the network in the backward-play operation. We then propose a newly mixed VLC/DCT-domain technique to handle FBMBs in order to further reduce the computational complexity of the decoder. With these compressed-domain techniques, the proposed architecture only manipulates macroblocks either in the VLC domain or the quantized DCT domain resulting in low server complexity. Experimental results show that, as compared to the conventional system, the new streaming system reduces the required network bandwidth and the decoder complexity significantly.**

***Index Terms*—Compressed-domain processing, digital video cassette recording, MPEG video, streaming video.**

Manuscript received December 13, 2004; revised May 8, 2007. This work was supported in part by the Centre for Multimedia Signal Processing, in part by the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, and in part by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (PolyU 5204/04E). C.-H. Fu was supported by the Hong Kong Polytechnic University research studentships.

The authors are with the Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: enchfu@polyu.edu.hk; enylchan@polyu.edu.hk; enbill@polyu.edu.hk; enwcsiu@polyu.edu.hk).

Digital Object Identifier 10.1109/TIP.2007.902330

I. Introduction

VIDEO streaming applications have received tremendous attention from both academia and industry in recent years [1]–[6], mainly due to the emergence of efficient multimedia compression and broadband networking technologies [7]–[10]. With the proliferation of online multimedia content, it is highly desirable that video streaming systems should have the capability of providing fast and effective browsing. However, various video coding standards nowadays [8]–[10] were developed primarily for forward playback. In order to complete the transition to digital video from its current analog state, the MPEG technology needs to encompass not just compression and streaming methodologies but also a video-processing framework. This will allow MPEG to be usable not just for the purposes of efficient storage and transmission of digital videos, but also for systems wherein the user needs to interact with the digital videos. A key technique that enables fast and user-friendly browsing of video content is to provide full video cassette recording (VCR) functionality such as forward, backward, stop, pause, fast forward, fast backward, and random access.

The predictive processing techniques employed in MPEG [8]–[10] severely complicate the backward-play operations. For uncompressed videos, the solution for backward playback is just to reorder the video frame data in reverse order. The simplicity of this solution relies on two properties: the data for each video frame is self-contained and it is independent of its placement in the data stream. These properties typically do not hold true for MPEG video data because MPEG compression uses predictive processing techniques that are not invariant to changes in frame order. In other words, simply reversing the order of the input frame data will not reverse the order of the decoded video frames. For example, consider the case with simple I-P structure of MPEG encoded sequence. If the requested frame is an I-frame, the server only needs to send this frame, and the decoder can decode it immediately. However, if the requested frame is a P-frame, the server needs to send all the P-frames from the previous nearest I-frame to this requested frame. For example, consider the case as shown in Fig. 1. Suppose frame n is the starting point of backward playback. Since the next frame to be displayed is frame $n-1$, the server sends frame 0 to frame $n-1$ from the MPEG video stream in forward order. At the client side, frame 0 to frame $n-2$ are decoded. However, they do not need to be displayed and only frame $n-1$ should be displayed on the client's screen. Frame $n-1$ is then decoded and stored into the display buffer so that this frame is displayed on the client's screen. If the

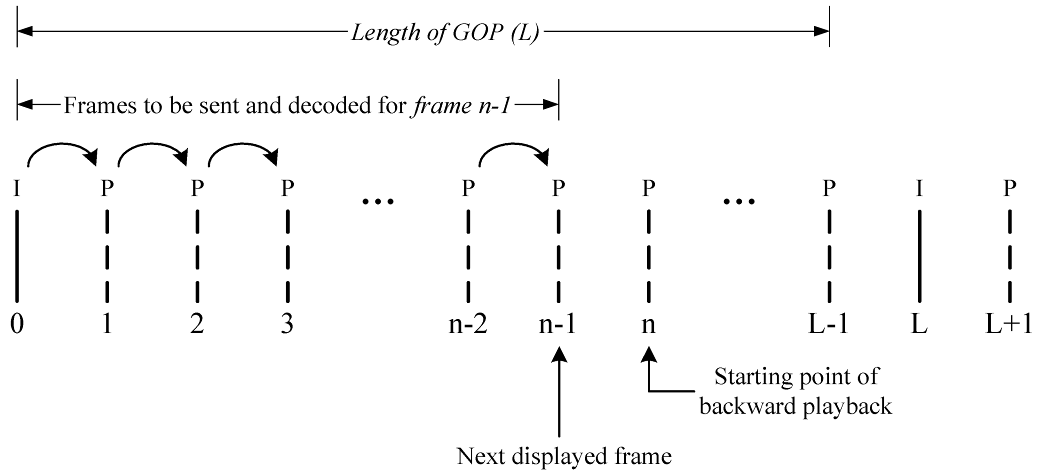

Length of GOP (L)
Frames to be sent and decoded for frame n-1
I P P P ... P P P ... P I P
0 1 2 3 n-2 n-1 n L-1 L L+1
Starting point of backward playback
Next displayed frame

Fig. 1. Example of backward playback.

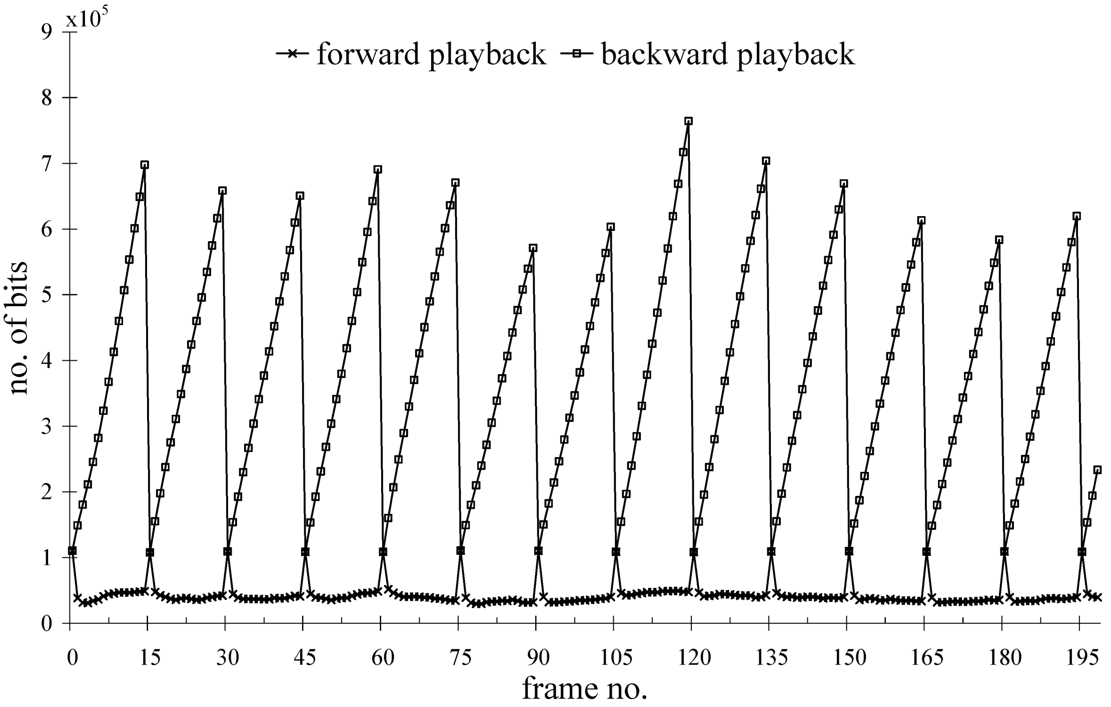


Fig. 2. Bandwith requirement for sending the "Salesman" sequence over network with respect to the forward-play and backward-play operations.

decoder of the client side has huge buffer, it can store all the decoded frames of the whole GOP and play the decoded frames in the buffer backward. This approach is impractical when the GOP size is large or the decoder has limited memory. Another possible solution to implement the backward-play operation is to send frame 0 to frame $n-2$ from the server and these frames are decoded by the client again. This process continues until the backward-play operation is stopped. Consider that when a backward-play operation is requested, it always lasts for few seconds or minutes. During the period of backward playback, a large number of frames need to be sent over the network and decoded by the client decoder. The impact of backward playback on the decoding complexity and the network traffic is depicted in Fig. 2 where the test video stream used for simulation is "Salesman" sequence with a length of 200 frames. The "Salesman" sequence is encoded at 1.5 Mb/s with a frame-rate of 30 frames/s and the length of group-of-picture (GOP) is 15 with an I-P structure. The starting point of the backward-play operation is at the end of the sequence. This figure shows that the server needs to send an excessively large amount of extra bits to the decoder to display one frame during backward playback. Thus, a straightforward implementation of the backward-play operation requires much higher network bandwidth and decoder complexity as compared to those required for the forward-play operation.

Recently, some works on the implementation of backward playback for an MPEG compressed video in streaming applications have been introduced [11]–[14]. Chen and Kandlur [11] 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 results in breaking the interframe dependencies between the P-frames and the I-frames. After the frame conversion and frame re-ordering, the motion-vector reversing approach developed in [12] 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 [13] described a backward-play transcoder which is used to convert the I-P frames into another I-P bitstream with reverse order. A method of estimating the reverse motion vectors for the new I-P bitstream based on the forward motion vectors of the original I-P bitstream as described in [12] is adopted to reduce the computational complexity of this transcoding process. The transcoding process, however, still requires much computation and will cause drift due to the motion vector approximation [13]. The dual-bitstream approach [14] was recently proposed to store the forward-encoded bitstream, as well as the backward-encoded bitstream in the server to simplify the backward-play complexity while maintaining the low network bandwidth requirement. However, it approximately doubles the storage requirement of the server.

In this paper, we will suggest some macroblock-based solutions for providing efficient backward playback. By exploring the motion information of the compressed bitstream, the proposed scheme can adaptively select the necessary macroblocks, manipulate them in the compressed domain and send the processed macroblocks to the client. Since the proposed scheme mainly operates in 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 presents the proposed video streaming architecture with the new macroblock-based techniques for backward playback. In Section III, we show a technique which operates on the variable length code (VLC) domain of the proposed macroblock-based system. Section IV describes a mixed VLC/discrete cosine transform (DCT) domain technique which can further enhance the efficiency of the proposed system. Simulation results are then presented in Section V. Finally, some concluding remarks are provided in Section VI.

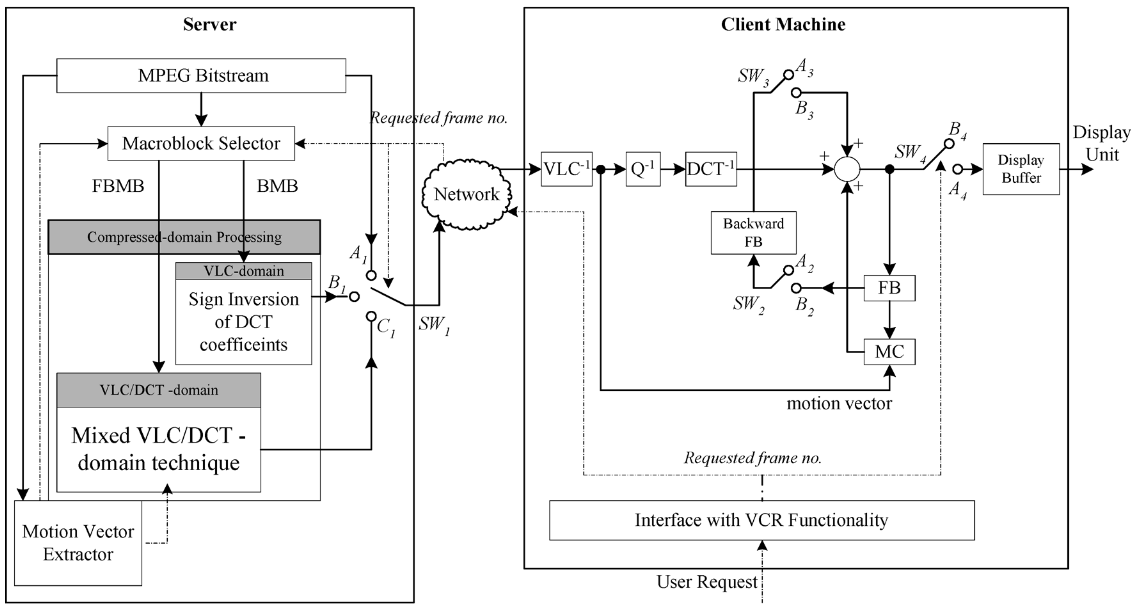
II. Proposed Architecture Using the Macroblock-Based Algorithm for Backward Playback

In [15] and [16], we proposed macroblock-based techniques for providing backward-play service of a video streaming system with VCR support to reduce the requirements of the decoder complexity and network traffic. In this paper, we present a new macroblock-based video streaming architecture which is an extended work of [15] and [16]. The architecture of the proposed system is shown in Fig. 3. Consider that a precompressed video stream using the MPEG-2 video coding standard [9], [17] is stored in a storage device. Upon the client request, the streaming server retrieves the compressed video data from the storage device and the user can view the video while the video is being streamed over the network. To support VCR services, considerable functionality must be built into the client machine which consists of an MPEG decoder and an interface for VCR functionality. The use of the MPEG decoder is to decode the incoming video stream and deliver it to the client's screen. The VCR interface then translates user interactions from the remote control or keyboard to appropriate signals for network transfer. It interprets the user commands and forwards them to the decoder and the server for appropriate actions.

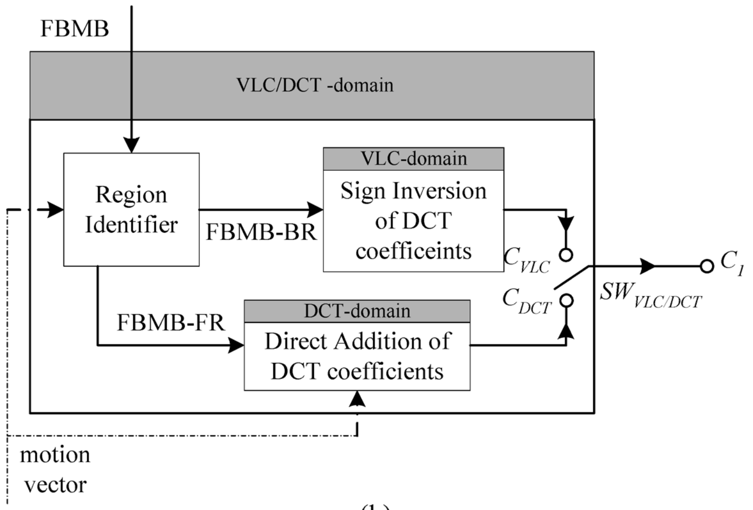
In Fig. 3(a), there are a total of four switches SW_1, SW_2, SW_3, and SW_4 in the server and the client machine. They are used to enable various VCR operations. In the forward-play operation, switches SW_1, SW_2, SW_3, and SW_4 are connected to A_1, A_2, A_3, and A_4, respectively, as illustrated in Table I. In this mode, the proposed architecture is the same as the conventional frame-based architecture. On the other hand, in contrast to the frame-based scheme used in the conventional architecture, a macroblock-based scheme is proposed for use in the backward-play operation. The switch positions for the backward-play mode are also shown in Table I, which will be described in Sections III and IV. At the server side of Fig. 3(a), motion vectors are extracted from the MPEG bitstream and a macroblock selector utilizes these motion vectors to identify two different types of macroblocks. Again, we use the example in Fig. 1 for illustration. Let us assume that a user requests a backward-play command at frame n, the next frame to be displayed is frame $n-1$. Note that B-frames are not used as references for later frames. It means they are not involved in decoding other frames. For simplicity, but without loss of generality, we focus our discussions on the case that the MPEG bitstream contains I-and P-frames only. The situation in macroblock level is depicted in Fig. 4. We assume that $\mathrm{MB}_{(k,l)}^{n-1}$ represents the macroblock at the k^{th} row and l^{th} column of frame $n-1$ (the next displayed frame). $\mathrm{MB}_{(k,l)}^{n-1}$ is defined as a backward macroblock (BMB) if the macroblock in frame n having the same spatial position of $\mathrm{MB}_{(k,l)}^{n-1}$, i.e. $\mathrm{MB}_{(k,l)}^{n}$, is coded without motion compensation (non-MC macroblock). Otherwise, it is defined as a forward/backward macroblock (FBMB). The reconstruction of BMBs in the backward direction which is opposite to the encoding direction of the original MPEG video stream will be described in detail later. BMBs use only the MPEG data of the future frame while FBMBs need the MPEG data from both the past and future frames. For example, in Fig. 4, since the motion vector of $\mathrm{MB}_{(0,1)}^{n}$,$mv_{(0,1)}^{n}$, is zero, it means that $\mathrm{MB}_{(0,1)}^{n}$ is a non-MC macroblock and the macroblock selector classifies $\mathrm{MB}_{(0,1)}^{n-1}$ as a BMB. On the other hand, since $\mathrm{MB}_{(1,1)}^{n}$ is coded with motion compensation (MC-macroblock), $\mathrm{MB}_{(1,1)}^{n-1}$ is categorized as a FBMB. In this paper, our scheme works at the level of macroblocks. Our contributions are:

1) to adopt a VLC-domain technique for BMBs [15], [16] in the proposed architecture;
2) to design a mixed VLC/DCT-domain technique for FBMBs.

Since the server will process each type of macroblock in the compressed domain, complete decoding and encoding are not required at the server and the increase in the computational



(a)



(b)

Fig. 3. Proposed video streaming system with VCR functionality: (a) the overall architecture, and (b) illustration of the mixed VLC/DCT-domain technique for FBMBs.

burden of the server is negligible. The advantages of the proposed video streaming system, together with the details of the novel techniques, are described in the following sections.

III. VLC-Domain Techniques for BMBs

In motion-compensated prediction [8]–[10], the previously decoded frame serves as the prediction for the current frame. The difference between the predicted and the actual current frame is the prediction error. The coded prediction error is added to the prediction to obtain the final representation of the current reconstructed frame. At the decoder of Fig. 3(a), each macroblock in frame n, $\mathrm{MB}^n_{(k,l)}$, is reconstructed as

$$\mathrm{MB}^n_{(k,l)} = \mathrm{MCMB}^{n-1}\left(mv^n_{(k,l)}\right) + e^n_{(k,l)} \tag{1}$$

where $\mathrm{MCMB}^{n-1}(mv^n_{(k,l)})$ represents the motion-compensated macroblock of $\mathrm{MB}^n_{(k,l)}$ which is translated by the motion vector $mv^n_{(k,l)}$ in the reconstructed frame $n-1$ and $e^n_{(k,l)}$ is the prediction error between $\mathrm{MB}^n_{(k,l)}$ and its motion-compensated macroblock,$\mathrm{MCMB}^{n-1}(mv^n_{(k,l)})$. Frame n is then stored in the frame buffer (FB) since it is used for decoding the subsequent frame $n+1$ during forward playback. When a backward-play command is requested at frame n, the next frame to be displayed is frame $n-1$. It means that all $\mathrm{MB}^{n-1}_{(k,l)}$in frame $n-1$ are requested. To reconstruct each $\mathrm{MB}^{n-1}_{(k,l)}$, all the related previous macroblocks in P-/I-frames need to be sent over the network and decoded by the decoder in the conventional video streaming system. It becomes impractical when the GOP size is large. However, if $\mathrm{MB}^{n-1}_{(k,l)}$ is found to be a BMB, its corresponding macroblock in frame n, $\mathrm{MB}^n_{(k,l)}$, is coded without motion compensation. It means that the spatial

TABLE I
SWITCH POSITIONS OF THE PROPOSED VIDEO STREAMING SYSTEM

Playback modes	Macroblock type	SW_1	SW_2	SW_3	SW_4	$SW_{VLC/DCT}$
Forward	—	A_1	A_2	A_3	A_4	—
Backward	BMB	B_1	B_2	A_3	B_4	—
	FBMB-BR	C_1	B_2	A_3	B_4	C_{DCT}
	FBMB-FR (nearest I-frame to frame *n-2*)	C_1	A_2	A_3	B_4	C_{DCT}
	FBMB-FR (frame *n-1*)	C_1	A_2	B_3	A_4	C_{VLC}

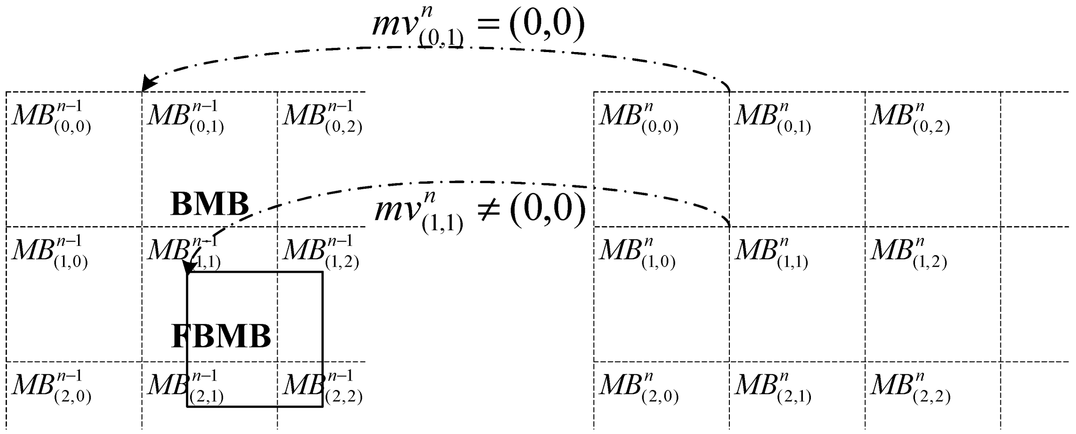


Fig. 4. Definition of the BMB and the FBMB.

position of $\mathrm{MB}^{n-1}_{(k,l)}$ is the same as that of $\mathrm{MB}^{n}_{(k,l)}$. Hence, for this specific case, $\mathrm{MCMB}^{n-1}(mv^n_{(k,l)})$ is equal to $\mathrm{MB}^{n-1}_{(k,l)}$, and (1) can be rewritten as

$$\mathrm{MB}^{n-1}_{(k,l)} = \mathrm{MB}^{n}_{(k,l)} + \tilde{e}^{n}_{(k,l)} \tag{2}$$

where $\tilde{e}^{n}_{(k,l)} = -e^{n}_{(k,l)}$. Note that frame n is stored in FB at the client machine when a user issues the backward-play operation at frame n. In other words, pixels of $\mathrm{MB}^{n}_{(k,l)}$ are available at the decoder. To reconstruct $\mathrm{MB}^{n-1}_{(k,l)}$ in the backward-play operation, (2) indicates that, for a BMB, the only data that the server needs to send is the quantized DCT coefficients of $\tilde{e}^{n}_{(k,l)}$. In the following discussions, we will describe how to compute these quantized DCT coefficients of $\tilde{e}^{n}_{(k,l)}$ from the existing MPEG video stream in the server [15], [16].

By applying the DCT to $\tilde{e}^{n}_{(k,l)}$ and considering that the DCT is an odd transform, we can find the DCT of $\tilde{e}^{n}_{(k,l)}$ in the DCT-domain, as indicated in the following:

$$\mathrm{DCT}\left(\tilde{e}^{n}_{(k,l)}\right) = -\mathrm{DCT}\left(e^{n}_{(k,l)}\right). \tag{3}$$

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

$$Q\left[\mathrm{DCT}\left(\tilde{e}^{n}_{(k,l)}\right)\right] = -Q\left[\mathrm{DCT}\left(e^{n}_{(k,l)}\right)\right]. \tag{4}$$

From (4), $Q[\mathrm{DCT}(\tilde{e}^{n}_{(k,l)})]$ can be obtained by inverting the sign of all DCT coefficients in $Q[\mathrm{DCT}(e^{n}_{(k,l)})]$, which can be directly extracted from the MPEG video stream in the server. $Q[\mathrm{DCT}(\tilde{e}^{n}_{(k,l)})]$ is then transmitted to the client by switching SW_1 to B_1. At the client side, as shown in Fig. 3(a), switch SW_2 is connected to B_2 so that all reconstructed BMBs are stored in the backward frame buffer (backward-FB) which is an additional frame buffer in the client machine for backward playback. The reconstructed BMBs stored in the backward-FB are used for further composition of FBMBs. From the above derivation, we can conclude that the server and the client only need to send and decode the prediction errors of one macroblock for each BMB, respectively. For a real-world image sequence, the block motion field is usually gentle, smooth, and varies slowly. As a consequence, the distribution of motion vector is center biased [18]–[20], as demonstrated by the typical examples as shown in Table II which shows the distribution of BMB for various sequences including “Claire,” “Grandma,” “Salesman,” “Carphone,” “Foreman,” “Table Tennis,” and “Football.” These sequences have been selected to emphasize different amount of motion activities. It is clear that over 90% and 27% of the macroblocks are classified as BMBs for sequences containing low and high amount of motion activities respectively. By inverting the sign of all DCT coefficients in the server, the sequence containing more BMBs can alleviate the decoder complexity and the network traffic significantly.

In the video streaming server, the sign inversion of DCT coefficients requires additional variable length decoding and re-encoding. To reduce the computational load of the server, the newly quantized DCT coefficients $Q[\mathrm{DCT}(\tilde{e}^{n}_{(k,l)})]$ can be computed in the VLC domain [15], [16].

For encoding of the quantized DCT coefficients, they are arranged into a 1-D array following the zigzag scan order. This scan order puts the low-frequency coefficients in front of the

TABLE II
PERCENTAGE OF BMB FOR VARIOUS SEQUENCES

Claire	Grandma	Salesman	Carphone	Table Tennis	Foreman	Football
89.75	81.57	61.26	52.37	49.30	43.59	27.51

TABLE III
VLC TABLE FOR RUN-LEVEL COMBINATIONS. THE SIGN BIT "s" "0" FOR POSITIVE AND "1" FOR NEGATIVE

Variable length codes	Run	Level
10	End of Block	
11 s	0	1
011 s	1	1
0100 s	0	2
:	:	:
0000 01	Escape	
:	:	:

high-frequency coefficients. Since visually weighted quantization strongly deemphasizes higher spatial frequencies, only a few lower-frequency coefficients are nonzero in a typical block. Thus, the zigzag scan order puts the longest runs of zeros at the end of the 1-D array such that the EOB (end-of-block) symbol can efficiently code all of these trailing zero coefficients with a single codeword. Typically, the EOB occurs well before the midpoint of the array. Runs of zero coefficients also occur quite frequently before the EOB. In this case, better coding efficiency is obtained when codewords are defined by combining the run of zero coefficients with the amplitude of the nonzero coefficient terminating the run. Each nonzero sequence of DCT coefficients is then coded in the RUN-LEVEL symbol structure with different VLCs. RUN refers to the number of zero coefficients before the next nonzero coefficient; LEVEL refers to the amplitude of the nonzero coefficient. Table III illustrates this. The trailing bit of each VLC is the "s" bit which codes the sign of the nonzero coefficient. If "s" is 0, the coefficient is positive; otherwise, it is negative.

To convert $Q[\mathrm{DCT}(\tilde{e}^n_{(k,l)})]$ from $Q[\mathrm{DCT}(e^n_{(k,l)})]$, the server just parses the MPEG video bitstream and inverts all "s" bits of VLCs in each BMB. On the other hand, RUN-LEVEL combinations that are not in the Table III are coded using a 6-bit "Escape" code followed by a 6-bit fixed length code (FLC) for RUN and a 12-bit FLC for LEVEL. The FLCs for RUN and LEVEL are shown in Table IV. In this case, the 12-bit FLC for LEVEL is converted into its 2's complement. The bit manipulation of VLCs in the BMB is summarized in Fig. 5. Since it is not necessary to perform VLC encoding, motion compensation, DCT, quantization, inverse DCT, inverse quantization and VLC decoding in the server, the loading of the server is reduced significantly.

IV. MIXED VLC/DCT-DOMAIN TECHNIQUE FOR FBMBS

For FBMBs, the situation is different. The bit manipulation of VLCs mentioned in Section III cannot be directly applied to FBMBs since $\mathrm{MCMB}^{n-1}(mv^n_{(k,l)})$ is no longer equal to $\mathrm{MB}^{n-1}_{(k,l)}$ and (2) does not hold true for FBMBs. In other words, $\mathrm{MB}^{n-1}_{(k,l)}$ cannot be reconstructed from $\mathrm{MB}^n_{(k,l)}$. To reconstruct FBMBs of frame $n-1$ directly, the server will examine the motion vectors in the MPEG video stream and all the related macroblocks from the previous nearest I-frame to frame $n-2$ should be transmitted and decoded. In Fig. 6, a situation in which $\mathrm{MB}^{n-1}_{(1,1)}$ is a FBMB and its corresponding motion vector is $mv^{n-1}_{(1,1)}$ is illustrated. Two macroblocks (the shaded macroblocks) in frame $n-2$ are required to act as references for performing motion compensation of $\mathrm{MB}^{n-1}_{(1,1)}$. These macroblocks in frame $n-2$ further requires their corresponding macroblocks in frame $n-3$. This process continues until the previous nearest I-frame. In this example, the server needs to send seven macroblocks for $\mathrm{MB}^{n-1}_{(1,1)}$ from frame $n-1$ to frame $n-3$.

To further reduce the decoding complexity and network traffic in processing FBMBs, we suggest exploring the redundancies among the macroblocks that are required for the reconstruction of FBMBs. Let us use Fig. 7 to give a clearer account of our idea for reconstructing FBMBs. In Fig. 7, $\mathrm{MB}^{n-1}_{(1,1)}$ is a FBMB whose pixels are divided into two regions. The macroblock enclosed by the thick line in frame $n-1$ is the motion-compensated macroblock of $\mathrm{MB}^n_{(1,1)}$, $\mathrm{MCMB}^{n-1}(mv^n_{(1,1)})$. Pixels in $\mathrm{MB}^{n-1}_{(1,1)}$ which overlap with $\mathrm{MCMB}^{n-1}(mv^n_{(1,1)})$ belong to a backward region (FBMB-BR). Otherwise, they are considered as a forward region (FBMB-FR). In the following, we show that the VLC-domain technique can still be used for FBMB-BRs so that all pixels of a FBMB-BR would be backward reconstructed from the future frame in order to reduce the decoder and channel burdens. On the other hand, the pixels of the corresponding FBMB-FR are forward reconstructed from the past frames. Section IV-A describes how the VLC-domain technique can be applied to FBMB-BRs. Section IV-B presents a DCT-domain technique for the case of FBMB-FRs to further improve the performance of the proposed video streaming system.

A. VLC-Domain Technique for FBMB-BR

The VLC-domain technique mentioned in Section III helps to reduce both network traffic and decoder complexity. Besides, this technique can also minimize the computational complexity required for the server since it only processes macroblocks in the VLC domain. In order to keep the benefits of the VLC-domain technique, we propose to process the MPEG bitstream as much in the VLC domain as possible. Specifically, we propose to reconstruct all pixels in each FBMB-BR in the VLC domain according to the following formulation. For block motion-compensated prediction, each macroblock in frame n, $\mathrm{MB}^n_{(k,l)}$, is reconstructed by motion-compensated prediction and it is given by (1), which can be rewritten as

$$\mathrm{MCMB}^{n-1}\left(mv^n_{(k,l)}\right) = \mathrm{MB}^n_{(k,l)} + \tilde{e}^n_{(k,l)} \tag{5}$$

where, $\tilde{e}^n_{(k,l)} = -e^n_{(k,l)}$. Note that pixel values of $\mathrm{MB}^n_{(k,l)}$ are available at the decoder. Equation (5) implies that pixels in

TABLE IV
FLC TABLE FOR RUNS AND LEVELS. IT IS USED FOLLOWING THE ESCAPE CODE OF A VLC

FLCs	Run
0000 00	0
0000 01	1
:	:
:	:
:	:
1111 11	63

FLCs	Signed_level
1000 0000 0001	-2047
1000 0000 0010	-2046
:	:
0000 0000 0001	+1
:	:
0111 1111 1111	+2047

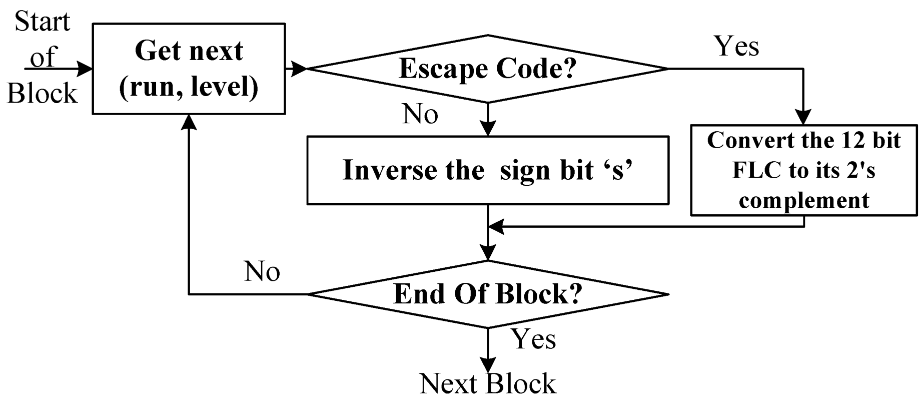


Fig. 5. Execution flow of the server during bit manipulation of VLCs in a BMB.

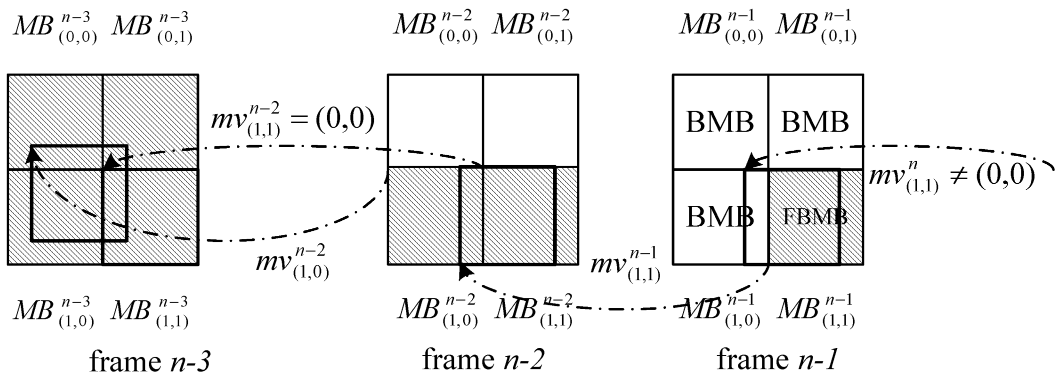

$MB^{n-3}_{(0,0)}$ $MB^{n-3}_{(0,1)}$
$MB^{n-2}_{(0,0)}$ $MB^{n-2}_{(0,1)}$
$MB^{n-1}_{(0,0)}$ $MB^{n-1}_{(0,1)}$
$mv^{n-2}_{(1,1)} = (0,0)$
BMB
BMB
BMB
FBMB
$mv^{n}_{(1,1)} \neq (0,0)$
$mv^{n-2}_{(1,0)}$
$mv^{n-1}_{(1,1)}$
$MB^{n-3}_{(1,0)}$ $MB^{n-3}_{(1,1)}$
$MB^{n-2}_{(1,0)}$ $MB^{n-2}_{(1,1)}$
$MB^{n-1}_{(1,0)}$ $MB^{n-1}_{(1,1)}$
frame *n-3*
frame *n-2*
frame *n-1*

Fig. 6. Situation in which there is one FBMB in frame $n-1$.

the FBMB-BR can be reconstructed by only sending the sign inversion of the quantized DCT coefficients of $e^n_{(k,l)}$. Similar to the technique used in BMBs, all these sign-inverted coefficients can be generated in the VLC domain. In other words, the pixels in the requested FBMB of frame $n-1$ overlapping with $\mathrm{MCMB}^{n-1}(mv^n_{(1,1)})$ (FBMB-BR) are computed from frame n, which is already stored in the client machine. This signifies that the server only needs to send the prediction errors of one macroblock to the client side for decoding the FBMB-BR.

We will now explain the significance of using the VLC-domain technique in the FBMB-BR. Let us refer to the example in Fig. 6 again. We can see that seven macroblocks are required for the reconstruction of $\mathrm{MB}^n_{(1,1)}$. The situation gets worse when the requested FBMB is far away from the previous nearest I-frame. However, by applying the VLC-domain technique, we find that only FBMB-FR of the requested FBMB needs to be reconstructed from the referenced macroblocks in the previous frames. This is illustrated in the example of Fig. 7. In this example, only the shaded region of $\mathrm{MB}^{n-1}_{(1,1)}$ (FBMB-FR) is reconstructed from the previous frames. In frame $n-2$, only $\mathrm{MB}^{n-2}_{(1,1)}$ is actually required. Although another macroblock $\mathrm{MB}^{n-2}_{(1,0)}$ is also covered by the motion-compensated macroblock of $\mathrm{MB}^{n-1}_{(1,1)}$, it is no longer required. The reason is that the FBMB-BR of $\mathrm{MB}^{n-1}_{(1,1)}$ can be predicted from frame n which is available at the decoder. Similarly, in frame $n-3$, only $\mathrm{MB}^{n-3}_{(0,1)}$ needs to be sent over the network and decoded by the decoder. The necessary macroblocks used to reconstruct $\mathrm{MB}^{n-1}_{(1,1)}$ can be reduced considerably. Altogether, instead of sending seven macroblocks as depicted in Fig. 6, the server needs to transmit only four macroblocks, including three macroblocks from the previous frames and one macroblock from frame n for the FBMB-FR and FBMB-BR, respectively. If the length of GOP is longer, the savings of the proposed technique could be even larger.

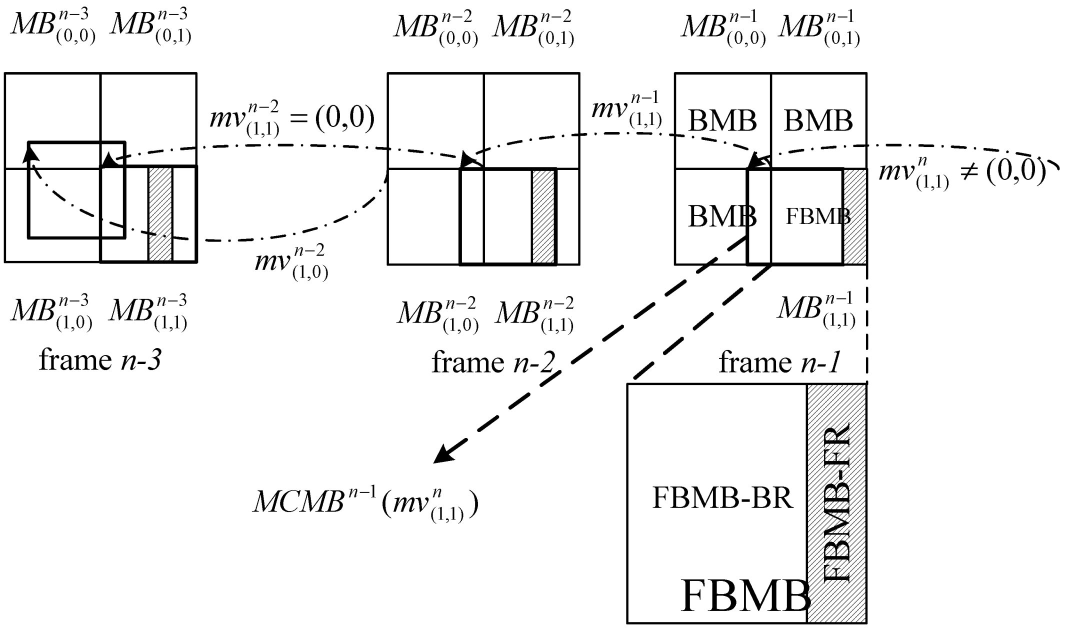

$MB^{n-3}_{(0,0)}$ $MB^{n-3}_{(0,1)}$
$MB^{n-2}_{(0,0)}$ $MB^{n-2}_{(0,1)}$
$MB^{n-1}_{(0,0)}$ $MB^{n-1}_{(0,1)}$
$mv^{n-2}_{(1,1)} = (0,0)$
$mv^{n-1}_{(1,1)}$
BMB BMB
BMB FBMB
$mv^{n}_{(1,1)} \neq (0,0)$
$mv^{n-2}_{(1,0)}$
$MB^{n-3}_{(1,0)}$ $MB^{n-3}_{(1,1)}$
$MB^{n-2}_{(1,0)}$ $MB^{n-2}_{(1,1)}$
$MB^{n-1}_{(1,1)}$
frame n-3
frame n-2
frame n-1
$MCMB^{n-1}(mv^{n}_{(1,1)})$
FBMB-BR
FBMB-FR
FBMB

Fig. 7. Illustration of the FBMB-BR and the FBMB-FR.

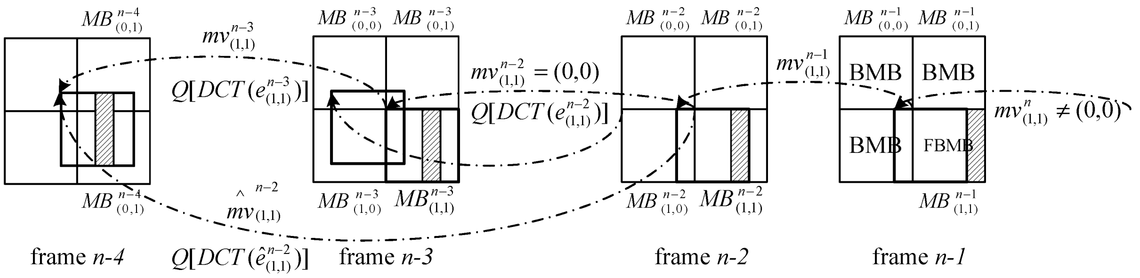

$MB^{n-4}_{(0,1)}$
$mv^{n-3}_{(1,1)}$
$Q[DCT(e^{n-3}_{(1,1)})]$
$MB^{n-3}_{(0,0)}$ $MB^{n-3}_{(0,1)}$
$mv^{n-2}_{(1,1)} = (0,0)$
$Q[DCT(e^{n-2}_{(1,1)})]$
$MB^{n-2}_{(0,0)}$ $MB^{n-2}_{(0,1)}$
$mv^{n-1}_{(1,1)}$
$MB^{n-1}_{(0,0)}$ $MB^{n-1}_{(0,1)}$
BMB BMB
BMB FBMB
$mv^{n}_{(1,1)} \neq (0,0)$
$MB^{n-4}_{(0,1)}$
$\hat{mv}^{n-2}_{(1,1)}$
$MB^{n-3}_{(1,0)}$ $MB^{n-3}_{(1,1)}$
$MB^{n-2}_{(1,0)}$ $MB^{n-2}_{(1,1)}$
$MB^{n-1}_{(1,1)}$
frame n-4
$Q[DCT(\hat{e}^{n-2}_{(1,1)})]$
frame n-3
frame n-2
frame n-1

Fig. 8. Direct addition of DCT coefficients for a FBMB-FR.

B. DCT-Domain Technique for FBMB-FR

We are now interested in obtaining pixels of each FBMB-FR from the previous frames. To further enhance the efficiency of the proposed system, we suggest using a technique of direct addition of DCT coefficients for FBMB-FRs. This technique was originally designed for the frame-skipping video transcoder which is mainly performed in the DCT domain to achieve a transcoder with low complexity [21]–[23]. Let us borrow this idea to reconstruct FBMB-FRs when one of related macroblocks of the requested FBMB is coded without motion compensation (non-MC macroblock) and further extend it to alleviate the computational burden of the client decoder in the backward-play operation.

Fig. 8 further extends the process of motion compensation to frame $n-4$ of the example as shown in Fig. 7. Among those referenced macroblocks of $\mathrm{MB}^{n-1}_{(1,1)}$, $\mathrm{MB}^{n-2}_{(1,1)}$ is a non-MC macroblock since its motion vector, $mv^{n-2}_{(0,1)}$, is equal to zero. In the reconstruction process of $\mathrm{MB}^{n-1}_{(1,1)}$, $\mathrm{MB}^{n-2}_{(1,1)}$ is needed. Therefore, the decoder needs to decode the prediction errors $e^{n-2}_{(1,1)}$ in frame $n-2$, $e^{n-3}_{(1,1)}$ in frame $n-3$, and so on. If pixels in $\mathrm{MB}^{n-3}_{(1,1)}$ are used as reference for $\mathrm{MB}^{n-2}_{(1,1)}$ only, it is too wasteful for the client machine to decode $e^{n-3}_{(1,1)}$. Thus, in the proposed scheme, the server employs the DCT-domain technique to combine $e^{n-2}_{(1,1)}$ and $e^{n-3}_{(1,1)}$ in one single macroblock for the non-MC macroblock. The required number of macroblocks which are decoded by the decoder is then reduced.

Now, let us formulate an efficient way to combine $e^{n-2}_{(1,1)}$ and $e^{n-3}_{(1,1)}$ in the server for $\mathrm{MB}^{n-2}_{(1,1)}$. When pixels in $\mathrm{MB}^{n-3}_{(1,1)}$ are not decoded directly in the client machine, it means that the incoming quantized DCT coefficients of the prediction error from the original video stream, $Q[\mathrm{DCT}(e^{n-2}_{(1,1)})]$, are no longer valid because they refer to the pixels which are not stored in FB. The server needs to compute the new motion vector $\hat{mv}^{n-2}_{(1,1)}$ and prediction errors in the quantized DCT domain, $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$, by using frame $n-4$ as a reference (see Fig. 8). Since $mv^{n-2}_{(1,1)}$ is zero, we have

$$\hat{mv}^{n-2}_{(1,1)} = mv^{n-3}_{(1,1)}. \tag{6}$$

One straightforward approach for computing $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$ is to decode $\mathrm{MB}^{n-3}_{(1,1)}$ in the pixel domain, and the decoded macroblock is re-encoded by using frame $n-4$ as a reference and can be written as

$$\begin{aligned} Q\left[\mathrm{DCT}\left(\hat{e}^{n-2}_{(1,1)}\right)\right] &= Q\left[\mathrm{DCT}\left(\mathrm{MB}^{n-2}_{(1,1)} - \mathrm{MCMB}^{n-4}\left(\hat{mv}^{n-2}_{(1,1)}\right)\right)\right] \\ &= Q\left[\mathrm{DCT}\left(\mathrm{MB}^{n-2}_{(1,1)} - \mathrm{MCMB}^{n-4}\left(mv^{n-3}_{(1,1)}\right)\right)\right]. \end{aligned} \tag{7}$$

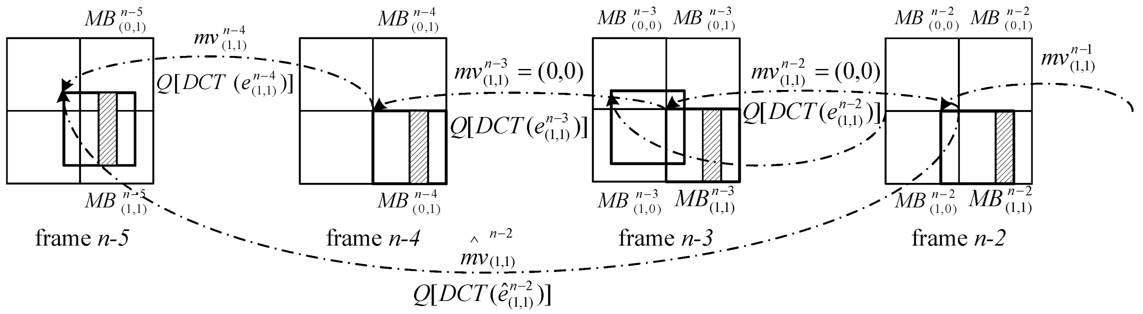

$MB^{n-5}_{(0,1)}$
$mv^{n-4}_{(1,1)}$
$Q[DCT\,(e^{n-4}_{(1,1)})]$
$MB^{n-4}_{(0,1)}$
$mv^{n-3}_{(1,1)}=(0,0)$
$Q[DCT(e^{n-3}_{(1,1)})]$
$MB^{n-3}_{(0,0)}$ $MB^{n-3}_{(0,1)}$
$mv^{n-2}_{(1,1)}=(0,0)$
$Q[DCT(e^{n-2}_{(1,1)})]$
$MB^{n-2}_{(0,0)}$ $MB^{n-2}_{(0,1)}$
$mv^{n-1}_{(1,1)}$
$MB^{n-5}_{(1,1)}$
$MB^{n-4}_{(0,1)}$
$MB^{n-3}_{(1,0)}$ $MB^{n-3}_{(1,1)}$
$MB^{n-2}_{(1,0)}$ $MB^{n-2}_{(1,1)}$
frame *n-5*
frame *n-4*
frame *n-3*
frame *n-2*
$\hat{mv}^{n-2}_{(1,1)}$
$Q[DCT(\hat{e}^{n-2}_{(1,1)})]$

Fig. 9. Using direct addition of DCT coefficients iteratively.

The decoding and re-encoding processes can create undesirable complexity on the server and also the video quality of the pixel-domain approach suffers from its intrinsic double-encoding process, which introduces additional degradation [21]. In the proposed video server, we employ a DCT-domain technique to compute the new $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$. Using (1) $\mathrm{MB}^{n-2}_{(1,1)}$ can be written as

$$\mathrm{MB}^{n-2}_{(1,1)} = \mathrm{MCMB}^{n-3}\left(mv^{n-2}_{(1,1)}\right) + e^{n-2}_{(1,1)}. \tag{8}$$

If $\mathrm{MB}^{n-2}_{(1,1)}$ is coded without motion compensation, $mv^{n-2}_{(1,1)}$ is zero and $\mathrm{MCMB}^{n-3}(mv^{n-2}_{(1,1)})$ is equal to $\mathrm{MB}^{n-3}_{(1,1)}$. We can simplify (8) into

$$\mathrm{MB}^{n-2}_{(1,1)} = \mathrm{MB}^{n-3}_{(1,1)} + e^{n-2}_{(1,1)}. \tag{9}$$

Similarly

$$\mathrm{MB}^{n-3}_{(1,1)} = \mathrm{MCMB}^{n-4}\left(mv^{n-3}_{(1,1)}\right) + e^{n-3}_{(1,1)}. \tag{10}$$

Substituting (10) into (9), we obtain

$$\mathrm{MB}^{n-2}_{(1,1)} - \mathrm{MCMB}^{n-4}\left(mv^{n-3}_{(1,1)}\right) = e^{n-2}_{(1,1)} + e^{n-3}_{(1,1)}. \tag{11}$$

Using (8) and (11), the newly quantized DCT coefficients of the prediction error between the current non-MC macroblock and its corresponding reference macroblock in frame $n-4$, $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$, can be written as

$$Q\left[\mathrm{DCT}\left(\hat{e}^{n-2}_{(1,1)}\right)\right] = Q\left[\mathrm{DCT}\left(e^{n-2}_{(1,1)} + e^{n-3}_{(1,1)}\right)\right]. \tag{12}$$

Taking into account the linearity of DCT, (12) becomes

$$Q\left[\mathrm{DCT}\left(\hat{e}^{n-2}_{(1,1)}\right)\right] = Q\left[\mathrm{DCT}\left(e^{n-2}_{(1,1)}\right) + \mathrm{DCT}\left(e^{n-3}_{(1,1)}\right)\right]. \tag{13}$$

Note that, in general, quantization is not a linear operation because of the integer truncation. However, $\mathrm{DCT}(e^{n-2}_{(1,1)})$ and $\mathrm{DCT}(e^{n-3}_{(1,1)})$ are divisible by the quantizer step-size. Thus, we obtain the final expression of $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$ by using frame $n-4$ as a reference

$$Q\left[\mathrm{DCT}\left(\hat{e}^{n-2}_{(1,1)}\right)\right] = Q\left[\mathrm{DCT}\left(e^{n-2}_{(1,1)}\right)\right] + Q\left[\mathrm{DCT}\left(e^{n-3}_{(1,1)}\right)\right]. \tag{14}$$

Equation (14) implies that the newly quantized DCT coefficient $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$ can be computed in the quantized DCT domain by adding $Q[\mathrm{DCT}(e^{n-2}_{(1,1)})]$ and $Q[\mathrm{DCT}(e^{n-3}_{(1,1)})]$. Both of them can be directly extracted from the MPEG video stream in the server. Since it is not necessary to perform decoding and re-encoding, the computational complexity required for the server is limited. Instead of transmitting and decoding $Q[\mathrm{DCT}(e^{n-2}_{(1,1)})]$ and $Q[\mathrm{DCT}(e^{n-3}_{(1,1)})]$ for $\mathrm{MB}^{n-2}_{(1,1)}$, by using the direct addition of DCT coefficients, the server only needs to send $Q[\mathrm{DCT}(\hat{e}^{n-2}_{(1,1)})]$ and only one macroblock is required to be decoded in the client machine. The DCT-domain technique can, thus, minimize the computational complexity of the decoder. Furthermore, this DCT-domain technique can be computed iteratively if $mv^{n-3}_{(1,1)}$ is also equal to zero and pixels in $\mathrm{MB}^{n-4}_{(1,1)}$ are used as reference for $\mathrm{MB}^{n-3}_{(1,1)}$ only, as depicted in Fig. 9.

We will now examine the switch positions of the proposed video streaming system when FBMBs are empolyed for the backward-play operation. The internal switch $SW_{\mathrm{VLC/DCT}}$ is employed to facilitate the use of the mixed VLC/DCT technique for FBMBs, as depicted in Fig. 3(b). Owing to the adoption of the VLC-domain technique, for processing FBMB-FRs, switch $SW_{\mathrm{VLC/DCT}}$ is connected to C_{VLC} and the switch positions of SW_2, SW_3, and SW_4 are the same as that of BMBs, as shown in Table IV, so that all reconstructed pixels of FBMB-BRs are stored in the backward-FB. To reconstruct the FBMB-FR in frame $n-1$, switches SW_1 and $SW_{\mathrm{VLC/DCT}}$ in the server is connected to C_1 and C_{DCT}, respectively, when the DCT-domain technique is adopted. The macroblock selector then extracts all the related macroblocks from the previous nearest I-frame to frame $n-2$. These macroblocks may be manipulated by the DCT-domain technique if non-MC macroblocks exist and the combined macroblocks will be sent to the client machine. In the client machine, all the switches are open and the decoder decodes the necessary macroblocks in the forward order from the previous nearest I-frame to frame $n-2$. All the decoded pixels in frame $n-2$, which are referred by FBMB-FRs in frame $n-1$, are stored in FB. Afterward, switches SW_3 and SW_4 are connected to B_3 and A_4 respectively. The switch positions of the proposed video streaming system are summarized in Table I. During decoding FBMB-FRs in frame $n-1$, the prediction error between each FBMB and its corresponding motion-compensated macroblock is decoded. Each reconstructed pixel in the FBMB-FR can be obtained by adding its prediction error to its motion-compensated pixels of frame $n-2$ in FB, as shown in Fig. 3. Meanwhile, pixels of BMBs and FBMB-BRs stored in backward-FB are then composed with the reconstructed pixels of FBMB-FRs to form frame $n-1$, which is the desired frame to be displayed in the

TABLE V
DETAILED COMPARISONS AMONG $\mathrm{BMB}_{\mathrm{VLC}}$, $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$, AND THE CONVENTIONAL SYSTEM

Sequences	Bitrate	Average no. of macroblocks to be decoded by the decoder			Average no. of bits to be sent over the network		
		Conventional System	*BMB*BMB_{VLC}	BMB_{VLC} $+FBMB_{VLC/DCT}$	Conventional System	BMB_{VLC}	BMB_{VLC} $+FBMB_{VLC/DCT}$
Salesman	1.5M	3168	1900	1302	380050	220937	157910
(*352×288*)	3M	3168	1900	1302	810431	469194	323300
Football	1.5M	2640	2508	1853	376133	305439	283293
(*352×240*)	3M	2640	2508	1853	832146	666120	610162
Table Tennis	1.5M	2640	1670	1369	374415	276566	248482
(*352×240*)	3M	2640	1670	1369	759569	552555	490197
Foreman	64K	792	593	450	9700	6924	5840
(*176×144*)	128K	792	593	450	26569	19110	15450
Carphone	64K	792	563	370	8500	5811	4659
(*176×144*)	128K	792	563	370	25398	18172	13548
Claire	64K	792	226	118	10661	1891	1434
(*176×144*)	128K	792	226	118	31844	8149	5288
Grandma	64K	792	319	191	12668	2990	2037
(*176×144*)	128K	792	319	191	30982	7875	4724

TABLE VI
PERFORMANCE IMPROVEMENTS OF $\mathrm{BMB}_{\mathrm{VLC}}$ AND $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$, OVER THE CONVENTIONAL SYSTEM IN TERMS OF THE NUMBER OF MACROBLOCKS TO BE DECODED BY THE DECODER

Sequences	Bitrate	Saving of macroblocks to be decoded by the decoder	
		BMB_{VLC}	$BMB_{VLC}+ FBMB_{VLC/DCT}$
Salesman	1.5M	40.02%	58.88%
(*352×288*)	3M	40.02%	58.88%
Football	1.5M	22.02%	29.80%
(*352×240*)	3M	22.02%	29.80%
Table Tennis	1.5M	36.74%	48.16%
(*352×240*)	3M	36.74%	48.16%
Foreman	64K	25.13%	43.23%
(*176×144*)	128K	25.13%	43.23%
Carphone	64K	28.83%	53.27%
(*176×144*)	128K	28.83%	53.27%
Claire	64K	71.44%	85.07%
(*176×144*)	128K	71.44%	85.07%
Grandma	64K	59.69%	75.82%
(*176×144*)	128K	59.69%	75.82%

TABLE VII
PERFORMANCE IMPROVEMENTS OF $\mathrm{BMB}_{\mathrm{VLC}}$ AND $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$, OVER THE CONVENTIONAL SYSTEM IN TERMS OF THE NUMBER OF BITS TO BE SENT OVER THE NETWORK

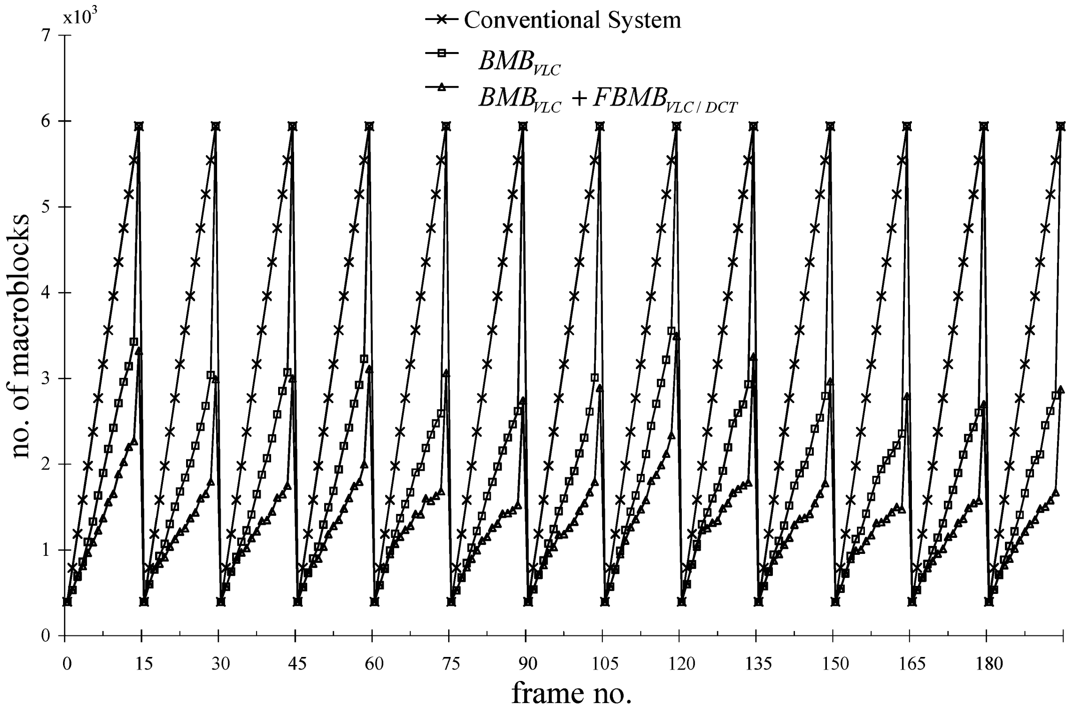
Sequences	Bitrate	Saving of bits to be sent over the network	
		BMB_{VLC}	$BMB_{VLC}+ FBMB_{VLC/DCT}$
Salesman	1.5M	41.87%	58.45%
(*352×288*)	3M	42.11%	60.11%
Football	1.5M	18.80%	24.68%
(*352×240*)	3M	19.95%	26.68%
Table Tennis	1.5M	26.13%	33.63%
(*352×240*)	3M	27.25%	35.46%
Foreman	64K	28.61%	39.80%
(*176×144*)	128K	28.07%	41.85%
Carphone	64K	31.63%	45.18%
(*176×144*)	128K	28.45%	46.66%
Claire	64K	82.26%	85.64%
(*176×144*)	128K	74.41%	83.39%
Grandma	64K	76.39%	83.91%
(*176×144*)	128K	74.58%	84.57%

backward-play operation. Frame $n-1$ is then stored in both the display buffer and the frame buffer (FB). On the other hand, frame $n-1$ stored in FB can be further used for reconstructing BMBs and FBMB-BRs of the consequent frame, frame $n-2$, in the backward-play operation.

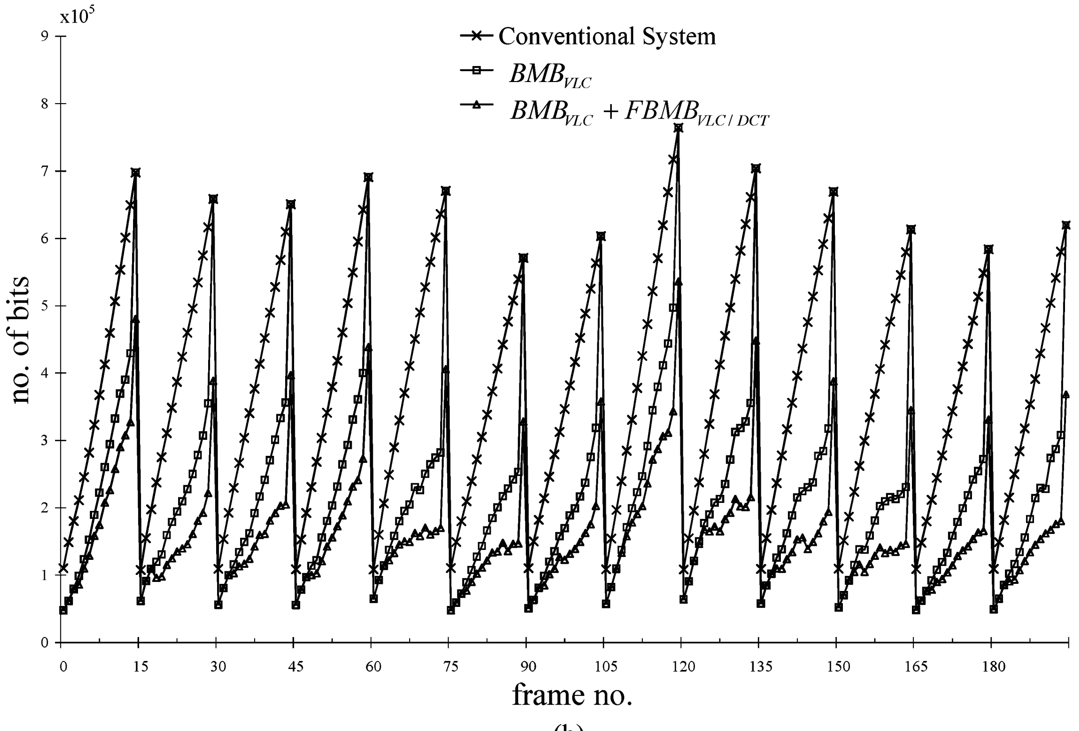
V. SIMULATION RESULTS

Extensive computer simulations have been conducted to evaluate the performances of the proposed techniques including the VLC-domain technique [15], [16] for BMBs and FBMB-BRs, and the mixed VLC/DCT-domain technique for FBMB-FRs when applied to the video streaming system with VCR support. MPEG-2 encoder [17] was employed to encode various video sequences with different spatial resolutions and motion characteristics. All the test sequences have a length of 195 frames. "Claire," "Grandma," and "Carphone" are typical videophone sequences in QCIF (176 $\times$ 144 pixels) format, which were encoded at different bitrates (64 Kb/s and 128 Kb/s). "Salesman," "Table Tennis" and "Football" in either CIF (352 $\times$ 288 pixels) format or SIF(352 $\times$ 240 pixels) format were encoded at 1.5 Mb/s and 3.0 Mb/s. For all testing sequences, the frame-rate of the video stream was 30 frames/s.

We have simulated the situation of the I-P structure with $L = 15$. The starting point of the backward-play operation is at the end of the sequence. Results of the simulations are used to compare the performance of the conventional video streaming system. Different techniques have been applied to our proposed streaming system, and let us call them



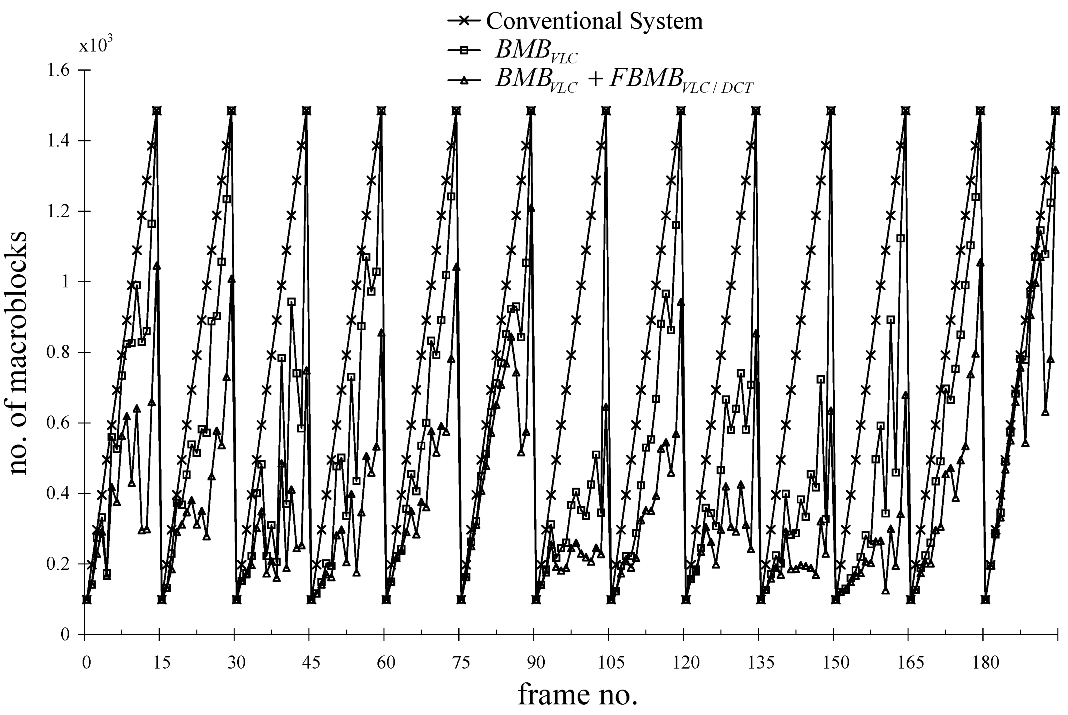
(a)



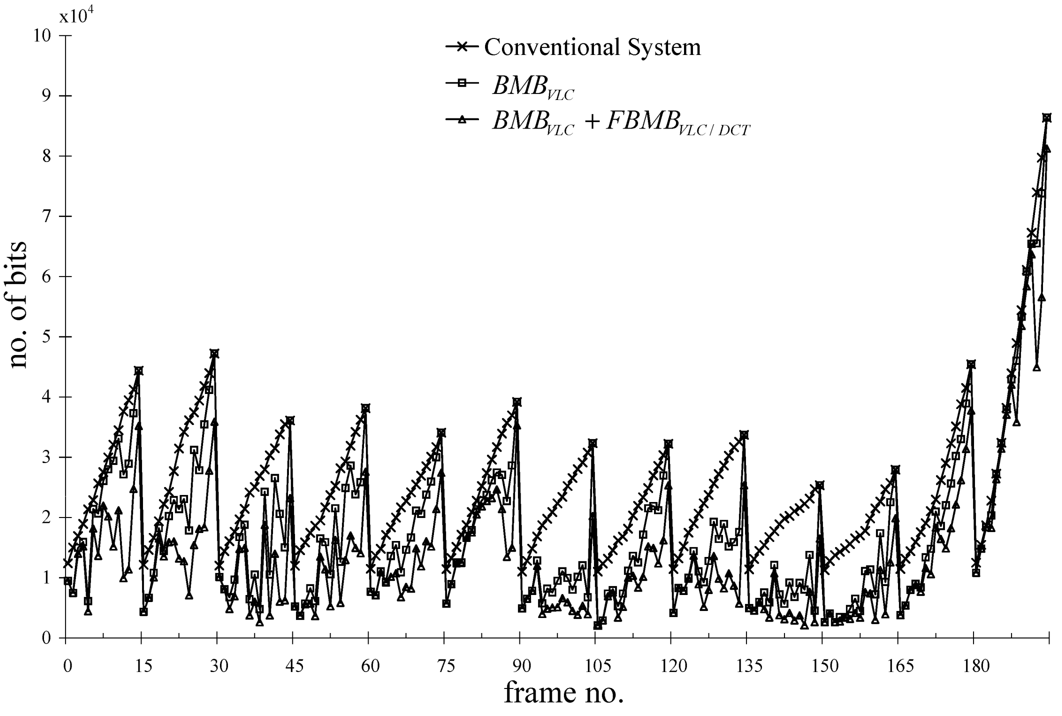
(b)

Fig. 10. Performance of the conventional system and the proposed systems, $\mathrm{BMB_{VLC}}$ and $\mathrm{BMB_{VLC}} + \mathrm{FBMB_{VLC/DCT}}$, for the "Salesman" sequence encoded at 3.0 Mb/s in the backward-play operation. (a) Number of macroblocks to be decoded by the decoder, and (b) number of bits to be sent over the network.

$\mathrm{BMB_{VLC}}$ and $\mathrm{BMB_{VLC}} + \mathrm{FBMB_{VLC/DCT}}$. $\mathrm{BMB_{VLC}}$ uses the VLC-domain technique for BMBs [15], [16]. For $\mathrm{BMB_{VLC}} + \mathrm{FBMB_{VLC/DCT}}$, we further identify two regions in each FBMB and use the VLC-domain technique for the FBMB-BR in order to achieve redundancy reduction for the FBMB-FR. Besides, $\mathrm{BMB_{VLC}} + \mathrm{FBMB_{VLC/DCT}}$ adopts the DCT-domain technique to manipulate the FBMB-FR. Note that, both $\mathrm{BMB_{VLC}}$ and $\mathrm{BMB_{VLC}} + \mathrm{FBMB_{VLC/DCT}}$ retain the same reconstruction quality as that of the conventional system since re-encoding is not necessary in the proposed techniques. The detailed comparisons of the average number of macroblocks to be decoded and bits to be sent are tabulated in Table V. The average number of macroblocks sent for decoding is directly proportional to the decoder complexity. In



(a)



(b)

Fig. 11. Performance of the conventional system and the proposed systems, $\mathrm{BMB}_{\mathrm{VLC}}$ and $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$, for the "Carphone" sequence encoded at 128 Kb/s in the backward-play operation. (a) Number of macroblocks to be decoded by the decoder, and (b) number of bits to be sent over the network.

Table V, we show that $\mathrm{BMB}_{\mathrm{VLC}}$ outperforms the conventional system in all sequences. The results are more noticeable for the sequences "Claire," "Grandma," and "Salesman" as shown in Tables VI and VII. The savings in terms of number of bits to be sent over the network and macroblocks to be decoded by the decoder is between 40%–75% for these sequences. It is due to the reason that these sequences contain more BMBs in which the technique of sign inversion can be employed. For sequences containing high motion activities such as "Football," "Table Tennis," "Foreman," and "Carphone," there are still good savings in the range of about 20%–40%. To further reduce the number of macroblocks to be decoded and bits to be sent, $\mathrm{BMB}_{\mathrm{VLC}}$ can work with the mixed DCT/VLC-domain technique for FBMBs, $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$.

Tables V–VII show that, by applying the VLC-domain and DCT-domain techniques to FBMB-BRs and FBMB-FRs, respectively, $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ produces further savings in the range of both the number of macroblocks to be decoded and bits to be sent as compared with that of $\mathrm{BMB}_{\mathrm{VLC}}$. Note that the number of macroblocks requested by the decoder is kept constant at different bitrates, as shown in Table VI, since the number of macroblocks to be decoded only depends on the numbers of BMBs and non-MC FBMBs in the encoded sequence during backward playback. In fact, the types of macroblocks are computed based on the distribution of motion vectors in the encoded sequence, which do not vary at different encoded bitrates. On the other hand, it is significant to note that the number of bits to be sent over the network can be reduced to a certain extent for sequences encoded at high bitrate, as shown in Table VII. The reason is that, at low bitrate, a considerable percentage of DCT blocks have a significant amount of zero elements. For direct addition of the DCT coefficients, all new quantized DCT coefficients are obtained in the DCT domain by adding two DCT coefficients, which are directly extracted from the MPEG video stream in the server. If either one of the DCT coefficients is zero, it does not save the bits required to encode the combined DCT coefficients. Although the direction addition of DCT coefficients can combine several macroblocks into one and save the number of macroblocks to be sent, it cannot achieve as much saving of bits as sending over the network at low bitrate.

Figs. 10 and 11 show the frame-by-frame comparisons of the required number of macroblocks decoded by the decoder and bits transmitted over the network of the conventional approach, $\mathrm{BMB}_{\mathrm{VLC}}$ and $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ for the "Salesman" and "Carphone" sequences in the backward-play operation respectively. It is obvious that the proposed techniques can achieve significant performance improvements in terms of the decoder complexity and network traffic. In Figs. 10 and 11, they show that the required number of macroblocks and bits at each last frame of GOPs of the conventional approach and $\mathrm{BMB}_{\mathrm{VLC}}$ are the same since there is no interframe dependency between the last frame of the current GOP and the first frame of the next GOP, which is an I-frame. In this case, no BMB exists in the last frame of the current GOP. Thus, the VLC-domain technique cannot be applied in the last frame of each GOP. However, $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ can get some savings of the last frame of the GOP in both the bits to be sent and the macroblocks to be decoded due to the contribution of the DCT-domain technique. For this frame, there is no BMB, all the macroblocks are treated as FBMBs. When a non-MC macroblock exists for reconstructing FBMB, the technique of direct addition of DCT coefficients can combine several non-MC macroblocks into one. The savings can then be achieved. This is another improvement of $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ over $\mathrm{BMB}_{\mathrm{VLC}}$ as shown in Figs. 10 and 11.

We have also demonstrated the performance of the proposed $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ as L, the length of the group, is varied. Tables VIII and IX show the performance improvements of the average number of macroblocks to be decoded and the average number of bits to be sent with different L respectively. For the conventional approach, if L is large, the average number of frames need to be transmitted and decoded for a requested frame in the backward-play operation is increased, which induces significant increase of decoding complexity and network traffic. Tables VIII and IX also show that $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ has a better improvement in both decoder complexity and network traffic for large L. These further demonstrate the effect of the proposed techniques when applied to the video streaming system with VCR functionality.

TABLE VIII
SAVING OF THE AVERAGE NUMBER OF MACROBLOCKS THAT NEED TO BE DECODED OF $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ AS COMPARED WITH THE CONVENTIONAL SYSTEM FOR DIFFERENT L

Sequences	Bitrate	Saving of macroblocks to be sent over the network		
		L=7	*L=15*	*L=30*
Salesman	1.5M	43.16%	58.88%	68.30%
Football	1.5M	21.64%	29.80%	42.21%
Table Tennis	1.5M	37.21%	48.16%	57.33%
Foreman	64K	29.55%	43.23%	51.94%
Carphone	64K	38.44%	53.27%	66.67%
Claire	64K	69.03%	85.07%	92.97%
Grandma	64K	60.59%	75.82%	84.37%

TABLE IX
SAVING OF THE AVERAGE NUMBER OF BITS THAT NEED TO BE SENT OF $\mathrm{BMB}_{\mathrm{VLC}} + \mathrm{FBMB}_{\mathrm{VLC/DCT}}$ AS COMPARED WITH THE CONVENTIONAL SYSTEM FOR DIFFERENT L

Sequences	Bitrate	Saving of bits to be sent over the network		
		L=7	*L=15*	*L=30*
Salesman	1.5M	48.18%	58.45%	66.52%
Football	1.5M	20.27%	24.68%	34.72%
Table Tennis	1.5M	32.69%	33.63%	37.98%
Foreman	64K	34.83%	39.80%	44.66%
Carphone	64K	41.45%	45.18%	50.96%
Claire	64K	79.78%	85.64%	90.99%
Grandma	64K	76.34%	83.91%	89.69%

VI. CONCLUSION

In this paper, we have proposed a new architecture for implementing efficient backward-playback for the MPEG video streaming system with VCR support. The proposed techniques utilize the property of center-biased motion vectors in real-world video sequences. With the motion information, the video streaming server divides the macroblocks in the requested frame into two different types—a BMB and a FBMB. Then it processes them either in the VLC domain or DCT domain, and sends the processed macroblocks to the client machine. For BMBs, we have adopted a technique of sign inversion of DCT coefficients, which is operated in the VLC domain, to simplify the decoder complexity while maintaining low network bandwidth requirement. For FBMBs, we have also shown that some pixels can still be handled in the VLC domain while the remaining pixels can be manipulated efficiently by using direction addition of DCT coefficients, which is a DCT-domain technique. This DCT-domain technique can further alleviate the computational burden of the client decoder during backward playback. Since BMBs and FBMBs are performed in the VLC domain and DCT domain only, it is not necessary to do decoding and re-encoding of the video streams in the server. This reduces the computational complexity required for the server significantly. Furthermore, since the process of re-encoding is not required, the visual quality during backward playback will be exactly the same as that of forward playback. All the points related to our proposed scheme have been verified experimentally. Our MPEG video streaming system provides remarkable backward-play quality and is able to minimize the required network bandwidth and decoder complexity significantly.

Besides, the proposed techniques are not restricted to backward playback, they can also be beneficial to other VCR operations when the GOP size is large. For instance, if the next requested frame in random-access is in the same GOP and is far away from the previous I-frame, frames between the requested frame and the current displayed frame could be decoded in reverse order by using the proposed techniques in order to save network traffic and decoder complexity. This further shows that the results of our work will certainly be useful for the future development of digital VCR.

REFERENCES

[1] T. Stockhammer, H. Jenkac, and G. Kuhn, "Streaming video over variable bit-rate wireless channels," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 268–277, Apr. 2004.

[2] D. Deloddere, W. Verbiest, and H. Verhille, "Interactive video-on-demand," *IEEE Commun. Mag.*, vol. 32, no. 5, pp. 82–88, May 1994.

[3] R. V. Cox, B. G. Haskell, Y. LeCun, B. Shahrary, and L. Rabiner, "On the applications of multimedia processing to communications," *Proc. IEEE*, vol. 86, no. 5, pp. 755–824, May 1998.

[4] D. Wu, Y. T. Thomas, and Y.-Q. Zhang, "Transporting real-time video over the internet: Challenges and approaches," *Proc. IEEE*, vol. 88, no. 12, pp. 1855–1877, Dec. 2000.

[5] Real Networks RealPlayer [Online]. Available: http://www.real.com/

[6] Microsoft Window Media Microsoft Corporation Inc. [Online]. Available: http://www.microsoft.com/windows/windowsmedia/

[7] Video Coding for Low Bitrate Communication ITU-T Recommendation H.263, May 1997.

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

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

[10] *Information Technology-Coding of Audio-Visual Objects-Part 2: Video*, ISO/IEC 14496-2, 2004.

[11] 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*, 1995, pp. 73–80.

[12] S. J. Wee, "Reversing motion vector fields," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 1998, pp. 209–212.

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

[14] C. W. Lin, J. Zhou, J. Youn, and M. T. Sun, "MPEG video streaming with VCR functionality," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 415–425, Mar. 2001.

[15] C. H. Fu, Y. L. Chan, and W. C. Siu, "Improved macroblock-based reverse play algorithm for MPEG video streaming," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 24–27, 2004, pp. 2063–2066.

[16] C. H. Fu, Y. L. Chan, and W. C. Siu, "Macroblock-based reverse play algorithm for MPEG video streaming," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 23–26, 2004, vol. 3, pp. 753–756.

[17] *Video Codec Test Model, TMN8*, ITU-T/SG15, Jun. 1997.

[18] C. H. Cheung and L. M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1168–1177, Dec. 2002.

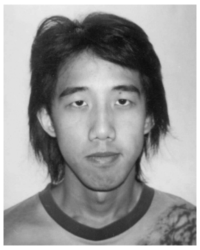
[19] L. J. Luo, C. R. Zou, X. Q. Gao, and Z. Y. He, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. Consum. Electron.*, vol. 43, no. 1, pp. 56–61, Feb. 1997.

[20] V. Christopoulos and J. Cornelis, "A center-biased adaptive search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 4, pp. 423–426, Apr. 2000.

[21] K. T. Fung, Y. L. Chan, and W. C. Siu, "Low-complexity and high quality frame-skipping transcoder," in *Proc.IEEE Int. Symp. Circuits and Systems*, Sydney, Australia, May 6–9, 2001, pp. 29–32.

[22] K. T. Fung, Y. L. Chan, and W. C. Siu, "New architecture for dynamic frame-skipping transcoder," *IEEE Trans. Image Process.*, vol. 11, no. 8, pp. 886–900, Aug. 2002.

[23] K. T. Fung, Y. L. Chan, and W. C. Siu, "Low-complexity and high-quality frame-skipping transcoder for continuous presence multipoint video conferencing," *IEEE Trans. Multimedia*, vol. 6, no. 1, pp. 31–46, Feb. 2004.



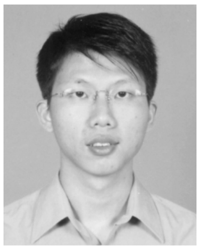
Chang-Hong Fu received the B.Eng. degree with first class honors from the Hong Kong Polytechnic University in 2002, where he is currently pursuing the Ph.D. degree.

His research interests include multimedia technologies, signal processing, and video coding.

Yui-Lam Chan (A'94–M'00) received the B.Eng. degree with first class honors and the Ph.D. degree from the Hong Kong Polytechnic University in 1993 and 1997, respectively.

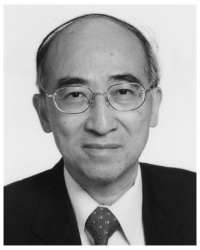
He joined the Hong Kong Polytechnic University in 1997, and is now an Assistant Professor in the Department of Electronic and Information Engineering. He has published over 40 research papers in various international journals and conferences. His research interests include multimedia technologies, signal processing, image and video compression, video streaming, video transcoding, video conferencing, digital TV, error-resilient coding, and digital VCR.

Dr. Chan has been the recipient of more than ten famous prizes, scholarships, and fellowships for his outstanding academic achievement, such as being the Champion in Varsity Competition in Electronic Design, the Sir Edward Youde Memorial Fellowship, and the Croucher Foundation Scholarships. He received the Faculty Merit Award in Teaching (Team) in 2005 and the Faculty of Engineering Research Grant Achievement Award in 2005. He has also been actively involved in professional activities. In particular, he has been a reviewer, session chairman, and organizing/technical committee member of many international conferences. He was the Registration Chair and Session Chair of the 2004 International Symposium on Intelligent Multimedia, Video, and Speech Processing (ISIMP 2004) held in Hong Kong October 20–22, 2004.



Tak-Piu Ip received the B.Sc. (Hons.) degree from the Hong Kong Polytechnic University in 2003, where he is currently pursuing the M.Phil. degree.

His research interests include image and video technology, video compression, and digital VCR.



Wan-Chi Siu (S'77–M'77–SM'90) received the Associateship from The Hong Kong Polytechnic University in 1975, the M.Phil. degree from The Chinese University of Hong Kong in 1977, and the Ph.D. Degree from the Imperial College of Science, Technology, and Medicine, London, U.K., in October 1984.

He was with The Chinese University of Hong Kong as a Tutor and later as an Engineer between 1975 and 1980. He then joined The Hong Kong Polytechnic University as a Lecturer in 1980. He was promoted to Senior Lecturer, Principle Lecturer, and Reader in 1985, 1987, and 1990, respectively, and has been Chair Professor of the Department of Electronic and Information Engineering since 1992. He was Head of Department of Electronic and Information Engineering Department and subsequently Dean of Engineering Faculty between 1994 and 2002, and he is currently the Director of the Centre for Multimedia Signal Processing. He has published 300 research papers, over 130 of which appeared in international journals, and is an Editor of the recent book *Multimedia Information Retrieval and Management* (Springer, 2003). His research interests include digital signal processing, fast computational algorithms, transforms, wavelets, image and video coding, and computational aspects of pattern recognition.

Prof. Siu was a Guest Editor, Associate Editor, and Editorial Board Member of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: PATTERN RECOGNITION, the *Journal of VLSI Signal Processing Systems for Signal, Image, Video Technology,* the *EURASIP Journal on Applied Signal Processing*, in addition to other journals. He has been a keynote speaker at many international conferences, including the IEEE PCM 2002 (Taiwan, R.O.C.) and the IEEE 2003 ICNNSP (Nanjing, China). He has held the position of General Chair or Technical Program Chair of many international conferences, including IEEE Society-sponsored flagship conferences such as ISCAS 1997, ICASSP 2003, and ICIP 2010 (to be held in Hong Kong). Between 1991 and 1995, he was a member of the Physical Sciences and Engineering Panel of the Research Grants Council (RGC), Hong Kong Government, and in 1994, he chaired the first Engineering and Information Technology Panel of the Research Assessment Exercise (RAE) to assess the research quality of 19 Cost Centers (departments) from all universities in Hong Kong. He has received many awards, including Distinguished Presenter Award (1997), IEEE Third Millennium Medal (2000), the Best Teacher Award (2003), the Outstanding Award in Research (2003), the Plaque for Exceptional Leadership from IEEE SPCB (2003), and Honorable Mention Winner Award from Pattern Recognition (2004).