The Hong Kong Polytechnic University

Department of Electronic and
Information Engineering

Digital Video Browsing using Efficient Bitstream Switching Techniques

by IP Tak-Piu

A thesis submitted in partial fulfillment of the
requirements for the Master of Philosophy

November 2007

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written nor material which has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

IP Tak Piu (Name of student)

Abstract

Recent advances in networking and multimedia technologies open the possibilities for network/service/content providers to offer residential customers with video-on-demand services. Interactive browsing operations such as random-access and fast-forward/backward playback are desirable features in these services. However, many video coding standards use motion-compensated prediction to reduce temporal redundancy of video sequences. Although it is efficient, this predictive scheme imposes extra constraints on how a compressed video displays since a predicted frame cannot be restored before any of its reference frames. It means that a compressed video should be played back in a pre-determined frame order. Displaying the digital video in other orders always requires extra resources for both network traffic and decoder complexity.

Recently, a dual-bitstream technique has been suggested storing an additional reverse-encoded bitstream in the server to facilitate video browsing. Once the client requests a Video Cassette Recorder (VCR) operation, the server will select an appropriate frame for the client from either the forward or reverse-encoded bitstreams by considering the decoding effort at the decoder

and the traffic over the networks. Therefore, bitstream switching is necessary for selecting appropriate frames in these two bitstreams. However, switching between these bitstreams is not a straightforward task. Since a P-frame is encoded using the prediction from the previously reconstructed reference, switching between the bitstreams at a P-frame would lead to drift errors due to the mismatch of the reconstructed references at that frame. Such errors will be propagated to subsequent P-frames.

In this thesis, we investigate the impact of bitstream switching between the forward and reverse-encoded bitstreams. Two approaches based on both frame level and macroblock level are then proposed to solve the drift problem. In the proposed frame-level approach, we modify the original dual-bitstream structure by adopting SI/SP-frames to eliminate drift errors and some efficient algorithms are investigated for implementing the required components and addressing this challenging issue. Experimental results show that, as compared to the original dual-bitstream structure, this new approach enhances the quality of the reconstructed video significantly.

For the proposed macroblock-level approach, a video server classifies macroblocks in the requested frame into two categories – a reference-mismatched macroblock (RMMB) and a non-reference-mismatched

macroblock (non-RMMB). A novel technique is used to manipulate the necessary macroblocks in the compressed domain and then the server sends the processed macroblocks to the client machine. For non-RMMBs, we propose a sign inversion technique in the Variable Length Coding (VLC) domain to eliminate the drift errors at certain areas that have static or slow motion activities. Besides, a simple version for low-cost video servers, which makes use of the redundancy inherent between the forward and reverse-encoded bitstreams in order to achieve a substantial reduction on the size of the reverse-encoded bitstream, is also proposed.

It is exciting to report in this thesis that significant improvements in terms of the server complexity, the storage requirement of a server, and the browsing quality of reconstructed video can be achieved by employing our frame-level and macroblock-level approaches. Undoubtedly, these techniques are able to effectively facilitate video browsing in the dual-bitstream system.

Acknowledgment

I am deeply indebted to my supervisor, Dr. Chan Yui-Lam, for his guidance, support and encouragement during my years at The Hong Kong Polytechnic University. Without his appreciation of my work, it is impossible for me to complete this research study. I would like to express my sincere thank to my co-supervisor, Professor Siu Wan-Chi, for his valuable advice and suggestion contributed to paper and thesis writing.

I would also like to thank my colleagues at the Hong Kong Polytechnic University for their friendship and support, in particular, Mr. Alvin Cheung Hoi Kin and Mr. Rainbow Fu Changhong. Their expert knowledge and friendly encouragement have helped me resolved many difficult problems in my study.

Meanwhile, I also thank to the Department of Electronic and Information Engineering and the Centre of Multimedia Signal Processing for providing me a supportive workplace, and the grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (PolyU 5204/04E) for their generous financial support.

I am also grateful to the reviewers whose feedback helped us to improve the presentation of the papers.

Table of Contents

List of Figures

List of Tables

Abbreviations

3G	Third Generation Mobile Wireless Network
B-frame	Bi-directional Predicted Frame
DC	Drift Compensation
DCT	Discrete Cosine Transform
DRB	Direct Reference Bitstream
DVD	Digital Versatile Disc
FB	Forward-encoded Bitstream
FLC	Fixed Length Coding
FPS	Frames per Second
GOP	Group of Pictures
I-frame	Intra-coded Frame
JPEG	Joint Picture Experts Group
MB	Macroblock
MC	Motion Compensation
MCMB	Motion-Compensated Macroblock
MCP	Motion-Compensated Prediction
ME	Motion Estimation

M-JPEG	Motion JPEG
MP3	MPEG Layer-3 Audio Coding Standard
MPEG	Moving Picture Experts Group
MV	Motion Vector
P-frame	Forward Predicted Frame
QoS	Quality-of-Service
RB	Reverse-encoded Bitstream
RMMB	Reference-Mismatched Macroblock
S-frame	Traditional Switching Frame
SI-frame	Intra-coded Switching Frame
SMB	Skipped Macroblock
SP-frame	Forward Predicted Switching Frame
SRB	Simplified Reverse-encoded Bitstream
TV	Television
VCD	Video Compact Disc
VCR	Video Cassette Recorder
VLC	Variable Length Coding
VLD	Variable Length Decoding

List of Publications

Accepted Papers

1. Tak-Piu Ip, Yui-Lam Chan, Chang-Hong Fu and Wan-Chi Siu, “A Simplified Dual-Bitstream MPEG Video Streaming System with VCR Functionalities,” Proceedings, IEEE International Conference on Image Processing (ICIP’07), Vol. 6, pp. 481-484, September 16-19, 2007, San Antonio, Texas, USA.

2. Tak-Piu Ip, Yui-Lam Chan and Wan-Chi Siu, “Adopting SP/SI-FRAMES in Dual-bitstream Video Streaming with VCR Support,” Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP’06), Vol. 2, pp. 549-552, May 15-19, 2006, Toulouse, France.

3. Tak-Piu Ip, Yui-Lam Chan, Chang-Hong Fu and Wan-Chi Siu, “Efficient Reverse-Play Algorithm for Dual-bitstream MPEG Video Streaming System,” Proceedings, IEEE International Conference on Circuits and Systems 2005 (ISCAS’05), pp. 2671-2674, May 23-26, 2005, Japan.

Submitted Papers

1. Tak-Piu Ip, Yui-Lam Chan and Wan-Chi Siu, ”New Dual-bitstream Structure for Video Streaming with VCR Functionalities,” Submitted for possible publication in IET Image Processing.

2. Tak-Piu Ip, Yui-Lam Chan and Wan-Chi Siu, “Redundancy Reduction Technique for Dual-Bitstream MPEG Video Streaming with VCR Functionalities,” submitted for possible publication in IEEE Transactions on Broadcasting.

Papers under Preparation

1. Tak-Piu Ip, Yui-Lam Chan and Wan-Chi Siu, “Browsing H.264 Video using Macroblock-Based Dual-Bitstream Structure with SP/SI-frames.”

Chapter 1 – Introduction

1.1 Overview

The advance of video streaming technologies [1-3] and video compression standards such as MPEG-1/2/4 and H.26x [4-9] are fuelling the emergence of a wide range of devices and applications that exploit the compressed-domain representation of digital video. It facilitates visual data transmission through the Internet, contributes to the advent of digital broadcast system, and makes the storage of digital video on VCDs, DVDs, and video servers for future delivery possible.

Video streaming [10-12] refers to real-time transmission of stored video. Real-time streaming always implies that the video content need not be downloaded in full, but is being played out while parts of the content are being received and decoded. It has timing constraints. That is, if the video data cannot arrive in time, the playback process will pause and it is annoying to viewers. Video streaming also has bandwidth, loss, and delay requirements. However, the current available networks do not offer these guarantees. For instance, the best-effort Internet does not provide any quality-of-service (QoS) guarantees for streaming video over the Internet. Besides, the heterogeneous

nature of the Internet further makes it complicated to efficiently support video multicast while providing service flexibility to meet a wide range of QoS requirements from viewers. Thus, the realization of a video streaming system poses many challenges such as the complexity of the encoder, QoS control of streaming video, the high storage-capacity, throughput in the video server, and the high bandwidth in the network to deliver video streams. Recent advances in computing technology, compression and network standards, high-band storage devices, and high speed-networks have made it feasible to provide video streaming applications which have emerged as one of the indispensable applications over the Internet and 3G wireless networks.

Besides this, owing to the diversity of multimedia devices and applications, the video streaming system should have the capability to serve a wide variety of clients' needs. In order to fulfill the requirements for various clients, an efficient organization of stored video provides the flexibility for structure conversion under various conditions. For example, in the Internet streaming video applications, the servers always have to serve a large amount of users with heterogeneous display resolutions and network bandwidth. When the users' display resolution is small or the bandwidth between the server and some users is not enough to support higher resolution sequences, scalable coding [13-19]

can provide different resolutions and bit rates to accommodate various users. In general, different scalable coding schemes adopted in H.263 [7], MPEG-2 [6], and MPEG-4 [8] divide a video sequence into two layers – the base and enhancement layers. The video sequence is encoded at low quality or resolution to form the base layer, while the enhancement layer is encoded by computing the difference between the original image and the reconstructed image of the base layer. Such arrangement of stored video is capable of gracefully coping with the users with heterogeneous devices or the bandwidth fluctuations of the network. In the past decade, many researchers are working on the problem of scalable coding [13-19] to handle the heterogeneity of the receiver and network.

The scalable coding schemes are very successful for video streaming with channel bandwidth variation. It allows users to ubiquitously access and retrieve various video contents over heterogeneous networks by using software players [20-21] or digital set-top box devices [22-23] with different display resolutions. But they were developed primarily for compression and transmission, but are not convenient for browsing. In order to enhance the user's viewing experience, browsing video contents interactively is also desirable in these online video streaming services.

In this chapter, the fundamental of digital video browsing and its motive are introduced. We then mention the crux of the digital video browsing problem. Afterwards, some multiple bitstream techniques for browsing digital video are also briefly discussed. Finally, the motivation, objectives and the organization of this thesis are presented.

1.2 Digital Video Browsing for Compressed Video

Video streaming services coupled with software/hardware playback devices provide great opportunities for significantly enhancing the user's viewing experience. They are currently ready to replace older analog devices and systems such as analog TV and Video Cassette Recorders (VCRs). In order to complete the transition to digital video from its current analog state, a video server needs to be able to support a variety of VCR operations such as forward, backward, stop, pause, step forward, step backward, fast forward, fast backward, and random access. Many digital video players [20-23] offer relatively few controls for video browsing nowadays. For example, these players have only limited fast-forward/backward play and even they cannot provide the backward play. The limitation is due to the use of motion-compensated prediction in various video coding standards. In the

following, we briefly discuss the problem of browsing compressed video.

Since the large amount of data required for digital video transmission and storage, the need for effective compression is evident in almost all applications where storage and transmission of digital video are involved. Compression of video data without significant degradation of the visual quality is viable because a video sequence always contains a high degree of spatial and temporal redundancy. By eliminating them, it is possible to represent the information in a more compact form. To remove spatial redundancy, a single frame in a video sequence is transformed from the spatial domain into another domain in which the information is represented more compactly. Certain components of the transformed information can then be discarded without seriously degrading the visual quality of the decoded image. Besides, scene and objects in the video content changes smoothly and gently over time, so successive frames are often similar. It implies that a moving video sequence contains temporal redundancy. In most modern video coding standards, a motion estimation and compensation technique is widely used to obtain further compression [24-29]. This technique is to reduce temporal redundancy between successive frames by forming a predicted frame and subtracting this from the current frame. The predicted frame is created from the reference frame, by estimating motion between the

current frame and the reference frame. The predicted frame can then be formed by compensating for motion between the two frames. The output of this process is a residual frame or predicted difference. The more accurate the prediction process, the less energy is contained in the residual frame. The residual frame is undergone transformation, quantization and entropy coding processes in order to further remove the spatial redundancy before it is stored or sent to the decoder. In the decoder, by using the motion information and the reference frame, the predicted frame is re-created and it adds to the decoded residual to reconstruct the current frame.

However, this motion estimation and compensation technique drastically complicates the video browsing operations. The predictive structure allows a straightforward realization of forward playback, but imposes great constraints on other VCR operations such as backward playback, fast-backward playback, fast-forward playback and random access. By taking backward playback as an example, the solution for backward playback is simple for uncompressed video; it just reorders 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 video data because the present video

compression standards use motion-compensated prediction in which the compressed data is 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.

1.3 Multiple-bitstream Techniques for Interactive Video Browsing

As mentioned before, the scalable video coding can be used to adapt channel bandwidth variation during video streaming. However, the motion prediction of the enhancement layer is always based on the lowest quality base layer. As a result, low coding efficiency is a major disadvantage that prevents scalable coding from being widely deployed in video streaming applications under a wide range of bandwidth variations. Recently, both the MPEG-4 and H.264 standards are interested in developing an alterative – multiple-bitstream coding [30-37]. For adopting multiple bitstreams in the video server, video is independently encoded into several non-scalable streams with different bitrates. By dynamically switching among these bitstreams, adaptation to channel bandwidth variation can be achieved.

The bitstream switching technique is an effective and efficient solution in

solving the bandwidth variation problem in providing video streaming services for the current Internet and forthcoming 3G wireless network. Besides, bitstream switching will play a vital role in the future market of other video applications. One possible way to provide interactive video browsing is to borrow the idea of the multiple bitstream coding approach by storing multiple video bitstreams with different temporal resolutions or coding directions.

For example, *Omoigui et al.* [41] investigated possible client-server time-compression implementations for fast-forward play and video browsing. The time-compression approach can be implemented by storing multiple pre-encoded bitstreams with different temporal resolutions. It sends a bitstream with suitable temporal resolution according to a user's request. This approach does not introduce excessive network traffic but the speed-up granularity is limited by the number of pre-stored bitstreams. Besides, *Lin et al.* [42] recently proposed to store the forward-encoded bitstream as well as the backward-encoded bitstream in the server to simplify the reverse-play complexity while maintaining the low network bandwidth requirement. Based on the dual-bitstream structure, a frame-selection scheme has been designed at the server to minimize the required network bandwidth and the decoder complexity. This scheme determines the frames to stream over the networks by

switching between the two bitstreams based on a least-cost criterion. However, this dual-bitstream system approximately doubles the storage requirement of the server.

1.4 Motivation and Objectives

Interactive browsing operations, such as fast-forward, fast-backward, backward and random access are desirable features in video-on-demand and other multimedia servers. However, owing to the motion-compensated predictive technique adopted in the current video coding standards, it is not a trivial task. Storing multiple bitstreams with different temporal resolutions or coding directions in the video server is a possible solution. By switching among these bitstreams, it facilitates various interactive browsing operations in digital video and maintains the low network bandwidth requirement.

However, we have done some careful investigation of this multiple-bitstream approach. Results of our investigation indicate that this method is still primitive, and there is plenty of room for improvement. For instance, this approach always leads to drift errors during interactive browsing operations since the frame in one bitstream may not be exactly identical to the frame in another bitstream. If one of these frames is used as the reference for

a frame in another bitstream, it induces mismatch errors. Such errors will be propagated to subsequent P-frames. One straightforward approach to accomplish drift-free switching is to do bitstream switching only at the I-frames. This solution is simple, but either incurs long delay when switching is required in browsing operations. Therefore, the objective of this thesis focuses on how to flexibly switch from one bitstream to another at any frame without causing any drifting errors which is a serious problem for the multiple-bitstream or dual-bitstream development in video browsing applications.

In order to bolster the user's viewing experience, our study is to enable the capability of browsing video contents interactively by using a dual-bitstream approach. In this thesis, we perform a detailed analysis and provide a practical solution for implementing the dual-bitstream system with high-quality video browsing capability. Several novel components are integrated into the dual-bitstream system so as to offer complete video browsing operations. We believe that our novel techniques will play a vital role in the future market of video servers, VOD, and movie entertainment industry.

1.5 Organization of the Thesis

Chapter 2 gives a broad overview of video compression techniques. The

overview covers some general introductory video coding materials for the purpose of clarifying certain definitions used in later chapters. Afterwards, a discussion addresses the impact of implementing interactive browsing operations in compressed video. At the end of this chapter, some current multiple-bitstream techniques related to video browsing are given.

In Chapter 3, we discuss the issues in implementing a dual-bitstream system with full VCR functionality in details. We argue that the dual-bitstream system suffers two problems. First, it introduces drift since the frames in one of the bitstream would be approximated by the frames in the other bitstreams during VCR operations. Second, it increases the storage requirement of the video server.

A novel frame-based approach is presented in Chapter 4. This approach proposes a new dual-bitstream structure with SI-/SP-frames in order to eliminate the drifting errors incurred from bitstream switching between the forward-encoded and reverse-encoded bitstreams. Chapter 5 then mainly investigates a macroblock-based approach for video streaming with VCR support. A compressed-domain technique is proposed to perform backward playback without introducing any drift. Since the technique proposed in this chapter is operated in the compressed domain, the increase in server's

complexity is very limited. Chapter 6 then extends this compressed-domain technique to reduce redundancy between the two bitstreams. It is found that this redundancy reduction technique is very suitable for a low-cost video streaming server with limited storage capacity.

Chapter 7 is devoted to a summary of the work herein and the conclusions reached as a result. Suggestions are also included for further research in the aspects of video browsing.

Chapter 2 – Literature Review

2.1 Introduction

With the widespread adoption of video technology in video streaming, digital television, video-on-demand, and DVD, video compression has become a crucial component of broadcast and entertainment media. The rapidly advancing video technology is continuously spawning new products and applications, and their emergence will have a significant impact on a large number of people from all walks of life. Its success is based on video coding standards [5-9] such as MPEG-1/-2/-4, H.26L and H.264. They have been developed to provide standard video formats for the convenience in storage, manipulation and transmission. Owing to the enormous amount of storage required by digital video, some compression techniques such as transform coding and motion-compensated prediction are adopted in the standards so as to reduce the required data rate while preserving its viewing quality. These techniques can exploit different kinds of redundancy inherent in the video data for achieving a substantial reduction in data rate. Meanwhile, they produce dependencies among the compressed video data. This is mainly due to various predictive coding techniques specified in the standards. These

dependencies do not cause any inconvenience in storage and normal playback of digital video, but induce difficulty when interactive browsing of video is desirable in a video system. Recently, various research works have been conducted in different ways and a number of algorithms have been proposed.

This chapter is organized as follows. First of all, we start with the brief description of the digital video representation. We then introduce the existing image and video coding standards, with an emphasis on the techniques that are related to this research. After that, we discuss the effects of some techniques used in the existing coding standards that complicate video browsing in video streaming applications with VCR operations. Afterwards, we provide the literature reviews on some existing works of video browsing using a multiple-bitstream approach. The chapter concludes with a dual-bitstream video streaming system with VCR support that will be addressed in this work.

2.2 Foundations of Digital Video Coding

Digital video is the representation of a sampled video scene in digital form. Each spatio-temporal sample or pixel describes its brightness and color using a number or set of numbers. However, current network throughput rates are

insufficient to handle uncompressed video in real time and even a DVD can only store a few seconds of uncompressed video at TV-quality resolution and frame rate. Therefore, these applications would not be practical without video compression. In this section, some techniques related to video coding are reviewed.

2.2.1 Representation of Video Information

Actually, a video sequence is a series of still frames or pictures sampled at regular intervals in time. In our human visual system, it responses slowly to rapid change of illumination [44]. For that reason, an instant displayed image persists with eye in a short period of time. If a frame is refreshed shortly, the eye may perceive a continuous video scene. Otherwise, visible fading between frames, called flicker, occurs. Flicker is caused by the previous frame fading from the eye retina before the next one is displayed. To avoid the flickering effect, the typical refresh rate of the television system is set to 25 or 30 frames per second (fps).

For each frame, it is sampled horizontally and vertically by a light sensitive device (e.g. CCD or CMOS sensor) to construct the image in digital form. The sensitive device is a silicon chip consisting of a two-dimensional grid of sensors,

each corresponding to one pixel, and the optical signal reaching each sensor is converted to an electronic signal. This signal is then sampled and quantized to discrete value through an analog to digital circuit (A/D circuit). The number of pixels per line and the number of lines per frame represent the horizontal and vertical resolution of digital video respectively. By placing pixels close together, the observer can perceive a continuous image. Thus, the more the number of pixels it has, the higher resolution the video frame as well as the higher quality it represents.

Pixel is then the most fundamental element of each frame. For grayscale pixels, the corresponding light sensitive sensors capture the luminance signal by accumulating the electronic charge corresponding to light intensity in the scene. While the higher light intensity the cell absorbed, the higher electrical charges are accumulated. Then, the discrete pixels are obtained by digitization operation. Typical pixel value is ranged from 0 to 255 intensity steps in digital video. A black pixel is represented by 0 while a white pixel is represented by 255. On the other hand, color images are represented by at least three components per pixel. Natural light sources consist of a mixture of electromagnetic waves in the spectrum of visible wavelengths from 400 to 700 nm [44]. Since the object surfaces are wavelength selective, light wave at the

wavelengths of the color can be reflected or absorbed by the materials. This causes an object to have its characteristic color. The trichromatic theory [44] states that any color can be reproduced by a combination of three primary colors. The traditional color space for computer graphics is RGB (red, green, and blue). In the RGB space, a color image sample is represented with three numbers that indicate the relative proportions of red, green, and blue. Any color can be created by combining red, green, and blue in varying proportions.

For digital video, in order to reduce the bandwidth requirement, the most typical color space used in video coding standards represents luminance (grayscale) and chrominance (color) components separately. It is due to the reason that the human visual system is less sensitive to the chrominance components than to the luminance component. However, the three colors in the RGB color space are treated equally. By separating the luminance from the chrominance information and representing luminance with a higher resolution than chrominance, a more compact color space can be used. To do so, the chrominance components can be represented with a lower spatial resolution than the luminance component.

Nowadays, the popular color space used in different video coding standards is YC_bC_r (Y is the luminance component; C_b and C_r are the two

chrominance components). The YC_bC_r signals are related to the RGB signals by:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix} \tag{2.1}$$

and

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.164 & 0 & 1.596 \\ 1.164 & -0.391 & -0.813 \\ 1.164 & 2.018 & 0 \end{pmatrix} \begin{pmatrix} Y-16 \\ Cb-128 \\ Cr-128 \end{pmatrix} \tag{2.2}$$

In fact, the YCbCr color space, as shown in Figure 2.1, was developed as a part of the CCIR 601 standard during the establishment of a worldwide digital video component standard. Its variations come from different sampling formats, such as 4:4:4, 4:2:2, and 4:2:0. As shown in Figure 2.1(a), YC_bC_r 4:4:4 means that the three components (Y, C_b and C_r) have full resolution. On the other hand, in YC_bC_r 4:2:2 and YC_bC_r 4:2:0, the color components have lower chrominance resolution, as depicted in Figures 2.1(b) and (c). By using various sub-sampling, bits allocated for chrominance components are reduced without significant visual quality degradation for the human visual system. We take 4:2:0 sampling as an example. Since each chrominance component contain one quarter of the number of samples in the Y component, YC_bC_r 4:2:0 video needs exactly half as many samples as YC_bC_r 4:2:0 video.

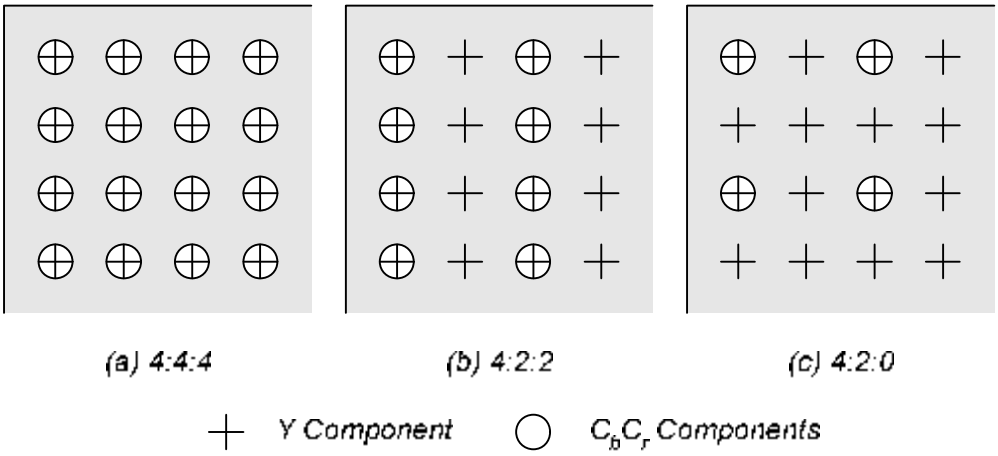


Figure 2.1. Various YCbCr video formats, (a) 4:4:4, (b) 4:2:2, and (c) 4:2:0.

2.2.2 Intra-frame Coding Techniques

In the preceding section, the various digitization formats have been defined for use in different application domains of digital video. Owing to a large amount of data in digital video, compression ratio achieved by only sub-sampling of color components is not enough.

In the context of compression, since a video sequence is simply a sequence of digitized pictures [45], it is also referred to as moving pictures and the terms “frame” and “picture” are used interchangeably. Then, the term “intra-frame coding” refers to the fact that various lossless and lossy compression techniques are performed relative to information that is contained only within the current frame and not relative to any other frame in the video sequence. On the other hand, the term “inter-frame coding” refers to the techniques that employ redundancy between a set of frames.

In this section, intra-frame coding techniques are described first because inter-frame coding techniques are extensions to these basics. As mentioned, no temporal processing is performed outside the current frame for intra-frame coding. Figure 2.2 shows the block diagram of a basic video encoder and decoder for intra frames. It turns out that this block diagram is very similar to that of a JPEG still image encoder and decoder, with only slight difference in implementation detail.

Within each frame, the values of neighboring pixels are often similar to each other. That is, it contains a large amount of spatial redundancy. This redundancy can be removed by coding the data in a more efficient way. Figure 2.2 shows that the intra-frame coding system employs a transform-based technique to remove spatial redundancy [46]. The most popular transform used in video coding standards is discrete cosine transform (DCT). Prior to performing the DCT on each frame, a frame is divided into a number of *8x8* pixel blocks. This block division is necessary since it would be too time consuming to compute the DCT of the entire frame in a single step. Each *8x8* pixel block is then fed sequentially to the DCT which transforms each block separately. That is to say, a block of pixels is transformed into another domain to produce a set of transform coefficients.

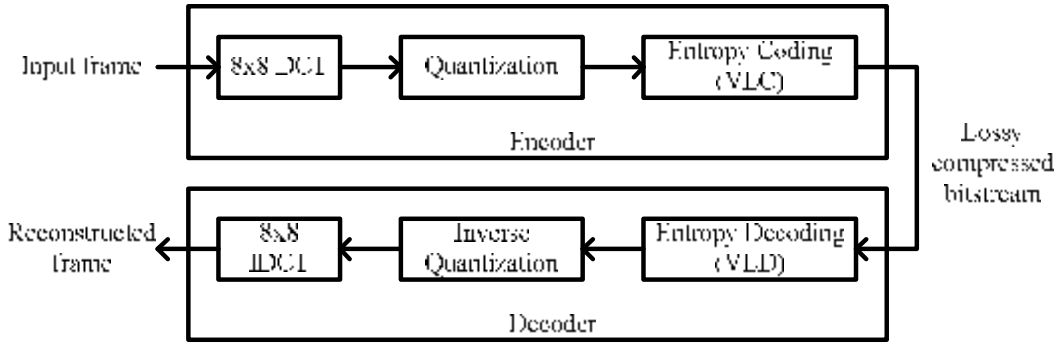


Figure 2.2. Intra-frame or JPEG image codec.

DCT is an effective transform to compact the energy in the block of pixels in which the useful information in the block will be concentrated into a few of the coefficients. This is because the intensity values in an image usually vary smoothly, and very high frequency components exist only near edges. In the *8x8* DCT coefficients, the arrangement of coefficients is followed by its frequency. The most top left-hand corner of the transformed block is the DC component. The rest of coefficients, known as AC coefficients, are arranged following their frequency in ascending order and placed from top left to bottom right of the *8x8* block. Hence the top left-hand corner of the transformed block contains the low-frequency coefficients while the bottom right-hand corner of the block contains the high-frequency coefficients. After transformation, the quantization process is used to discard unimportant transform coefficients, which is the only irreversible process during encoding. In this process, both coefficients are then quantized using uniform or non-uniform quantizers, each

with a different quantization step size. This step size is adjustable to control the desired bit rate and reconstruction quality. Typically, the higher orders of DCT coefficients are quantized with coarser quantization step sizes than the lower frequency ones due to the insensitivity of the human visual system to high frequency distortions. Such arrangement can have further bandwidth compression.

In general, because many of the AC coefficients become zero after quantization, a zig-zag scan is then used to group non-zero quantized coefficients together prior to entropy encoding. As shown in Figure 2.3, the zig-zag scan arranges the low-frequency coefficients before the high-frequency coefficients. Reordering in this way tends to generate long runs of zero-valued coefficients and it takes advantage of runlength coding.

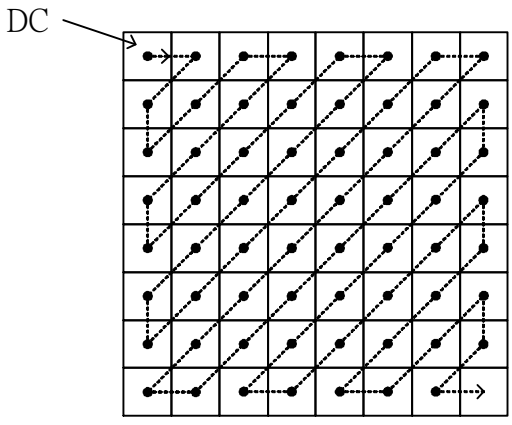


Figure 2.3. A zig-zag scan in an 8x8 DCT block.

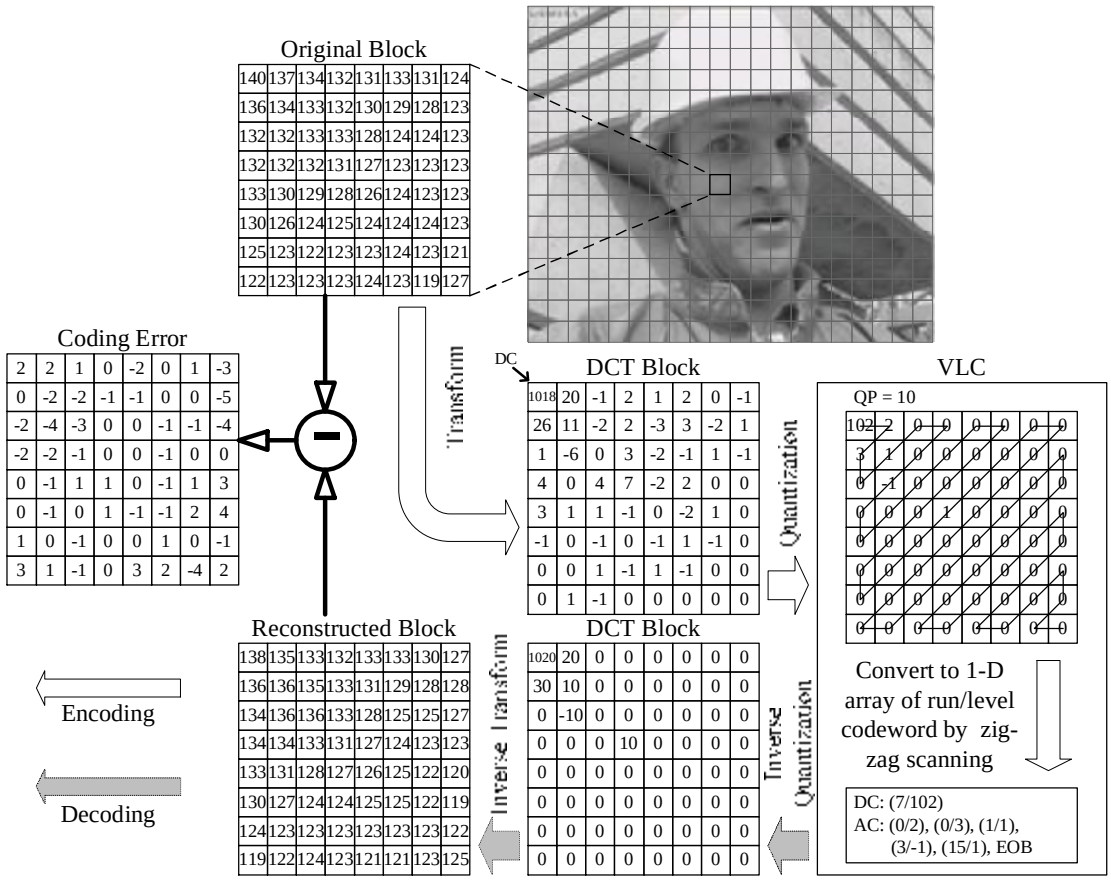


Figure 2.4. An example of intra-frame coding.

Figure 2.4 shows an example to illustrate the concept of intra-frame coding. The frame is first divided into several *8x8* blocks. In this figure, one of the blocks is extracted for illustration. All pixel values in this *8x8* block are similar. An *8x8* DCT is then applied to this block, as shown in Figure 2.4. After transformation, the coefficients with significant values are most likely to be concentrated on the low-frequency components (the top left-hand corner of the block), while the coefficients representing the high-frequency components (the bottom right-hand corner of the block) contain smaller values. All DCT

coefficients are quantized using a quantizer with a quantization factor (QP). In this example, the QP is set to 10. Therefore, the DCT coefficients are divided by 10 and are then rounded to nearest integer. In Figure 2.4, it is found that the output of a typical quantization is a 2-D matrix of coefficients which are mainly zeros except for a number of non-zero values in the top left-hand corner of the matrix. Clearly, if we simply scan the matrix using a raster-scan approach, then the resulting vector would contain a mix of non-zero and zero values. It is not good for run-length coding. To further exploit the presence of the large number of zeros in the matrix that the low-frequency coefficients are more likely to be non-zero than the high-frequency coefficients, a zig-zag scan is used. In this example, the quantized coefficients are converted into a one-dimensional (1-D) array following the zig-zag order, as shown in Figure 2.3. By using the zig-zag scan, it is easy to code the 1-D quantized coefficients efficiently with runlength and variable length coding.

2.2.3 Inter-frame Coding Techniques

In principle, a straightforward approach to compressing a video sequence is to apply the intra-frame coding described in the previous section to each frame independently. This approach is called motion JPEG (M-JPEG).

M-JPEG is frequently used in non-linear video editing since each frame can be simply accessed by any JPEG decoder. However, typical compression ratios obtainable with M-JPEG are between 10:1 and 20:1, neither of which is large enough on its own to produce the desired compression ratios in video coding.

In a general video sequence, scene and objects in the video content changes smoothly and gently. Thus, contents in successive frames are often similar. The similarity between successive frames introduces temporal redundancy, which is especially high in the stationary part of a video sequence such as a static background. However, the previously intra-frame coding techniques are limited to processing the video signal on a spatial basis, relative only to information within the current frame. Considerably more compression efficiency can be obtained if the inherent temporal redundancy is exploited as well.

Most consecutive frames within a video sequence are highly correlated to the frames both before and after the frame of interest. Temporal processing to exploit this redundancy uses a technique known as block-based motion compensated prediction. Therefore, there are two basic types of frames defined in MPEG standards: those are encoded independently and those are predicted from other frames. The first are known as intra-coded frames

(I-frames). I-frames are encoded independently from other frames without exploiting any temporal redundancy. On the other hand, there are two types of inter-coded frames: predicted frames (P-frames) and bi-directional predicted frames (B-frames). Figure 2.5 shows a sequence including all three frame types. The number of frames between successive I-frames is known as a group-of-pictures (GOP). In this example, the GOP size is 6. As we can see in Figure 2.5, I-frames should be used at regular intervals in order to act as an access point for normal video playback and allow a random access operation as it is encoded without predicting from other frames. Although I-frames can provide these important features, the level of compression obtained with I-frames is relatively small.

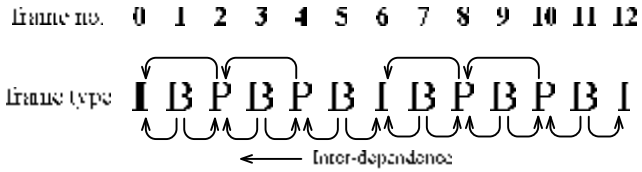


Figure 2.5. Frame structure of the MPEG video coding standard.

In contrast, the encoding of a P-frame attempts to reduce temporal redundancy by predicting a preceding I-frame or a preceding P-frame. P-frames are encoded using a combination of motion estimation and

compensation, and hence significantly higher compression ratio can be obtained with P-frames. In practice, however, the number of P-frames between each successive pair of I-frames is limited. The reason behind is that any errors in the P-frame will be propagated to the next frame. In B-frames, the motion estimation and compensation employs both past and future frames for prediction, which provides better motion estimation when an object moves in front of or behind another object. Note that B-frames have the highest compression ratio and they do not propagate errors since they will never be involved in the coding of other frames.

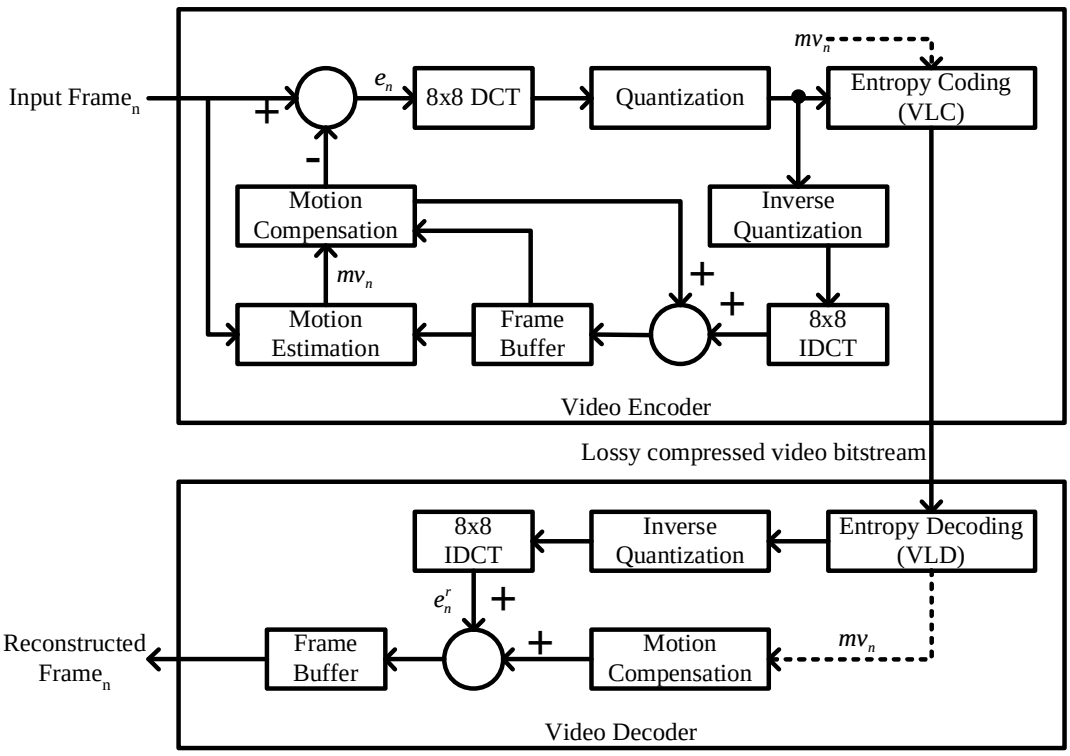


Figure 2.6. Encoding and decoding processes for inter-frames.

Figure 2.6 shows the key steps in inter-frame coding. For the first frame, it must be encoded as an I-frame. The encoding process of an I-frame is an exact procedure that described in Section 2.2.2. To encode a P-frame, each P-frame is predicted from the frame immediately preceding it, whether it is an I-frame or a P-frame. In the MPEG encoder, the video buffer is always filled with a reference frame. The reference frame is then used for motion-compensated prediction to eliminate the temporal redundancy. To achieve this, the encoded contents of P-frames are predicted by estimating any motion that has taken place between the frame being encoded and the preceding I- or P-frame. In the popular video coding standards, the block-matching motion estimation algorithm is always employed. For block-matching algorithm, each video frame is divided into non-overlapping blocks and they are coded using a combination of motion-compensated prediction and transform coding. In practice, the block size for motion estimation may not be the same as that for transform coding. Generally, motion estimation is operated on a larger block known as a macroblock (MB). Most likely, the MB size is *16x16* pixels and the block size is *8x8* pixels. The motion estimation process uses the MB as a basic unit in which pixels of each MB in the current frame are compared on a pixel-by-pixel basis with pixels of

the corresponding MB in the reference frame. If a close match is found, then the relative displacement between the current MB and the best-matched MB in the reference frame is encoded. This is known as a motion vector (MV). This motion estimation process requires intensive operations. Therefore, the computational complexity of encoding an inter-frame is remarkably increased as compared with that of encoding an intra-frame. The predicted MB is obtained from the reference frame based on the motion vector using motion compensation. Then, the prediction error (e) of the current MB is coded by transforming it, quantizing the DCT coefficients and converting them into variable length code words using entropy coding. This procedure is quite similar in principle to that described in encoding of I-frames. Figure 2.6 also depicts the corresponding decoder. The decoder is a reverse process of the encoder, such that variable-length decoding, inverse quantization and IDCT are needed to reconstruct the prediction error (*e*). Then, the reconstructed frame is obtained by adding up the motion compensated prediction and the reconstructed prediction error. Since the motion vectors have already been computed in the encoder, it is not necessary to perform the motion estimation process in the decoder side. The computational requirement of the decoder is significantly reduced as compared to the encoder. Note that the encoder must

emulate the decoding operation to generate the same reconstructed frame as the decoder in order to eliminate any mismatch between the reference frames used for prediction.

The major difference between B- and P-frames is that B-frames use both past and future frames as references. Thus B-frames may use either forward (past frame) or backward (future frame) motion-compensated prediction or both in order to increases the efficiency of motion compensation, particularly when occluded objects exist. There are two motion vectors for each MB – forward and backward motion vectors. In this type of MBs, the coder has a choice of selecting any of the forward, backward or their combination. Since the process of motion estimation doubles, the encoding of B-frame will further increase the computational burden of the encoder. Besides, the use of B-frames requires one additional frame buffer to store the extra reference frame.

Since the adoption of the motion estimation and compensation techniques in the nowadays video coding standards, the increase in frame dependency constrains the decoding or playback direction in which it is convenient for normal playback. Even though the periodic I-frames allow the implementation of random access, fast forward and fast backward operations, variable speed

fast forward/backward playback and smooth reverse playback are still complex if the GOP size is large. Therefore, interactive browsing on compressed video bitstream is still not a trivial task.

2.3 Impacts of Interactive Video Browsing on Decoder Complexity and Network Traffics

Streaming video over the Internet becomes popular in recent years, mainly due to the continued advance of video compression and broadband networking standards [1-3]. These streaming applications such as video-on-demand allow users to ubiquitously access and retrieve various video contents over networks by using software players [20-21] or digital set-top box devices [22-23], which are currently ready to take place of obsolete analog-based devices and systems such as analog TV and Video Cassette Recorders (VCRs). However, current compression standards [5-9] such as MPEG were primarily developed for broadcast applications. In order to complete the transition to digital video from its current analog state, 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 video, but also for

systems wherein the user needs to interact with digital video. A key technique that facilitates fast and user-friendly browsing of video content is to provide full VCR functionality. The set of effective VCR functionality consists of forward, backward, stop, pause, fast forward, fast backward, and random access. This set of VCR functionality allows users to control the video browsing completely and it is also useful for video editing.

However, MPEG video coding standards are suitable for forward-play operations and the predictive processing techniques described in Section 2.2.3 severely complicate other VCR operations. For uncompressed video, the simple solution for implementing various VCR operations is to reorder the video frame data in the desired order. It is viable because the data for each video frame is self-contained. This property typically does not hold true for MPEG video data that is not invariant to changes in frame order. For instance, during backward playback, simply reversing the order of the input frame data will not reverse the order of the decoded video frames. A more complicated process is required to implement a backward-play operation on MPEG data. This scenario is depicted in Figure 2.7 in which a simple I-P structure of an MPEG encoded sequence is considered. To decode a P-frame, the previously encoded I-/P-frames need to be transmitted and decoded first. One

straightforward approach to implement a backward-play operation of MPEG compressed video is to decode all the frames in the whole GOP, store all frames in a large buffer of the decoder, and play the decoded frames reversely. However, this approach requires a huge memory in the client and it is not desirable. Another way 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. For illustration in Figure 2.7, suppose frame *n* is the starting point of the backward-play operation. Since the next frame to be displayed is frame *n-1*, the server selects frame *0* to frame *n-1* from the video stream. At the client side, frame *0* to frame *n-2* do not need to be displayed and only frame *n-1* should be displayed on the user screen. Afterwards, frame *n-1* is decoded and stored into the display buffer so that this frame is displayed on the client screen. This process continues for frame *n-2*, frame *n-3* and so on. As a consequence, straightforward implementation of the backward-play operation requires much higher network bandwidth and decoder complexity than the forward-play operation though it does not require huge memory in the client. The problem is more serious if the GOP size is large. It could be concluded that the conventional video streaming system with VCR support is not practical and therefore some smart schemes are desirable.

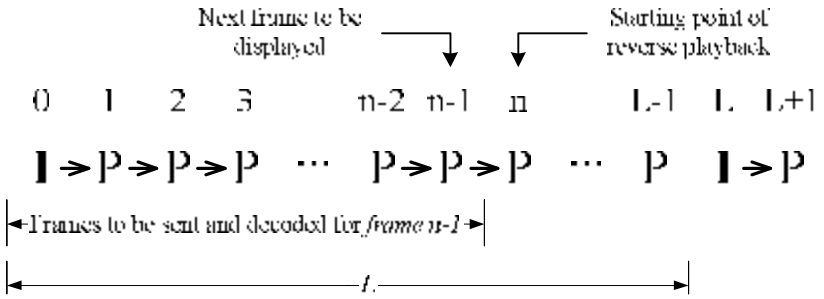


Figure 2.7. Example of backward-play in the MPEG streaming system.

2.4 Efficient Schemes for Enabling VCR Functionalities

In the past decade, several approaches were proposed to support various VCR operations for video streaming. They include a transcoding approach, a multiple time-compressed bitstream approach, and a dual-bitstream approach. These schemes are introduced and discussed in the following subsections.

2.4.1 Transcoding Approach

Transcoding [47-49] is a process to convert the encoded bitstream into another representation of the video contents, which serves for different proposes. Recently, some transcoding works on the implementation of browsing operations [50-57] for MPEG video have been investigated. In [50], the author proposed to reorganize the ordinary linear GOP structure into a binary-tree structured GOP with the I-frame in the centre of the GOP. By

selecting the frames in the new bitstream, the system can provide all directions video playback with any speed-up factor. In [51], a transcoding scheme was proposed to convert the incoming MPEG bitstream with an I-B-P structure into a local bitstream with an I-B structure at the playback device. This local bitstream then allows the device to support interactive playout. Specifically, the transcoding scheme is designed that converts all the retrieved P-frames into I-frames after the decompression and playout of each P frame at the client. This P-to-I conversion is performed after the decompression, thus no additional decoding effort is required. Besides, there is no extra motion estimation and compensation required for encoding this frame into an I-frame. In fact, the P-to-I conversion breaks the inter-frame dependencies between the P-frames and the I-frames. In order to perform the backward-play operation, [52-53] employed the motion vector swapping technique to obtain the reverse motion vectors between the reordered frames and encode these frame into the new I-B bitstream. With this stored I-B bitstream in the secondary storage, VCR functions including random access and fast-forward/backward become straightforward. Since this approach is based on the download mode, all the VCR functions can only be performed among those frames that have been downloaded and displayed. However, full file transfer in the download mode

usually suffers long and perhaps unacceptable transfer time. Besides it requires extra complexity in the decoder to perform the P-to-I conversion and higher storage cost to store the local bitstream in the client.

In [53-54], *Wee et al.* suggested another transcoding scheme which divides the incoming I-B-P bitstream into two parts: I-P frames and B-frames. Then the original I-P bitstream is converted into another I-P bitstream with reverse frame order in compressed domain. Note that motion estimation is known as a quite time consuming process. To make this scheme practical, various fast methods of estimating the reverse motion vectors for the new I-P bitstream with reverse frame order is required in order to reduce the computation involved in the motion estimation of the reverse I-P bitstream. Therefore, several reverse motion estimation methods for the reverse bitstream based on the forward motion vectors of the original I-P bitstream were discussed in [53-54]. An in-place reversal algorithm predicts reverse motion vectors from their corresponding forward motion vectors in the same spatial location. This algorithm is simple, but it only performs well for the interiors of objects that are stationary or undergo uniform translational motion. For regions with a large amount of motion, it frequently produces inaccurate reverse motion vectors near the object boundaries. Another way to perform

reverse motion estimation is to exploit the forward motion vectors of the eight neighboring MBs and the corresponding one. It includes maximum-overlap and weighted-overlap algorithms. In these algorithms, each forward motion vector in the neighborhood is assigned a weighted value that represents its relevance to the current MB for which we must estimate the reverse motion vector. The weighted relevance of each forward motion vector is defined as the size of the overlap area between the current MB and the motion-compensated MB translated by the forward motion vector. The maximum-overlap algorithm then selects the forward motion vector with the largest weight, and then negates its horizontal and vertical components. These components are used as the estimate of the reverse motion vector. For regions with very small translations, the maximum-overlap algorithm provides the identical estimate as the in-place reversal algorithm. But, the maximum-overlap algorithm gives better estimates of the reverse motion vectors near the object boundaries in case of larger translations. The weighted-overlap algorithm also makes use of the corresponding forward motion vector and its neighbors. It constructs the reverse motion vector by summing up all the nine motion vectors with their corresponding weighted values and negating the components of the resulting vector. The

weighted-overlap algorithm produces inaccurate motion vectors at the boundaries of objects undergoing translational motion, but it is better suitable for regions with rotational motion or camera zooms.

For B-frames, there are forward and backward motion vectors referring to the “past” and “future” reference frames correspondingly. We only require to decode them using VLC decoding, extract and swap the pair of motion vectors in B-frames and re-encode them by using VLC coding. The architecture of reverse-play transcoding is shown in Figure 2.8. However, this transcoding process still requires much computation for the decoding and re-encoding of residuals. Besides, the reuse of the incoming motion vectors results in non-optimized outgoing motion vectors. This inaccuracy causes quality degradation. In other words, the reconstructed video in the reverse bitstream are not exactly the same as the original one.

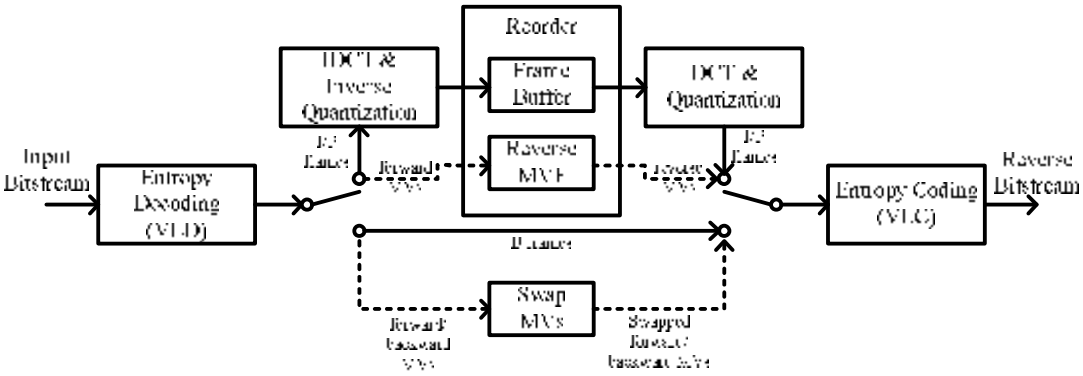


Figure 2.8. The architecture for reverse transcoding.

To reduce the computation required for transcoding, [55-56] proposed some motion re-estimation techniques to find the MVs required in the transcoded video. In [55], the authors proposed an approach to realizing fast forward and backward operation by generating a pre-encoded bitstream with a targeted speed-up factor. In order to compromise between computational complexity and transcoded video quality, this method combines the intra-coding and inter-coding with fast motion re-estimation method to efficiently transcode the required fast forward/backward bitstream. Besides, [56] proposed a fast reverse motion estimation and mode decision algorithm for H.264 reverse transcoding. This algorithm analyzes the MVs and modes decoded from the forward bitstream. The best mode for the backward transcoded MB can be estimated by exploit the relationship among various modes and MVs. In [57], a MB-based reverse-playback algorithm was proposed for video streaming system. This method provides the reverse-play operation at real-time by using the motion characteristic of the video sequence. This system can minimize the required bandwidth and decoder complexity significantly.

2.4.2 Multiple Time-compressed Bitstream Approach

For implementing fast-forward playback, dropping video frames regularly

according to the compression rate is one possible way, and it is known as time-compressed video. For example, when a fast-forward operation with a speed-up factor of 2 requires the compression rate of 50%, half of the total frames are dropped. An alternative is to change the rate at which video frames are rendered. Thus to get a speed-up factor of 2, the frames are displayed at twice this rate. The main disadvantage of this approach is that it is computationally more expensive for the client, as the decoder has to decode twice as many frames in the same amount of time.

In [41], a client-server based video streaming architecture was proposed. This architecture allows discrete speed-up granularity by storing separate pre-processed media files for each speed-up factor. Figure 2.9 illustrates an example of multiple time-compressed bitstreams with four pre-selected speed-up factors. A video sequence is then time-compressed at rate of 1.0, 2.0, 3.0, and 4.0. In this figure, bitstream 1 (B1) is the base bitstream for normal playback without time-compression. Besides, time-compressed bitstreams with speed-up factors of 2.0, 3.0, and 4.0 are represented by B2, B3, and B4, respectively. A bitstream with suitable temporal resolution according to the user's request is then sent. There are several advantages for this time-compressed approach. First, only minimal changes to the client and

server are required. Second, this approach does not introduce extra network traffic since the time-compressed bitstreams are also encoded at the appropriate bitrate. Third, the increase in computational complexity for both server and client is small because the pre-selected bitstreams are encoded offline.

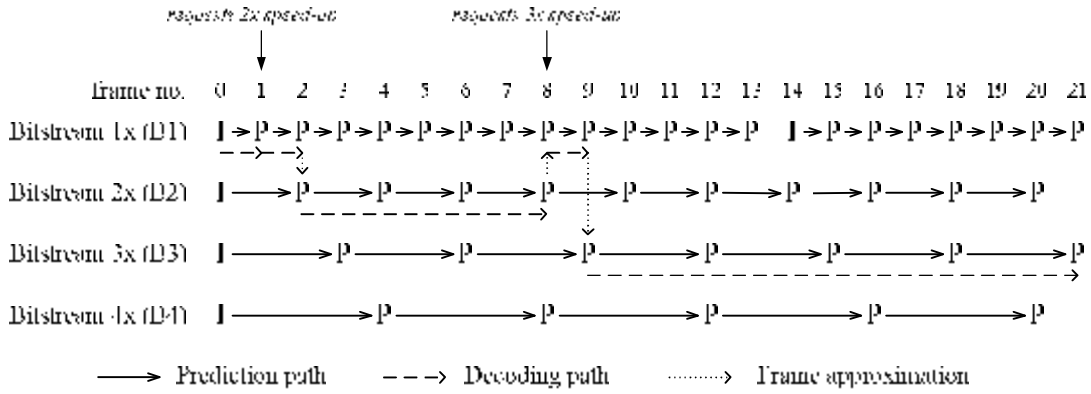


Figure 2.9. An example of time-compressed video: a user requests a fast-forward operation with a speed-up factor of 2 at frame 1 and then changes to a speed-up factor of 3 at frame 8 in the multiple time-compressed bitstream approach.

On the other hand, the speed-up granularity is limited by the number of pre-stored bitstreams and it forces all users to rely on the author's judgment as to speed-up granularity. Besides, additional storage is unavoidable at the server as the number of selected pre-encoded bitstreams increases. For the time-compressed bitstream approach, the storage requirement of the server in terms of the total number of frames is determined by

$$\sum_i \left\lfloor \frac{L}{speedup\_factors(i)} \right\rfloor \tag{2.3}$$

where *speedup_factors(i)* is the speed-up factor of the i^{th} pre-selected bitstreams and *L* is the total number of frames in the video sequence. For example, we use the example as shown in Figure 2.9 again. In this example, *L* is 200. The total number of frames stored in the server is $\lfloor 200/1 \rfloor + \lfloor 200/2 \rfloor + \lfloor 200/3 \rfloor + \lfloor 200/4 \rfloor = 416$, which is more than double as compared to the approach without time-compression.

Furthermore, when a user changes the desired speed-up factor, switching between video bitstreams is not a straightforward task if it is not at a key-frame boundary. The reason is that the current displayed frame may not be the reference of the next displayed frame. For example, the user requests normal playback at the beginning of the video sequence and then changes the speed-up factors to 2 and 3 at frame 1 and frame 8 respectively. The situation is illustrated in Figure 2.9. After reconstructing frame 1 in the decoder, the next nearest frame in B2 is frame 2. Although frame 2 in B2 is the next frame of frame 1, the actual reference frame of frame 2 in B2 is frame 0. Hence, the decoder cannot reconstruct frame 2 in B2 by using the current displayed frame as the reference. To do so, frame 2 in B1 is decoded instead of decoding frame 2 in B2. This reconstruction can be used as an approximation to the

reference for frame 4 in B2. It is noted that the approximation introduces reconstructed errors in frame 4. It will further propagate to subsequent frames. Similarly, when the user requests to increase the speed-up factor to 3 at frame 8, the current displayed frame is given by frame 8 in B2 and the next nearest frame in B3 is frame 9. However, it is easily seen that frame 9 in B3 cannot be decoded by using frame 8 as the reference. The decoder needs to decode frame 9 in B1 by using the reference frame in B2 (frame 8 in B2). Then, frame 12 can be reconstructed by using frame 9 in B1. Again, this induces more serious errors since two approximations are involved in this case.

2.4.3 Dual-bitstream Approach

In [42], a dual-bitstream system was proposed to deal with the problem in different VCR trick modes of the MPEG video streaming system. This approach adds a reverse-encoded bitstream (RB) in the server in addition to the traditional forward-encoded bitstream (FB). The generation of the RB is simply encoding the video in reverse order. Figure 2.10 shows an illustrative example of the FB and the RB in which the video is coded in I- and P-frames with a GOP size of 14 frames. Note that the coding arrangement of I-frames in the RB is interleaved between I-frames in the FB. Since B-frames are not

used for reconstructing 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 video stream contains I- and P-frames only. Upon the server receives the VCR command or the requested frame number from the client, the server employs a frame-selection scheme to determine which frames in either the FB or the RB to be transmitted to the client by minimizing the cost of decoding the next requested frame. In general, a larger number of frames to be sent induce much heavier network traffic and higher decoding complexity. For this reason, the cost can be approximated to the number of frames to be sent over the network [42]. The frame-selection scheme actually measures the distances from the next requested frame to the current displayed frame (d_C), the nearest I-frame in the FB (d_{FB}) and the nearest I-frame in the RB (d_{RB}). It then picks the frame with the minimum distance as the reference frame to the next requested frame to initiate the decoding. Since this selected reference frame has a shorter distance to the next requested frame, a smaller number of frames are necessary to be sent. The frame-selection scheme may switch from the FB to the RB and vice versa according to the current play-direction, the requested mode and the distances d_C, d_{FB}, and d_{RB}. In other words, it determines the selection of the next

bitstream and its decoding direction. To illustrate the scheme, let us recall the structure of dual bitstreams in Figure 2.10 again. Assume that the user requests a random access operation to frame 8, since the requested frame is a P-frame in both bitstreams, the current displayed frame, or the nearest I-frame either in the FB or the RB is firstly selected to initiate the decoding of the requested frames. In this example, frame 8 will be decoded from frame 7 of the RB (an I-frame) since d_{RB} has the smallest value among all distances. It implies the nearest I-frame of the RB (frame 7) is the closest reference to frame 8. Note that frame 7 of the RB (an I-frame) is used as an approximation of frame 7 of the FB (a P-frame) to reconstruct frame 8 of the FB, as depicted in Figure 2.10. This approximation will lead to the drift problem due to the mismatch between the reference frames. Although the drift problem can be compensated by adding the drift bitstreams [42], it increases the storage requirements of the dual-bitstream system. A detailed analysis for this scheme will further be investigated in the next chapter.

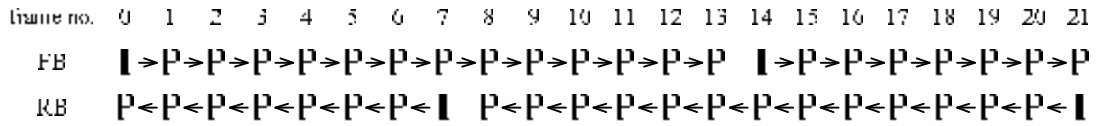


Figure 2.10. The dual-bitstream video streaming system.

An improved dual-bitstream system for VCR functionality and a transcoding technique of the motion vectors between dual bitstreams were

proposed by Huang [43] to solve the drift problem and further reduce the required network bandwidth and decoder complexity. A direct reference bitstream is used to replace the original RB in [42]. Figure 2.11 illustrates the new arrangement of the FB and the new direct reference bitstream. It shows the positions of I- and P-frames in the two bitstreams are the same as the original dual-bitstream system. In the direct reference bitstream, some I-frames in FB are reused. The other P-frames in the direct reference bitstream are directly referenced to their nearest I-frame either in the FB or the direct reference bitstream (DRB). In this way, the number of decoding frames to access any I- or P- frame in the VCR operations is restricted to 1 or 2, and no approximation is involved in the process. Besides, the transcoding technique developed in this work [43] is also developed to derive the direct reference bitstream with less computation complexity. However, due to the longer prediction distance in the direct reference bitstream, the correlation between P-frames and their reference I-frames decreases and thus it reduces the coding efficiency. Besides, in terms of the decoder complexity and network traffic, this approach works well in random access, fast forward and fast backward operations. However it performs worse in backward playback as compared to the original dual-bitstream system.

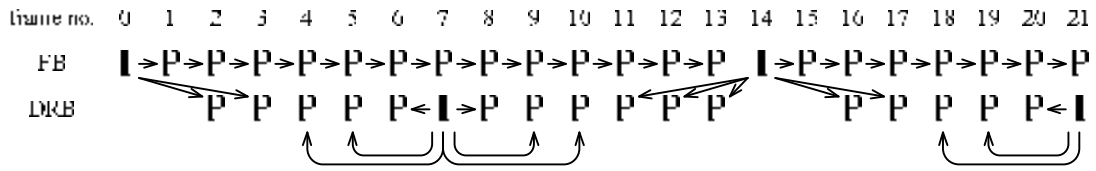


Figure 2.11. The structure of the improved dual-bitstream system.

2.5 Chapter Summary

To solve the difficulties of transmitting coded video data through various networks, many well-known video coding standards were designed. However, the techniques used in these standards are only convenient for normal playback. They put some constraints on other interactive browsing operations. In this chapter, we first gave an overview of some video coding techniques used in the existing coding standards that complicate the video streaming implementation with interactive browsing operations. The main difficulty of video browsing is due to the adoption of motion-compensated prediction. The temporal redundancy exploited by the motion-compensated prediction provides good compression effects. However, this predictive technique also produces dependencies among the frames of the coded bitstream. These dependencies make the implementation of the VCR functionalities in digital video complicated and hinder interactive video browsing. Therefore, we reviewed some approaches that are designed to handle various VCR operations in video streaming. In the transcoding approach, it always

transcodes the received bitstreams into other formats in the decoder for the implementation of VCR functions. It is found that extra storage and complexity in the client are required. This approach is well suited for “download mode” rather than “streaming mode”. After that, we introduced a multiple bitstream video streaming system with VCR support. Instead of using a single bitstream to provide VCR operations, additional bitstreams are encoded and stored in a video streaming server to facilitate fast and user-friendly video browsing. Since extra storage is required to store the additional bitstreams, the multiple-bitstream approach is mainly suitable for streaming servers with less storage constraint. Besides, switching between multiple bitstreams introduces quality degradation during VCR operations. Finally, we also discussed the issue in implementing a dual-bitstream video streaming system. This system simplifies the multiple-bitstream system by storing only two bitstreams, the FB and RB, in the server. A frame-selection scheme is then designed to select the necessary information from the dual bitstreams to access the desired frame in interactive operations. However, some technical challenges such as the drift and storage problems have not yet been well resolved. Therefore, in the following chapters, we investigate the possibility of improving the dual-bitstream system by introducing various novel techniques.

Chapter 3 – Dual-bitstream Video Streaming System with Full VCR Functionalities

3.1 Introduction

In Chapter 2, we have reviewed the dual-bitstream system that provides full VCR functionalities over a network with minimum requirements on the network bandwidth and the decoder complexity [42]. Besides the traditional forward-encoded bitstream (FB), a reverse-encoded bitstream (RB) is added in the server for backward playback, where the I-frames in the RB are interleaved between the I-frames in the FB. The server then employs a frame-selection scheme to choose and transmit appropriate frames in either the traditional FB or the RB for any speed-up factors of VCR operations. In practice, the frame-selection scheme uses a least-cost method to minimize the decoder effort at the client decoder or the traffic over the networks, or a combination of both. To achieve this, one simple approach is to measure the cost of decoding the next requested frame. Therefore, three costs are defined in the frame-selection scheme: d_C, d_{FB}, and d_{RB}. d_C is the number of frames from the next requested frame to the current displayed frame, d_{FB} is the number of frames from the next requested frame to the nearest I-frame in the FB, and d_{RB}

is the number of frames from the next requested frame to the nearest I-frame in the RB. Based on these costs, the current play-direction, and the requested trick mode, the frame with the minimum cost is selected to initiate the decoding. This mechanism enables the implementation of various VCR operations such as forward, backward, fast forward, fast backward, and random access.

The implementation of full VCR functionality in the dual-bitstream system, however, poses some difficulties that have not yet been well resolved. In this chapter, we investigate the impacts of the VCR functionality on the network traffic, the video decoder complexity, and the reconstructed video quality during various VCR operations for this dual-bitstream system.

3.2 Forward and Backward Playback Operations

Forward playback is the basic VCR mode. All the video coding standards are primarily designed to facilitate the forward playback operation. To achieve high compression ratios, they employ motion-compensated prediction to reduce temporal redundancy. In this case, the decoder uses the previous reconstructed frame to predict the current decoding frame, and then store the resulted frame in the frame buffer for acting as a future reference. In other words, when the decoder performs normal playback, the next frame to be

decoded always uses the current frame as the reference, thus d_C is the minimum among the aforementioned three costs.

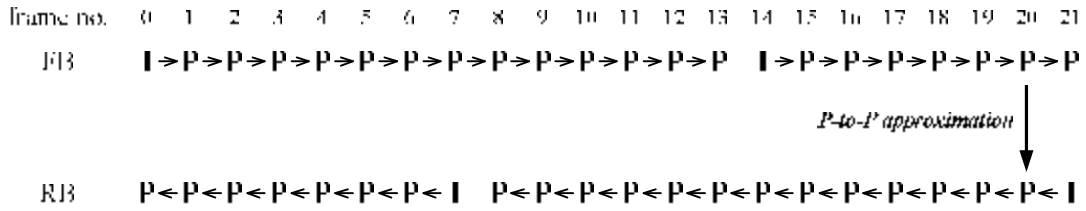


Figure 3.1. Backward operation in the dual-bitstream system.

On the other hand, providing backward playback on the dual bitstreams is mainly based on the RB. The decoder always uses the current decoded frame in the RB as the reference to reconstruct the next decoding frame except bitstream switching from the FB to RB is required. This is the case when the current VCR mode is in the forward play and then a backward operation is requested. Figure 3.1 shows this situation in which the current position is at frame 20 and then a backward-play request is launched. Frames 19, 18, 17, etc will be displayed in order. When frame 19 is requested, d_C has the minimum value. Thus the current displayed frame of the FB (frame 20, which is a P-frame) is selected as the approximation of frame 20 of the RB (P-frame) to predict the requested frame (frame 19) in the RB. This P-to-P approximation will lead to the drift problem due to the mismatch occurring in the

reference frame. Furthermore, the drift will not only be confined to the frame at the switching location, but will further propagate in time. Note that an P-to-I approximation occurs when a backward-play operation is requested at frame 21 in this example. The quality degradation in this situation for a GOP is depicted in Figure 3.2 where the test video stream used for simulation is "Salesman" sequence. The "Salesman" sequence is encoded at *3* Mb/s with a frame-rate of 30 fps and the GOP length is *14* with an I-P structure. The starting point of the backward-play operation is at frame 20. This figure shows that the drift caused by the bitstream switching can be as large as 2.5dB and will last until the next I-frame in the RB.

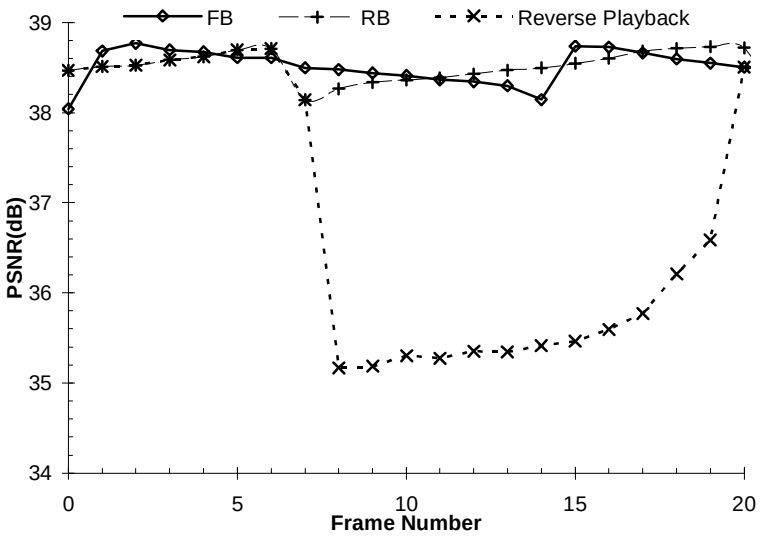


Figure 3.2. Quality degradation in the situation when the current position is at frame 20 and then a backward-play operation is requested.

3.3 Random Access Operation

In random access, a frame with an arbitrary distance from the current displayed frame is requested. It enables users to access different video scenes without decoding the unnecessary video contents. However, the decoding dependency of P-frames immediately implies that, in order to decode and display a specific frame within the video bitstream, it might be necessary that several other frames have to be first decoded. If the server only has the FB and the requested frame is a P-frame, the server needs to send all the P-frames from the previously nearest I-frame to this requested frame. Obviously, this requirement could be very expensive for the decoder but also for the network bandwidth. It is more serious for a video bitstream with a longer GOP length.

In the dual-bitstream system, the arrangement of interleaved I-frames shortens the distance between the requested frame and the closest I-frames in either the FB or the RB. We use Figure 3.1 again, if the user requests a random access operation to frame 7, the least-cost scheme finds that d_{RB} has the smallest value and frame 7 (an I-frame) in the RB will then be transmitted to the client. Comparing with the conventional approach, the number of frames to be transmitted reduces significantly.

3.4 Fast-forward and Fast-backward Operations

Fast-forward/backward operations allow fast preview of video contents, the speed-up factor determines the rate of time compression. A straightforward way to implement the fast-forward/backward operation in a single compressed bitstream is to decode the I-frames only and ignore all P-/B-frames. This scheme provides a limited speed-up ratio to users and the allowable speed-up factor is constrained by the size of GOP. Although it is simple, it cannot satisfy the user's need.

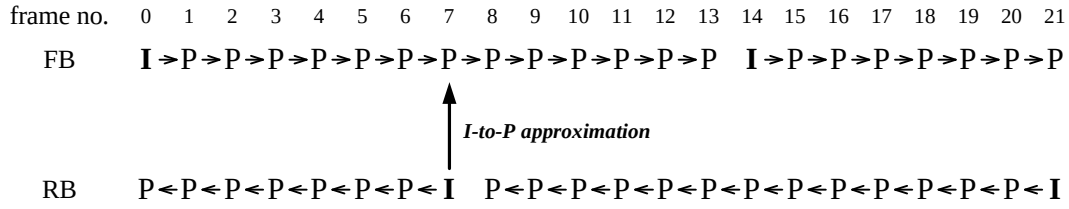


Figure 3.3. A Fast-backward operation with the speed-up ratio of 6 in the dual-bitstream video streaming system.

On the other hand, the dual-bitstream system can provide a smooth and variable speed-up factor for fast-forward/backward operations. To illustrate the scheme, let us use the structure of dual bitstreams in Figure 3.3. Assume that the previous mode is in the normal forward playback at frame 20 and the requested mode is fast-backward playback with a speed-up factor of 6. This operation needs to display frames 14, 8, etc. If the requested frame is an

I-frame in one of the two bitstreams, the frame can be decoded by itself. Thus, in the above example, frame 14 will be decoded from the FB directly since it is an I-frame. Then, the next frame to be decoded is frame 8. Since the requested frame is a P-frame in both bitstreams, the current displayed frame, or the nearest I-frame either in the FB or the RB is selected to initiate the decoding of the requested frame. In this example, frame 8 will be decoded from frame 7 of the RB (an I-frame) since d_{RB} has the smallest value among all distances. It implies the nearest I-frame of the RB (frame 7) is the closest reference to frame 8. Note that frame 7 of the RB (an I-frame) is used as an approximation of frame 7 of the FB (a P-frame) to reconstruct frame 8 of the FB, as depicted in Figure 3.3. This I-to-P approximation will introduce the drift problem since the mismatch of the reference frame exists. Besides, the drift will not only be restricted to the frame at the switching location, but will propagate to other frames if they are temporally predicted from the drifted frame. This is the case when the current fast-reverse mode is switched back to normal playback at frame 8. In this situation, frame 9 will be decoded by using the drifted frame, frame 8. This drift further propagates until the next I-frame in the FB. Figure 3.4 illustrates the quality degradation due to the I-to-P approximation. In this figure, the test video used for simulation is “Foreman” sequence. The

"Foreman" sequence is encoded at 330 Kb/s with a frame-rate of 30 fps. The GOP length of the encoded sequence is 14 with an I-P structure. The starting point of the fast-backward operation with a speed-up factor of 6 is launched at frame 20 and the normal-play operation is then requested at frame 8. This figure indicates that the drift caused by bitstream switching is around 1.3 dB and will last until the next I-frame in the FB.

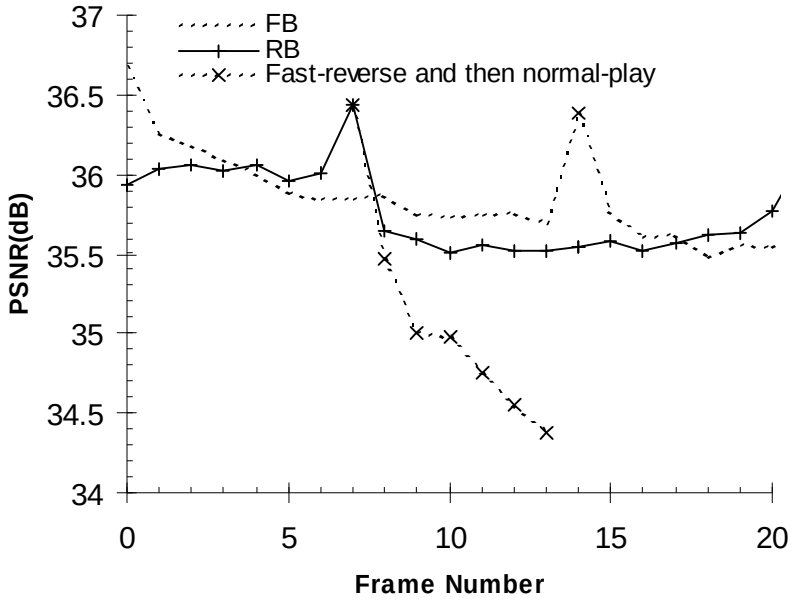


Figure 3.4. Quality degradation in the situation when the fast-reverse operation with a speed-up factor of 6 is requested at frame 20 and then the normal playback is issued at frame 8.

It should be noted that if the speed-up factor is high enough (larger than *N/4*, where N is the GOP length), we always find an I-frame in one of the two bitstreams which has shorter distance to the next request frame than the

current decoded P-frame since the distance for the nearest I-frame is guaranteed to be equal to or less than *N/4*. In this case, bitstream switching might happen. On the other hand, if the speed-up factor is less than or equal to *N/4*, it is not necessary to perform bitstream switching.

3.5 Drift Compensation

As discussed above, the reference mismatch problem caused by the P-to-P or I-to-P approximation will cause drift when the approximated frames are used as the reference frames to predict the subsequent frames. To resolve the problem, two extra drift-compensated bitstreams for switching from the FB to the RB (D^{FR}) and from the RB to the FB (D^{RF}), as depicted in Figure 3.5(a), were also suggested in [42]. Since bitstream switching may occur at any time, a switching point should be appeared at any frame position. Figure 3.5(a) shows that the drift-compensated frames are encoded at all possible switching points. For instance, when a VCR operation triggers bitstream switching from the FB to RB at frame *n* (from FB_n to RB_{n-1}), the server will send the drift-compensated frame at frame *n-1* (D_{n-1}^{FR}) instead of the RB at frame *n-1*(RB_{n-1}). To compensate the drift, D_{n-1}^{FR} can be obtained as

$$D_{n-1}^{FR} = RB_{n-1} - Interpred(FB_n, mv_{D_{n-1}^{FR}}) \quad (3.1)$$

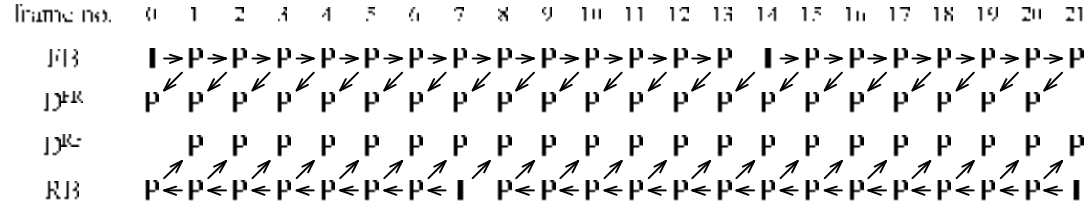
where $Interpred(FB_n, mv_{D_{n-1}^{FR}})$ is the motion compensation operator that RB_{n-1} is predicted from the reference, FB_n, with the given motion vector $mv_{D_{n-1}^{FR}}$. Since D^{FR} is encoded based on the reference frame from the FB, the drift at the switching point can be compensated to a certain extent. Although the drift is reduced by the drift-compensated frame, it does not guarantee that they can be completely eliminated. In fact, the amount of the drift reduced depends on the quantization step-size used in the encoding of D^{FR}. A finer quantizer will lead to lower drift, but result in larger storage for drift-compensated frames. In contrast, a coarser quantizer can reduce the storage requirement while causing larger drift.

Similarly, if bitstream switching from the RB to FB at frame *n* (from RB_n to FB_{n+1}) is required, D_{n+1}^{RF} can be constructed as

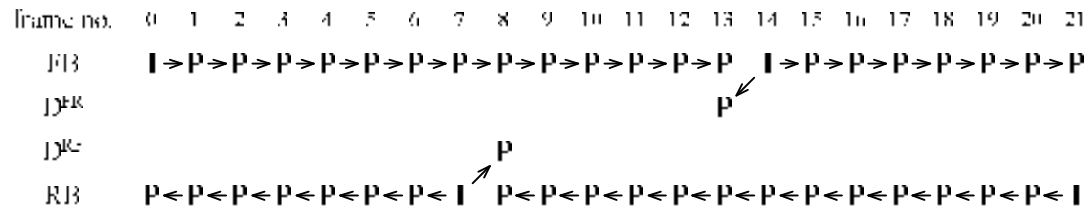
$$D_{n+1}^{RF} = FB_{n+1} - Interpred(RB_n, mv_{D_{n+1}^{RF}}) \tag{3.2}$$

Obviously, the overall storage requirement of the dual-bitstream system with drift compensation is about four times the conventional system with the FB only. To achieve a better trade-off between the storage requirement and quality performance, only drift compensation for I-to-P approximations is sufficient. It is because the I-to-P approximation frequently occurs in fast-forward/backward operations with the large speed-up factor (S). Since the I-frames in the two

bitstreams are interleaved, d_{FB} and d_{RB} of the next requested frame must be smaller or equal to $\lceil N/4 \rceil$. For the speed-up factor larger than *N/4* (i.e. *S>N/4*), d_C is equal to *S+1*. It means d_C must be larger than d_{FB} or d_{RB}. In this case, the I-to-P approximation is required for each requested frame and the mismatch problem continues until a user requests normal playback. Since the probability of requesting a fast-forward/backward operation with the large speed-up factor is likely, a possible low-complexity solution for the drift-compensation scheme can be simplified by only compensating for the drift due to the I-to-P approximation. Figure 3.5(b) shows the simplified drift-compensation scheme. The number of drift-compensated frames depends on the size of GOP. If the GOP size is small, more drift-compensated frames are necessary to be encoded.



(a) Drift compensation for all frames.



(b) Simplified drift compensation.

Figure 3.5. Drift compensation schemes for the dual-bitstream video streaming system.

3.6 Chapter Summary

In this chapter, we presented the implementation of various VCR operations including forward/backward playback, random access and the fast-forward/backward playback in the dual-bitstream system. Since the I-frames of the two bitstreams are interleaved, the least-cost frame-selection scheme can find a shorter decoding path to decode the requested frame from both the FB and the RB. However, an I-to-P approximation or a P-to-P approximation is required to switch between the FB and the RB. These reference frame approximations cause the reference mismatch for decoding the subsequent frames. The mismatch errors are not only affected the frame at the switching position, but will further propagate and stop until the next received I-frame. Although the mismatch errors can be alleviated by the additional drift-compensated bitstreams, it cannot completely solve the drift problem. Besides, the storage requirement of the dual-bitstream streaming server further increases when the drift compensation scheme is applied.

In conclusion, the novelty of the dual-bitstream system provides full VCR functionalities with the reduced network bandwidth and decoder complexity, but it brings the storage and quality issues. Results of our investigation indicate that this dual-bitstream approach is still primitive, and there is plenty of room for

improvement.

Chapter 4 – Dual-bitstream H.264 Video Streaming System with VCR Functionalities using SP/SI-frames

4.1 Introduction

In the previous chapter, we have identified the problems of the dual-bitstream MPEG video streaming system. For instance, the dual-bitstream structure is designed based on I/P-frames which would lead to drift errors during video browsing. A further need for supporting drift-free video browsing may arise. Therefore, in this chapter, we provide an efficient solution to perform various fast playbacks in the dual-bitstream streaming system to eliminate the problem arising from the mismatch between the forward-encoded and reverse-encoded bitstreams. Our solution adopts the concept from SP/SI-frames [35-37] and makes some modifications of the dual bitstreams so that it can ensure no mismatch problem existing in any fast-play operations.

4.2 Proposed Dual-bitstream Structure based on SP- and SI-frames

In this section, we present a completely new prediction structure of dual bitstreams that facilitates drift-free video browsing capability. By introducing

SP/SI-frames in the dual bitstreams, the problem of reference mismatch due to the I-to-P approximation can be avoided. Therefore, it can maintain the quality of reconstructed frames during fast playback.

The main feature of SP/SI-frames is that identical reconstruction can be achieved even when different reference frames are used for prediction. This property motivates us to adopt SP/SI-frames in the structure of dual bitstreams to prevent reference mismatch. In this modified dual-bitstream structure, we propose to adopt SP/SI-frame pairs at switching points which are the points when a frame of one bitstream may be used to replace a frame of the other bitstream. This arrangement ensures that the I-to-P approximation will not happen at switching points for fast-forward/reverse operations. Consequently, the new dual-bitstream system will not cause any mismatch at the decoder side. In the following, we provide a detailed description and formulation of the dual-bitstream structure.

4.2.1 Overview of SP/SI-Frame Coding in H.264

Video streaming over the Internet and 3G wireless networks has emerged as one of the popular video applications. The nature of these networks causes a fluctuation of the usable bandwidth available to a user, due to changing

networks conditions. One obvious approach for efficient adaptation to channel bandwidth is by compressing each video sequence into multiple and independent bitstreams of different bit rates. The video server then dynamically switches among these bitstreams to accommodate the channel bandwidth variation. However, the temporal predictive coding technique employed in P-frames leads to difficulties in this switching operation, i.e., switching at a P-frame would result in different references at the decoder, and such a mismatch may bring the so-called drift which could propagate and be accumulated in the subsequent frames until the next I-frame. For that reason, the newest H.264 standard introduces SP/SI-frame coding to enable seamless bitstream switching. SP/SI-frames can offer an identical reconstructed frame even when different reference frames are used for their prediction. An example of bitstream switching using SP-frames is shown in Figure 4.1, where a video sequence is encoded into two bitstreams (bitstream A and bitstream B) with different bit rates and quality level. Within each encoded bitstream, two SP-frames - $SP_{A,n}$ and $SP_{B,n}$, as shown in Figure 4.1, are placed at frame n where switching from one bitstream to another is allowed. These SP-frames are known as primary SP-frames. Besides, for each primary SP-frame, a corresponding secondary SP-frame ($SP_{AB,n}$ in Figure 4.1) is generated, which

has the same reconstructed values as the primary SP-frame. Such a secondary SP-frame is sent only during bitstream switching. For normal transmission, either bitstream A or bitstream B is sent to the user depending on the current available bandwidth. When there is a need to switch the transmitting bitstream from bitstream A to bitstream B at frame *n*, $SP_{AB,n}$ instead of $SP_{B,n}$ is transmitted. After decoding $SP_{AB,n}$, the decoder can obtain exactly the same reconstructed values as normally $SP_{B,n}$ decoded at frame *n*, therefore it can continually decode bitstream B at frame *n+1* seamlessly. The way to encode $SP_{AB,n}$ ensures that an identical reconstruction as that of $SP_{B,n}$ can be obtained by decoding it so that the bitstream switching process will not introduce any mismatch between the encoder and decoder.

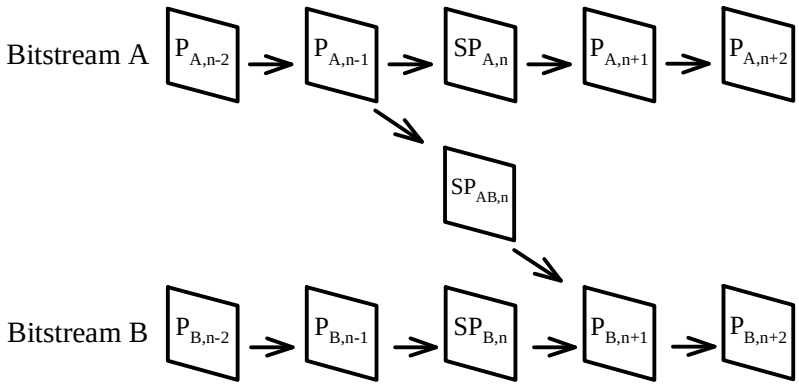


Figure 4.1. Bitstream switching using SP pictures.

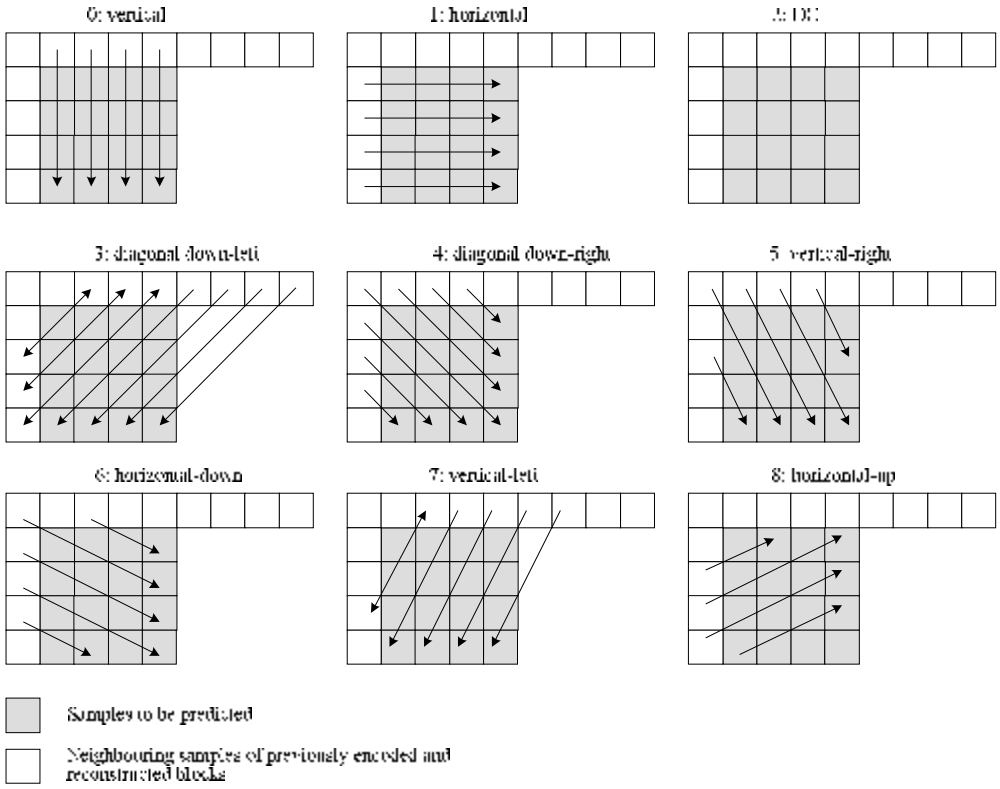


Figure 4.2. Nine intra prediction modes for a 4×4 subblock.

Similarly, an SI-frame can also be used in the bitstream-switching scenario. The only difference is that its prediction is formed using the intra-prediction modes from previously decoded samples of the reconstructed frame. The SI-frame may be used when the two bitstreams are completely different. In this case, it is inefficient to use motion-compensated prediction because there is no correlation between these two sequences. It is noted that, in H.264, it uses the method of predicting intra-coded MBs to reduce the high amount of bits coded by original input signal itself [38]. For encoding a block or MB in an intra-prediction mode, a prediction block is formed based on previously

reconstructed blocks. The residual signal between the current block and the prediction is then encoded. For the luminance samples, the prediction may be formed for each 4×4 subblock, each 8×8 block, or for a 16×16 MB. Figure 4.2 shows the nine intra-prediction modes defined in the H.264 standard and the arrows in Figure 4.2 indicate the direction of prediction in each mode. For more details about intra-prediction modes, interested readers are encouraged to refer to [38-40]

4.2.2 Motivation of using SP/SI-frames in the Dual-bitstream Structure

As mentioned in Chapter 3, the original dual-bitstream system suffers the drift problem due to the I-to-P approximation during fast playbacks. We suggest adopting the concept of SP/SI-frames in the structure of the dual bitstreams to eliminate the problem of reference mismatch for switching between the FB and the RB. In the following, we describe how to modify the original dual-bitstream structure with the help of SP/SI-frame coding. It aims at providing seamlessly approximation between frames in the forward-encoded bitstream (FB) and the reverse-encoded bitstream (RB) for any VCR operations.

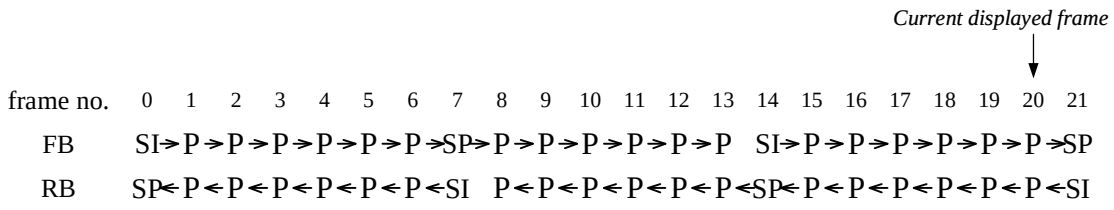


Figure 4.3. Structure of the proposed dual bitstreams with SP/SI-frame pairs.

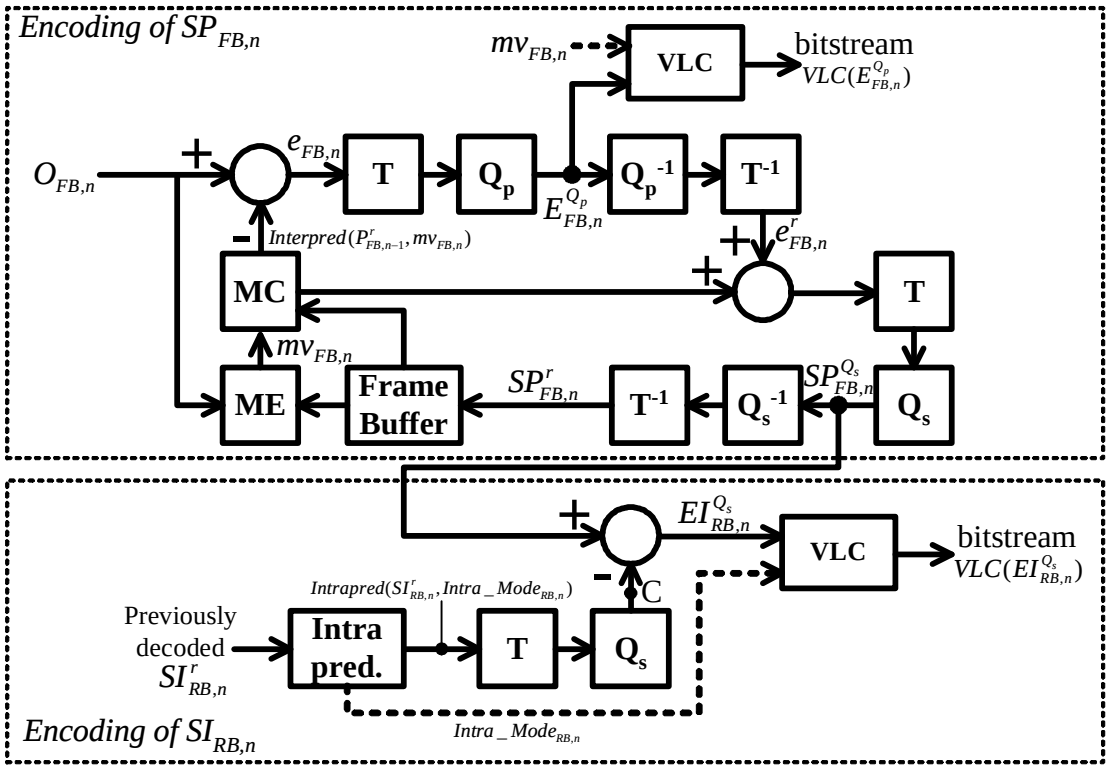
The idea of utilizing SP/SI-frame coding for the purpose of the seamless approximation in VCR operations is introduced in Figure 4.3. In the original structure of the dual bitstreams, I-frames represent the points of access to decode the sequence from any arbitrary position. In order to avoid the problem of reference mismatch, instead of using I-frames in the original dual bitstreams, SI-frames are used at the switching points. An SI-frame can be placed either in the FB or RB which is referred to as a forward-encoded SI-frame (SI_{FB}) or a reverse-encoded SI-frame (SI_{RB}) respectively. For each SI_{FB}, there is a corresponding reverse-encoded SP-frame (SP_{RB}) and these two frames form an SI_{FB}/SP_{RB} pair. Similarly, there is a corresponding forward-encoded SP-frame (SP_{FB}) for each SI_{RB} and they form an SP_{FB}/SI_{RB} pair. Identical frame reconstruction can be allowed at these SI_{FB}/SP_{RB} and SP_{FB}/SI_{RB} pairs when an SI-frame of one bitstream is used to replace an SP-frame of the other bitstream (SI-to-SP replacement). This facilitates the correctly predicted frame to be used if the SI-to-SP replacement is needed in

any VCR operation and hence no drift will occur. To illustrate the proposed scheme, we use the example in Figure 3.3 again in which the current playback is in the fast-reverse mode and this operation requires to display frames 14, 8, etc. After decoding and displaying frame 8, then a normal-play request is launched. Frame 8 will be decoded from frame 7 of the RB, which is an SI_{RB}. After decoding this SI_{RB}, the decoder can obtain exactly the identical reconstruction as normally decoding SP_{FB} at frame 7 in the FB, therefore it can decode frame 8 in the FB without any mismatch error and continue to decode the subsequent frames seamlessly when the normal playback is requested at frame 8. Hence, the proposed dual-bitstream system does not suffer the drift problem because no approximation between the SI-frame and the SP-frame on the dual bitstreams has been made.

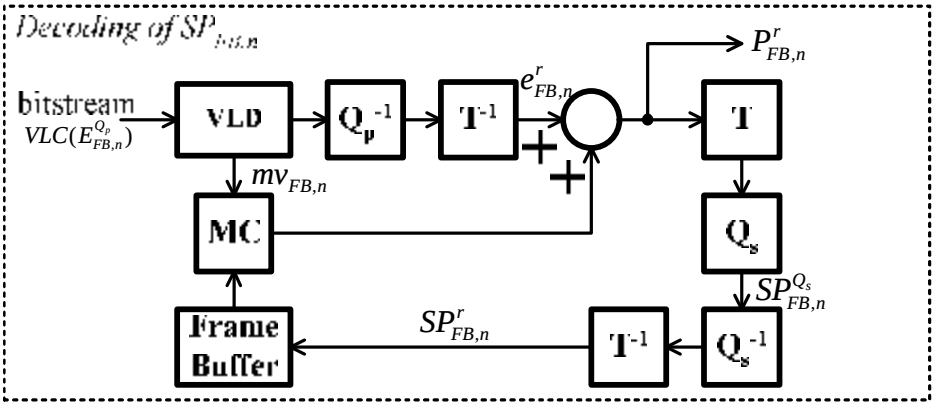
4.2.3 Encoding and Decoding Arrangement of SP_{FB}/SI_{RB} and SI_{FB}/SP_{RB} Pairs

This section provides a detailed description of how to encode the SP_{FB}/SI_{RB} and SI_{FB}/SP_{RB} pairs. All forward SP/SI-frames (SP_{FB} and SI_{FB}) are encoded as primary SP/SI-frames. Figure 4.4(a) shows the encoding processes of SP_{FB} at frame n ($SP_{FB,n}$) and its corresponding SI_{RB} ($SI_{RB,n}$). For

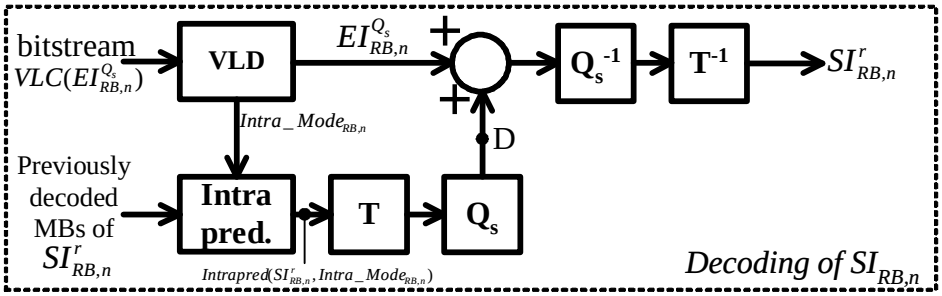
the sake of convenience, we use the superscript *r* to denote the reconstructed frame or reconstructed prediction error, and the capital letter with superscript Q_p or Q_s to represent the coefficients in the quantized transform domain with the quantization level Q_p or Q_s respectively for the rest of this chapter.



(a)



(b)



(c)

Figure 4.4. Encoding and decoding of an SP_{FB}/SI_{RB} pair. (a) SP_{FB} and SI_{RB} encoding, (b) SP_{FB} decoding, and (c) SI_{RB} decoding.

A predicted frame is formed by motion-compensated prediction using the original frame *n* ($O_{FB,n}$) in the FB and the previously reconstructed frame *n-1*, $P^r_{FB,n-1}$, stored in the frame buffer. It can be written as $Interpred(P^r_{FB,n-1}, mv_{FB,n})$ where *Interpred( )* is the motion-compensated prediction operator and $mv_{FB,n}$ is the motion vectors of frame *n* in FB. The prediction error $e_{FB,n}$ between $O_{FB,n}$ and its prediction is

$$e_{FB,n} = O_{FB,n} - Interpred(P^r_{FB,n-1}, mv_{FB,n}) \quad (4.1)$$

Performing the transformation and quantization on $e_{FB,n}$ with Q_p, we get

$$E^{Q_p}_{FB,n} = Q_p(T(e_{FB,n})) \quad (4.2)$$

$E^{Q_p}_{FB,n}$ is then compressed into the bitstream $VLC(E^{Q_p}_{FB,n})$ with entropy coding. The above encoding process is exactly the same as the normal P-frame encoding. To generate $SP_{FB,n}$, an additional quantization process is applied to the P-frame. This additional quantization process ensures that the transform

coefficients of the reconstructed frame $SP^r_{FB,n}$ can be quantized and de-quantized without loss at the quantization level Q_s, which is going to be used in the encoding process of $SI_{RB,n}$. Specifically, $P^r_{FB,n}$ is not the same as $SP^r_{FB,n}$, applying quantization Qs on $P^r_{FB,n}$ must degrade the visual quality of $P^r_{FB,n}$. However, it is difficult to achieve identical reconstruction of $P^r_{FB,n}$. By using the additional quantization Qs, the output signal $SP^r_{FB,n}$ is divisible by Qs (i.e. $SP^r_{FB,n} = T^{-1}(Q_s^{-1}(Q_s(T(SP^r_{FB,n}))))$) and can be reconstructed without introducing any loss. Note that the server using the proposed dual bitstreams with SP/SI-frames still stores $VLC(E^{Q_p}_{FB,n})$ instead of the bitstream generated by $SP_{FB,n}$. This $E^{Q_p}_{FB,n}$ can be used to generate $SP_{FB,n}$ in the local decoder loop of the SP-frame encoder as well as the decoder. In the following, we describe how to generate $SP_{FB,n}$ which is used to update the frame buffers in both encoder and decoder to act as a reference for the next P-frame.

Before performing an additional quantization process on the P-frame, the reconstructed prediction error $e^r_{FB,n}$ is obtained by applying dequantization with the quantization level Q_p and then inverse transformation on $E^{Q_p}_{FB,n}$.

$$e^r_{FB,n} = T^{-1}(Q_p^{-1}(E^{Q_p}_{FB,n})) \tag{4.3}$$

The reconstructed prediction error $e^r_{FB,n}$ is then added to $Interpred(P^r_{FB,n-1}, mv_{FB,n})$ for computing the reconstructed P-frame $P^r_{FB,n}$, which

can be written as

$$P^r_{FB,n} = e^r_{FB,n} + Interpred(P^r_{FB,n-1}, mv_{FB,n}) \quad (4.4)$$

$P^r_{FB,n}$ is then transformed again and the transform coefficients of $P^r_{FB,n}$ is quantized using the quantization level Q_S. The quantized transform coefficients of $P^r_{FB,n}$ become

$$SP^{Q_s}_{FB,n} = Q_s(T(P^r_{FB,n})) \quad (4.5)$$

After performing dequantization with Q_S and inverse transformation on $SP^{Q_s}_{FB,n}$, we obtain an expression of the reconstructed frame $SP^r_{FB,n}$.

$$SP^r_{FB,n} = T^{-1}(Q_s^{-1}(SP^{Q_s}_{FB,n})) \quad (4.6)$$

This reconstructed frame $SP^r_{FB,n}$ acts as a reference frame for the next P-frame, which is stored in the frame buffer of the encoder. Equations (4.5) and (4.6) indicate that the transform coefficients of $SP^r_{FB,n}$ can be quantized and de-quantized without loss at the quantization level Q_S. These represent the fundamental ideas of SP/SI-encoding to ensure identical reconstruction of frame *n* even when different predictions are used to encode $SI_{RB,n}$.

The decoding process of $SP^r_{FB,n}$ is shown in Figure 4.4(b) in which $VLC(E^{Q_p}_{FB,n})$ is an input bitstream. The decoder actually is the local decoder loop of encoding process for $SP_{FB,n}$ as shown in Figure 4.4(a). Its decoding operation follows (4.3) to (4.6). Finally, an identical $SP^r_{FB,n}$ to the encoder is

reconstructed to update the frame buffer in the decoder which is used for decoding the next P-frame.

In order to avoid the problem of reference mismatch due to the SI-to-SP replacement, the reconstructed frame $SI^{r}_{RB,n}$ at frame *n* in the RB must be identical to $SP^{r}_{FB,n}$ in the FB. To achieve this, the quantized values of transform coefficients in both $SP^{r}_{FB,n}$ and $SI^{r}_{RB,n}$ must be synchronized to the same quantization level Q_S. By using the same encoder as shown in Figure 4.4(a), $SP^{Q_s}_{FB,n}$ acts as an input instead of the original video frame in the RB, and the quantization operation is processed before calculating the prediction error of $SI_{RB,n}$, $EI^{Q_s}_{RB,n}$, which can be computed as

$$EI^{Q_s}_{RB,n} = SP^{Q_s}_{FB,n} - Q_s(T(Intrapred(SI^{r}_{RB,n}, Intra\_Mode_{RB,n}))) \quad (4.7)$$

where *Intrapred( )* is the intra-prediction operator and $Intra\_Mode_{RB,n}$ represents intra-prediction modes as depicted in Figure 4.2. For coding $SI_{RB,n}$, the prediction is formed by using the intra-prediction modes from previously decoded samples of the reconstructed frame. Again the encoder compresses the bitstream $VLC(EI^{Q_s}_{RB,n})$ with entropy coding. Since the same quantization level Q_S are used for $SP^{Q_s}_{FB,n}$ and $Q_s(T(Intrapred(SI^{r}_{RB,n}, Intra\_Mode_{RB,n})))$ in (4.7), $EI^{Q_s}_{RB,n}$ is also synchronized at Q_S and then it can be quantized and de-quantized without any loss at Q_S.

By decoding the bitstream $VLC(EI_{RB,n}^{Q_s})$ in the decoder, the identical reconstruction to $SP_{FB,n}^{r}$ can be obtained as explained below. First, the transformed and quantized coefficients of different predictions will not introduce any mismatch, that is, the value at position C in the encoder and at position D in the decoder (Figure 4.4(c)) are exactly the same. In other words, both positions C and D can create the prediction $Q_s(T(Intrapred(SI_{RB,n}^{r}, Intra\_Mode_{RB,n})))$. Second, this prediction in the decoder adds the entropy decoded $VLC(EI_{RB,n}^{Q_s})$ to generate the reconstructed SI-frame $SI_{RB,n}^{r}$, which is given by

$$SI_{RB,n}^{r} = T^{-1}(Q_s^{-1}(EI_{RB,n}^{Q_s} + Q_s(T(Intrapred(SI_{RB,n}^{r}, Intra\_Mode_{RB,n}))))) \quad (4.8)$$

By putting (4.7) into (4.8), we obtain

$$SI_{RB,n}^{r} = T^{-1}(Q_s^{-1}(SP_{FB,n}^{Q_s})) \quad (4.9)$$

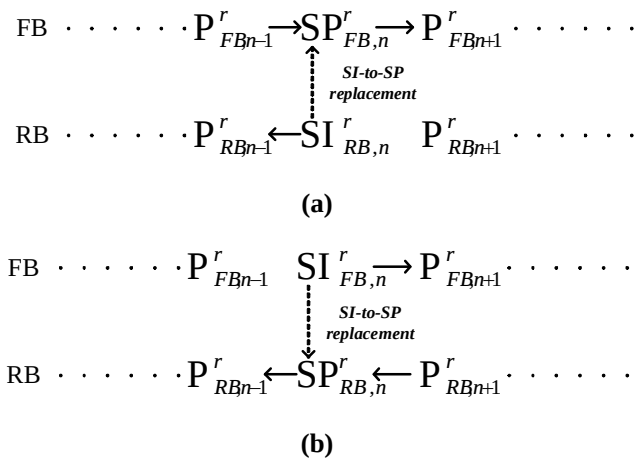


Figure 4.5. SI-to-SP replacement of (a) the SP_{FB}/SI_{RB} pair and (b) the SI_{FB}/SP_{RB} pair.

Equations (4.5) and (4.6) indicate that $SP_{FB,n}^{Q_s}$ is synchronized at *Qs*, (4.9) can be written as

$$SI_{RB,n}^{r} = SP_{FB,n}^{r} \tag{4.10}$$

Now $SI_{RB,n}^{r}$ of the RB is used to replace $SP_{FB,n}^{r}$ of the FB, which is needed during VCR operations. In this case, $SI_{RB,n}^{r}$ is a reference frame for decoding the next P-frame, and such SI-to-SP replacement will not introduced any mismatch error, as shown in Figure 4.5(a).

Figure 4.6 and Figure 4.7 show an example of encoding the SP_{FB}/SI_{RB} pair using a simplified block diagram. In Figure 4.6, all transformed coefficients are represented in bar charts for better illustration of the changes. To encode SP_{FB}, the reconstructed frame $P_{FB,n}^{r}$ is required by encoding the original input frame with a conventional P-frame encoder. Since $P_{FB,n}^{r}$ is not divisible by any quantization factor. It is difficult to find a reverse-encoded frame that is identical to $P_{FB,n}^{r}$. Therefore, $P_{FB,n}^{r}$ is being undergone an additional quantization Q_s to synchronize all transformed coefficients into certain Qs level, which provides a footprint for the reconstruction of SP_{FB}. Obviously, the selection of Q_s would affect the quality of the reconstructed frame $SP_{FB,n}^{r}$, since the quantization Q_s introduces lose. For example, the value of reconstructed coefficient at location 0 is 29, the quantized value becomes 28 after the addition quantization process

with Q_s where 28 is divisible by Qs=4 but the coefficient is degraded with an error of -1. Noted that Q_p is used for coding the residual errors and Q_s is used for generating the switching frame. There exist no direct relationship between quantization Q_p and Q_s.

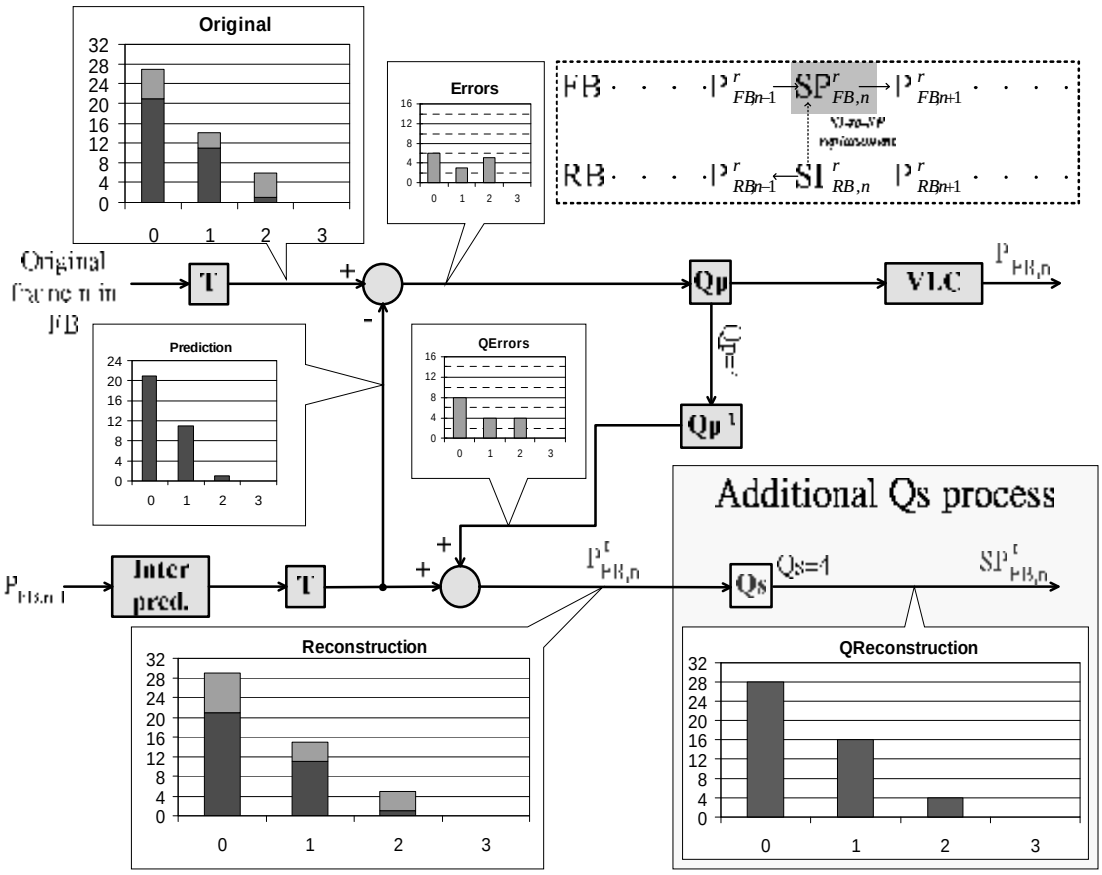


Figure 4.6. An example for encoding SP_{FB}.

In order to find an identical reconstructed reverse-encoded SI_{RB}, the output $SP^r_{FB,n}$ of the SP_{FB} encoder becomes the input of the SI_{RB} encoder as shown in Figure 4.7. To find the difference between the predicted MB and the input MB, the predicted MB should be quantized to Qs domain before performing the

subtraction. The prediction errors between the input MB and the intra-predicted MB at Qs domain are coded by VLC and then transmitted to the client. In Figure 4.7, the loss is only confined in the quantization of the predicted MB, but this loss can be foreseen in the decoder. Also, the previously decoded MBs of $SI_{RB,n}$ at the decoder side is identical to the MBs in the encoder, thus, the quantized predicted MB is also the same as that in the encoder. Meanwhile, the received prediction errors do not introduce any lose with VLD, the reconstructed $SI^r_{FB,n}$ must be identical to $SP^r_{FB,n}$.

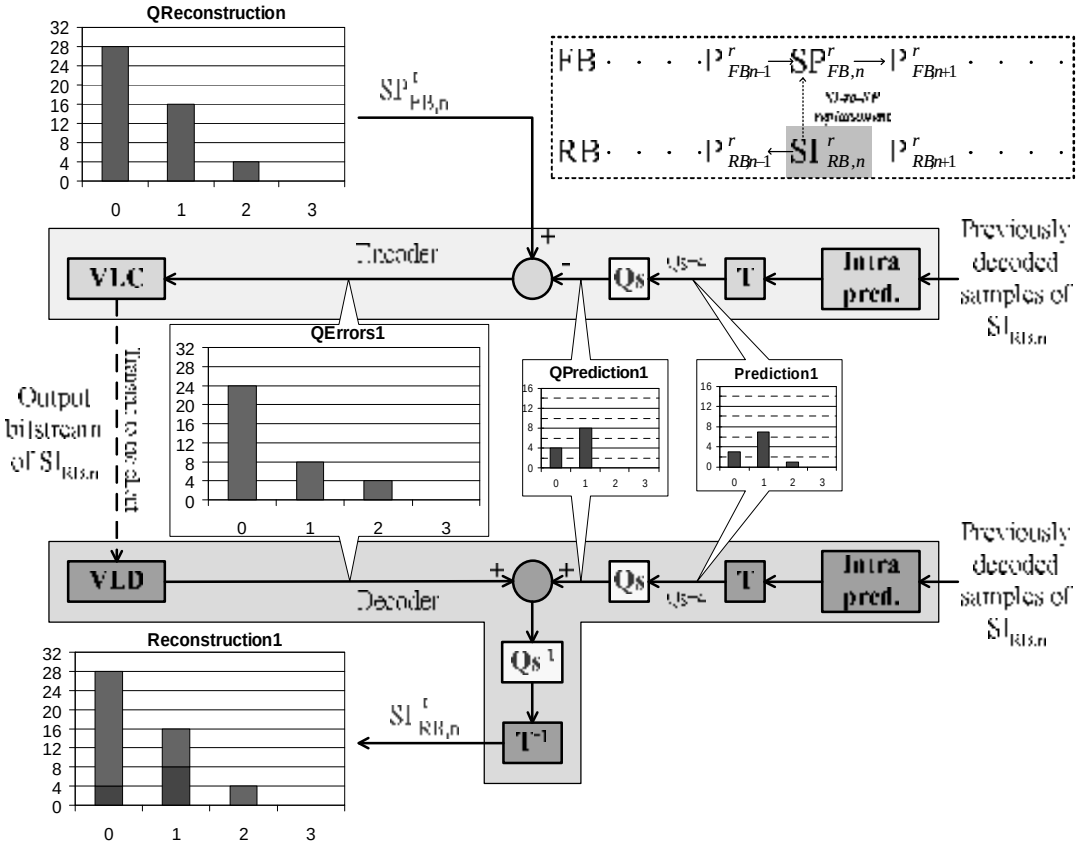


Figure 4.7. An example for encoding and decoding SI_{RB}.

Figure 4.5(b) shows the SI_{FB}/SP_{RB} pair which is used for switching from the FB to the RB at frame *n*. In this case, the reconstructed frame of $SI_{FB,n}$ ($SI^r_{FB,n}$) in the FB is used to replace the reconstructed frame of $SP_{RB,n}$ ($SP^r_{RB,n}$) in the RB. To avoid the problem of reference mismatch, the replacement of $SP^r_{RB,n}$ by $SI^r_{FB,n}$ should not introduce any mismatch error. For encoding $SI_{FB,n}$ of the SI_{FB}/SP_{RB} pair, it is similar to encode $SP_{FB,n}$ and the only difference is that $SI_{FB,n}$ does not use any reference frame. The MBs in $SI_{FB,n}$ are coded by using intra prediction, which means that in the encoding diagram as shown in Figure 4.8(a), the prediction is generated by intra prediction from neighboring blocks. The prediction error between $O_{FB,n}$ and its intra prediction in the quantized transform domain, $EI^{Q_p}_{FB,n}$, is

$$EI^{Q_p}_{FB,n} = Q_p(T(ei_{FB,n})) \tag{4.11}$$

where $ei_{FB,n} = O_{FB,n} - Intrapred(SI^r_{FB,n}, Intra\_Mode_{FB,n})$ and $Intra\_Mode_{FB,n}$ is an intra-prediction modes of the current block. To achieve identical reconstruction, $SI^r_{FB,n}$ should be synchronized at the quantization level *Qs*. To do so, the prediction error $ei^r_{FB,n}$ is firstly reconstructed and add to the prediction again and it can be written as

$$I^r_{FB,n} = ei^r_{FB,n} + Intrapred(SI^r_{FB,n}, Intra\_Mode_{FB,n}) \tag{4.12}$$

where $ei^r_{FB,n} = T^{-1}(Q_p^{-1}(EI^{Q_p}_{FB,n}))$. The reconstructed frame $I^r_{FB,n}$ is then

transformed and quantized with *Qs*. The quantized transform coefficients of $SI^{r}_{FB,n}$ and its reconstructed frame $SI^{r}_{FB,n}$ are

$$SI^{Q_s}_{FB,n} = Q_s(T(I^{r}_{FB,n})) \tag{4.13}$$

and

$$SI^{r}_{FB,n} = T^{-1}(Q_s^{-1}(SI^{Q_s}_{FB,n})) \tag{4.14}$$

respectively.

The lower part of Figure 4.8(a) shows how to encode $SP_{RB,n}$ in the RB. The reference for which it uses is the reconstructed frame at frame *n+1* ($P^{r}_{RB,n+1}$) in the RB and $SI^{Q_s}_{FB,n}$ acts as an input. Hence, the prediction error of $SP_{RB,n}$ in the quantized transform domain, $E^{Q_s}_{RB,n}$, is

$$E^{Q_s}_{RB,n} = SI^{Q_s}_{FB,n} - Q_s(T(Interpred(P^{r}_{RB,n+1}, mv_{RB,n}))) \tag{4.15}$$

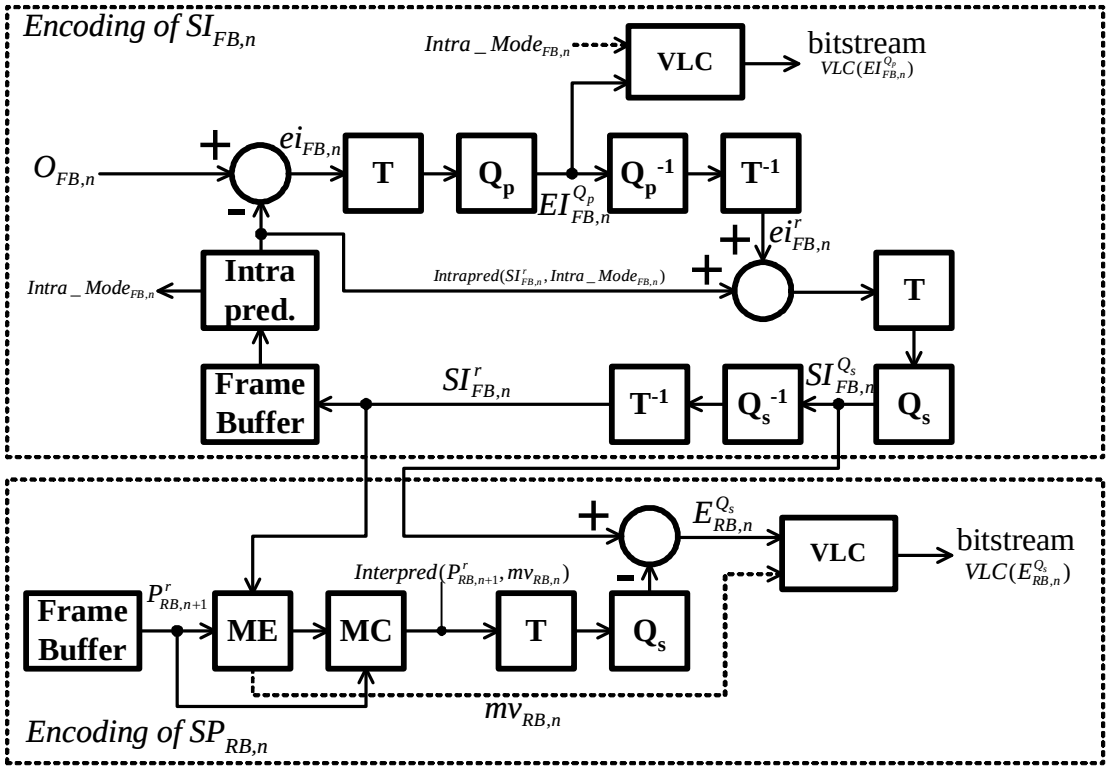
where $mv_{RB,n}$ is the motion vectors of frame *n* by using frame *n+1* of the RB as the reference. This $E^{Q_s}_{RB,n}$ is then entropy encoded as $VLC(E^{Q_s}_{RB,n})$. The decoding processes of SI_{FB}/SP_{RB} pair are depicted in Figure 4.8(b) and Figure 4.8(c) which are quite similar to that of SP_{FB}/SI_{RB} pair. When the replacement of $SI^{r}_{FB,n}$ in the FB for $SP^{r}_{RB,n}$ in the RB is needed, the bitstream $VLC(E^{Q_s}_{RB,n})$ is decoded according to the block diagram in Figure 4.8(c). Then $SP^{r}_{RB,n}$ is reconstructed as

$$SP^{r}_{RB,n} = T^{-1}(Q_s^{-1}(E^{Q_s}_{RB,n} + Q_s(T(Interpred(P^{r}_{RB,n+1}, mv_{RB,n}))))) \tag{4.16}$$

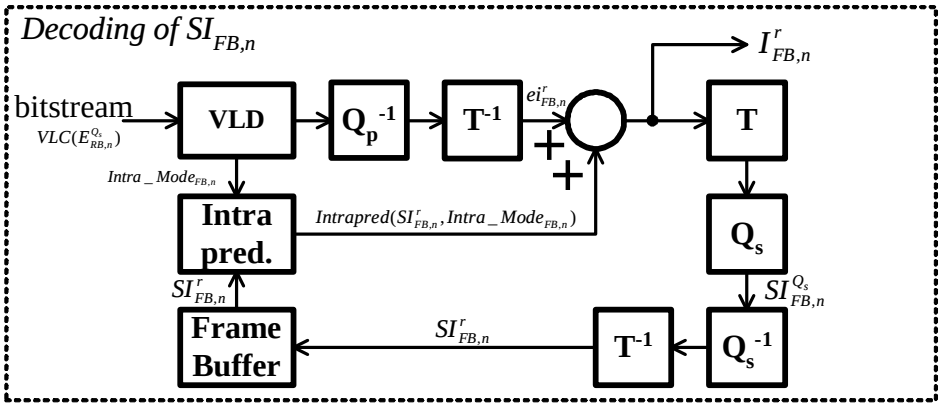
By using (4.13) and (4.15), (4.16) can be further simplified as

$$SP^{r}_{RB,n} = SI^{r}_{FB,n} \tag{4.17}$$

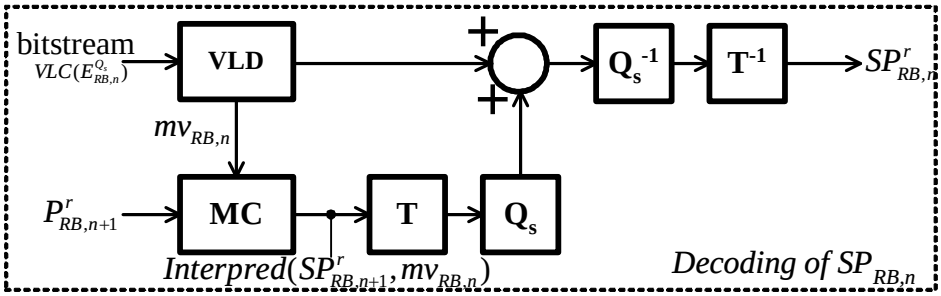
This indicates that $SI^{r}_{FB,n}$ can exactly replace $SP^{r}_{RB,n}$ without any mismatch when SI-to-SP replacement is needed.



(a)



(b)



(c)

Figure 4.8. Encoding and decoding of an SI_{FB}/SP_{RB}. (a) SI_{FB} and SP_{RB} encoding, (b) SI_{FB} decoding and (c) SP_{RB} decoding.

4.3 Experimental Results

In this section, we present some experimental results to evaluate the performance of the proposed dual-bitstream structure. The H.264 encoder [59] was employed to encode various video sequences with different spatial resolutions and motion characteristics. The video sequences used in the simulation are tabulated in Table 4.1. All these sequences have a length of 200 frames. "Carphone", "Claire" and "Grandma" are typical videophone sequences in QCIF (176×144 *pixels*) format, while "Salesman", "Foreman", "Football" and "Table Tennis" are in either CIF (352×288 *pixels*) format or SIF (352×240 *pixels*) format. For generating the original dual bitstreams [42], each test sequence has been encoded into two bitstreams, FB and RB, and I-frames in the RB are interleaved between I-frames in the FB. Note that a total of 52 values of the quantization level are supported by the H.264 standard, indexed

by a Quantization Parameter (QP) which is in the range of 0-51 and the quantization level doubles in size for every increment of 6 in QP. The wide range of the quantization levels makes it possible for an encoder to control the tradeoff accurately and flexibly between bit rate and quality. To encode the FB and the RB with only I-/P-frames, the rate control mechanism of the H.264 encoder was disabled and QP was set to 30 for encoding I- and P-frames. For all test sequences, the frame-rate of the video stream was 30 *frames/s* and the GOP length was fixed to 60 with an I-P structure. Table 4.2 shows the PSNR and the bit rates of the FB and the RB for different sequences when the original structure is used. In order to have a fair comparison between the original and proposed dual-bitstream structures, we encoded the SP_{FB}/SI_{RB} and SI_{FB}/SP_{RB} pairs such that the reconstructed qualities of the new dual bitstreams are similar to those of the original dual bitstreams. We also include in Table 4.2 that the PSNR and the bit rates of different sequences when the new dual bitstreams are encoded with SI-/SP-frames. It can be observed in this table, that the dual-bitstream structure using SP- and SI-frames have lower coding efficiency than the structure using P-/I-frames. As a result, our new structure introduces overhead in the bit rate, but it is not significant, especially in the normal playback. In this situation, only the FB is used and the increase in bit rate is

fewer than 8.1%. Note, however, that the proposed dual-bitstream structure provides better reconstruction quality for various VCR operations as discussed in the following.

Table 4.1. Spatial resolutions and motion characteristics of the testing sequences.

Sequences	Resolutions	Motion characteristics
Salesman	352x288	Low
Football	352x240	High
Table Tennis	352x240	High
Foreman	176x144	High
Carphone	176x144	Low
Claire	176x144	Low
Grandma	176x144	Low

On the other hand, the simplified drift compensation (DC) as shown in Figure 3.5(b) can be used to reduce the drift in the original dual-bitstream system. To do so, a number of drift-compensated frames are introduced at the switching points for switching from FB to RB and vice versa. The storage requirements of the drift-compensated bitstreams actually depend on the selection of the quantization parameters *Qp*. A finer *Qp* leads to lower drift, while increasing the storage for the drift-compensated bitstreams. Otherwise, a coarser *Qp* gives larger drift with a smaller storage requirement. To make a comparison to our proposed dual-bitstream structure, the drift-compensated

bitstreams are encoded with the quantization factor that generates similar storage requirement to our new dual bitstreams. In Table 4.2, it also shows the storage increases for the original dual-bitstream structure with DC.

Table 4.2. Average PSNR and bitrate comparisons for the original and proposed dual-bitstream structures in the video streaming system.

Sequences	Streams	Original structure		Proposed structure		ΔPSNR (dB)	Percentage increase in bitrate	
		PSNR (dB)	Bitrate (kbits/s)	PSNR (dB)	Bitrate (kbits/s)		Original Scheme with DC	Proposed Scheme
Salesman	FB	34.658	156.049	34.564	167.702	-0.094	10.99%	7.47%
	RB	34.690	157.567	34.566	179.700	-0.125	15.03%	14.05%
Foreman	FB	35.838	331.933	35.820	348.787	-0.017	7.27%	5.08%
	RB	35.796	336.050	35.775	364.472	-0.021	9.53%	8.46%
Football	FB	33.461	946.258	33.459	961.187	-0.002	2.13%	1.58%
	RB	33.432	938.954	33.433	967.981	0.001	3.12%	3.09%
Table Tennis	FB	32.824	528.680	32.805	545.380	-0.020	4.33%	3.16%
	RB	32.644	530.878	32.709	564.649	0.065	7.49%	6.36%
Claire	FB	38.419	31.249	38.410	32.767	-0.009	5.84%	4.86%
	RB	38.423	30.443	38.402	34.144	-0.022	12.95%	12.16%
Carphone	FB	35.537	93.846	35.524	97.141	-0.013	3.77%	3.51%
	RB	35.534	93.469	35.500	103.368	-0.034	13.74%	10.59%
Grandma	FB	35.166	34.045	35.021	36.775	-0.145	11.61%	8.02%
	RB	35.125	34.301	34.972	40.076	-0.153	22.08%	16.84%

Table 4.3 gives our experimental results on the average PSNR for different structures on all possible switching points between the FB and the RB. This table also shows that our new dual-bitstream system introduces increase in the total storage requirement (Δbits) since the coding efficiency of SP/SI-frames

has lower than that of P/I-frames. But it is still smaller that of the original scheme with drift compensation (DC). In Table 4.3, It is clear that the original dual-bitstream structure introduces quality degradation as compared with the quality of the original FB or RB. This is due to the fact that an I-frame of the FB (RB) is used to approximate a P-frame of the RB (FB) at the moment of fast forward/reverse operations in the original structure without DC and this approximation leads to reference mismatch which affects the reconstructed quality of the next requested frame. However, the switching points of the proposed dual-bitstream structure are at SI_{FB}/SP_{RB} and SP_{FB}/SI_{RB} pairs. These pairs can ensure identical frame reconstruction and no approximation is required. This means the reconstruction quality during the SI-to-SP replacement is the same as that of the FB or RB. The advantage of the proposed structure is indicated in the experimental results as shown in Table 4.3. This table shows that our proposed structure has significant improvement for all video sequences. The results are more noticeable for the sequences “Carphone” and “Foreman” which has PSNR improvement over 1dB. This table also shows that our proposed dual-bitstream structure has significant improvement over original one with DC for all video sequences.

Table 4.3. Average PSNR performance for the original and proposed dual-bitstream structures on every possible switching between the FB and RB.

	Original structure		Original structure with DC			Proposed structure		
Sequences	**FB→RB**	**RB→FB**	**FB→RB**	**RB→FB**	**Δbits**	**FB→RB**	**RB→FB**	**Δbits**
Salesman	33.997	34.161	34.534	34.654	13.02%	34.441	34.660	10.77%
Foreman	34.475	34.686	35.469	35.587	8.41%	35.733	35.784	6.78%
Football	33.107	33.212	33.372	33.649	2.62%	33.850	33.916	2.33%
Table Tennis	32.176	32.335	32.524	32.816	5.92%	32.894	32.988	4.76%
Claire	37.516	37.324	37.985	38.269	9.35%	38.175	38.587	8.46%
Carphone	34.218	34.583	35.112	35.500	8.74%	35.536	35.642	7.04%
Grandma	34.276	34.451	34.904	35.022	16.86%	34.772	34.672	12.44%

Furthermore, for the original dual-bitstream structure, the quality degradation will not only be confined to the frame at the switching point but can propagate and be accumulated in the subsequent P frames. Figure 4.9 shows the effect of drift using the original structure for the “Foreman” sequence in the situation where the current VCR is in the fast-reverse mode and subsequently a normal-play request is launched. Assume that the user requests a normal-play request at the start of each GOP in the RB. In this case, it has the longest drift propagation in the original dual-bitstream structure due to the I-to-P approximation at the switching point. This figure shows that the drift caused by the I-to-P approximation is very serious and lasts until the next I-frame in the FB. Although the scheme with DC can reduce drift propagation, it still happens. On the other hand, Figure 4.10 shows the performance of the

proposed structure with SI/SP-frames. It outperforms the original one and follows the same PSNR of the FB. This result indicates that the visual quality of the reconstructed frames in the proposed dual bitstreams for various VCR operations is exactly identical to that of the normal playback.

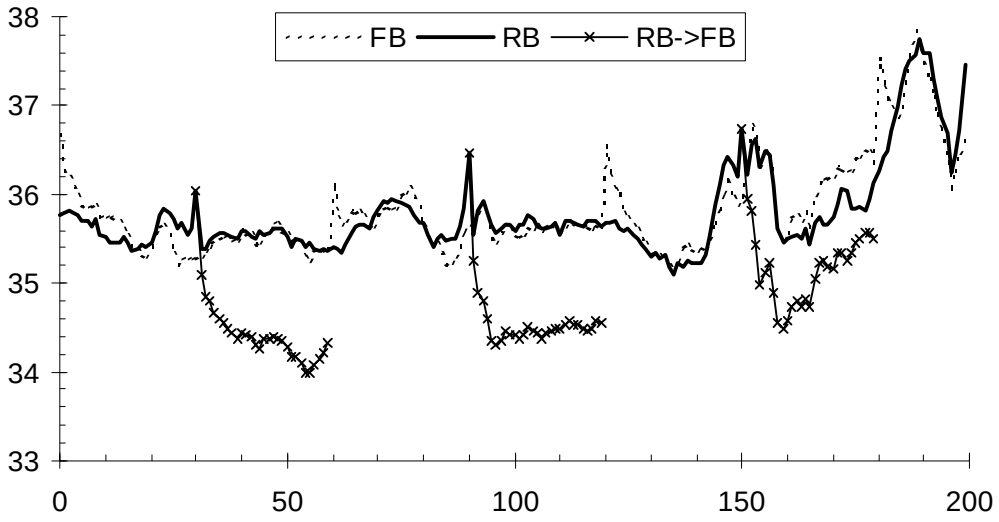


Figure 4.9. PSNR performances of the original dual bitstreams in the situation where the current VCR is in the fast-reverse mode and then a normal-play request is launched at the start of each GOP in the RB for the "Foreman" sequence.

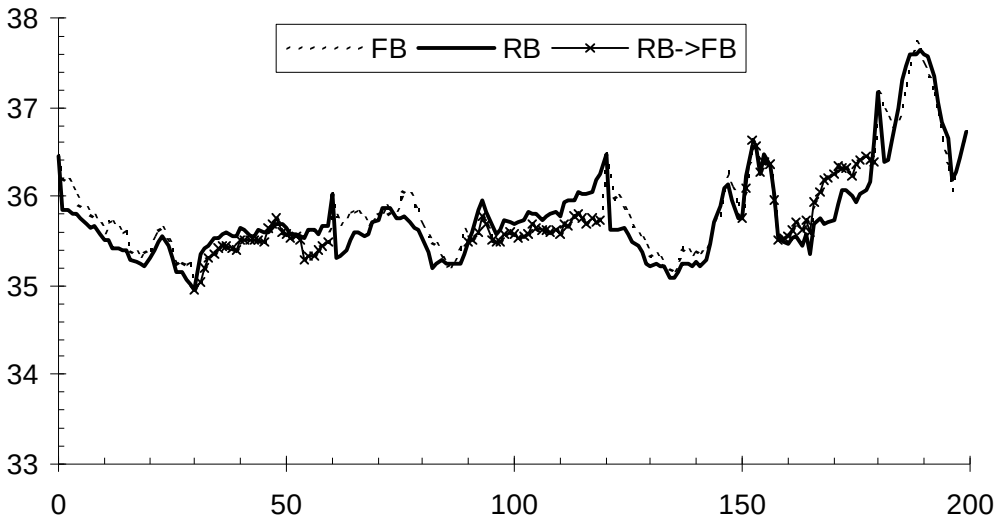


Figure 4.10. PSNR performances of the proposed dual bitstreams in the situation where the current VCR is in the fast-reverse mode and then a normal-play request is launched at the start of each GOP in the RB for the "Foreman" sequence.

4.4 Chapter Summary

In this chapter, we have proposed to adopt the concept of SP/SI-frame coding to improve the performance of the dual-bitstream structure with only I-/P-frames, which has been proven to be an efficient approach for providing VCR functionality in video streaming. The streaming system with the original dual bitstreams stores a forward-encoded bitstream (FB) and a reverse-encoded bitstream (RB) in the server in order to reduce the requirement of network bandwidth significantly. However, the mismatch between the encoding of the FB and RB happens and it arises the problem of drift when the I-to-P approximation is used for fast-forward and fast-reverse

playbacks. Therefore, our novel dual-bitstream structure is aimed at avoiding the problem of reference mismatch when a frame in the RB (FB) is selected as a reference for a frame in the FB (RB). The proposed structure modifies the dual bitstreams using the main feature of SP- and SI-frames which can provide an identical reconstruction even when different reference frames are used for prediction. To utilize this property, SP_{FB}/SI_{RB} and SI_{FB}/SP_{RB} pairs are added in the structure of dual bitstreams and this arrangement ensures that the I-to-P approximation will not happen in our dual-bitstream structure for various VCR operations. Consequently, the video streaming system adopting the new dual bitstreams will not cause any mismatch at the decoder side and the quality of reconstructed frames can be maintained. Besides, we have also shown how SP_{FB}/SI_{RB} and SI_{FB}/SP_{RB} pairs can be encoded such that identical frames can be reconstructed even when an SI-frame of one bitstream is used to replace an SP-frame of the other bitstream in any VCR operation. Experimental results show that the visual quality of all reconstructed frames in our new dual-bitstream structure for various VCR operations is exactly the same as that of normal playback. Therefore, the proposed structure with SP- and SI-frames is a promising solution for H.264 video streaming with VCR functionality.

Chapter 5 – Macroblock-based Algorithm for Dual-bitstream MPEG Video Streaming with VCR Functionalities

5.1 Introduction

The new arrangement of dual bitstreams proposed in Chapter 4 would affect the visual quality of normal forward playback in video streaming. Since the additional quantization Q_s introduces quality degradation to I/P pair of the FB. It is also the only disadvantage of this proposed method. In this chapter, we propose an alternative of performing bitstream switching in the dual-bitstream system in order to reduce the problem arising from the mismatch between the FB and RB. By using the motion information of the compressed bitstream, we propose a novel macroblock-selection scheme at the server to enhance the quality of the reconstructed frame during bitstream switching. The proposed scheme can adaptively select the necessary macroblocks (MBs), manipulate them in the compressed domain and send the processed MBs to the client. 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.

5.2 Macroblock-based Algorithm for Bitstream Switching in the Dual-bitstream System

In this chapter, we consider a MB-based solution to reduce the problem of reference mismatch at bitstream switching. As discussed in Chapter 3, the dual-bitstream system requires a P-to-P approximation when the FB is switched to the RB or the RB is switched to the FB with a small speed-up factor. In the following, we provide a detailed description and formulation of the proposed MB-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 in Figure 3.1 again in which the current VCR is in the normal forward play mode and then a backward-play request is launched. The situation in MB level is depicted in Figure 5.1. In the server, $MB_{(k,l)}^{FB_n}$ and $MB_{(k,l)}^{RB_n}$ represent the reconstructed MBs 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.

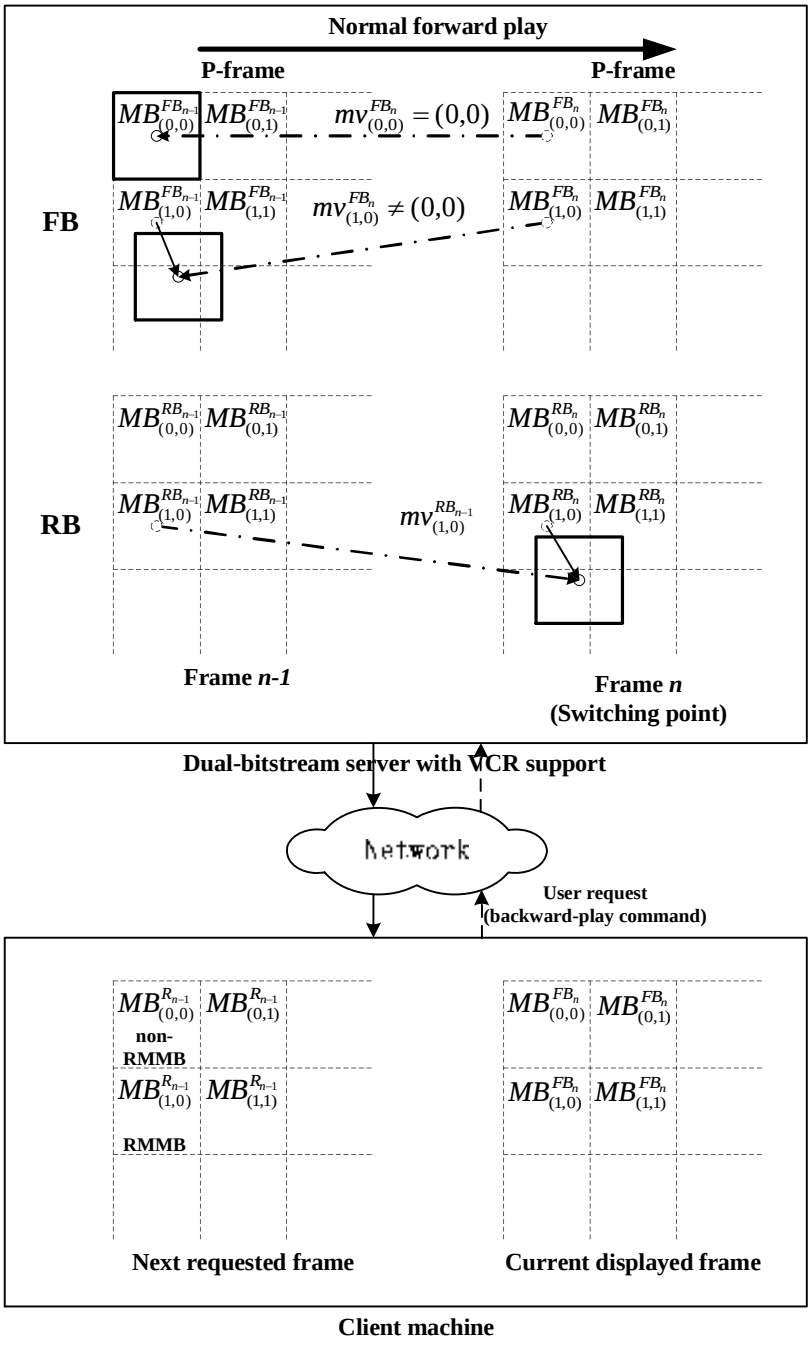


Figure 5.1. Definition of the non-RMMB and RMMB.

5.2.1 MB Viewpoint of the Dual-bitstream System

In MPEG video coding standards, block motion-compensated prediction (MCP) is used to reduce the temporal redundancy in video sources [4-9]. 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. During our discussion, we consider all $MB_{(k,l)}^{FB_n}$ as a coded-and-reconstructed signal. It is because the FB and the RB are already encoded and stored in the server, which means that the original signal is unavailable in both of the server and client sides. Therefore, the prediction errors in FB, $e_{(k,l)}^{FB_n}$, and RB, $e_{(k,l)}^{RB_{n-1}}$, are given by

$$e_{(k,l)}^{FB_n} = MB_{(k,l)}^{FB_n} - MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n}) \tag{5.1}$$

and

$$e_{(k,l)}^{RB_{n-1}} = MB_{(k,l)}^{RB_{n-1}} - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}}) \tag{5.2}$$

where $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$ stands for the motion-compensated MB 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 MB 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 reverse order. All

these prediction errors are transformed in the DCT domain. The transformed DCT coefficients are then quantized, variable-length encoded and stored in the server.

Figure 5.1 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 the frame buffer at the client machine as it is used for decoding the subsequent frame, frame *n+1*, in the forward play operation. In other words, all $MB_{(k,l)}^{FB_n}$ are available at the decoder. When MBs frame *n-1* are 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 decodes the coded prediction error by using the variable-length decoder which outputs 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 MB of frame *n-1*, named $MB_{(k,l)}^{R_{n-1}}$, can be reconstructed by adding the prediction which results from the previously decoded frame by applying motion compensation, as indicated below,

$$MB_{(k,l)}^{R_{n-1}} = MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) + e_{(k,l)}^{RB_{n-1}} \quad (5.3)$$

where $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}})$ is the motion-compensated MB 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 of the decoder. Note that, for $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}})$, the reference frame comes from the FB whereas the motion vector $mv_{(k,l)}^{RB_{n-1}}$ is extracted from the RB. Substitution of (5.2) into (5.3) yields

$$MB_{(k,l)}^{R_{n-1}} = MB_{(k,l)}^{RB_{n-1}} + [MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})] \quad (5.4)$$

This equation implies that the reconstructed MB 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 an amount of $MCMB^{FB_n}(mv_{(k,l)}^{RB_{n-1}}) - MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})$. This term indicates the drift 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 MBs in frame *n-1* of the RB. Such drift could propagate and be accumulated in the subsequent P-frames.

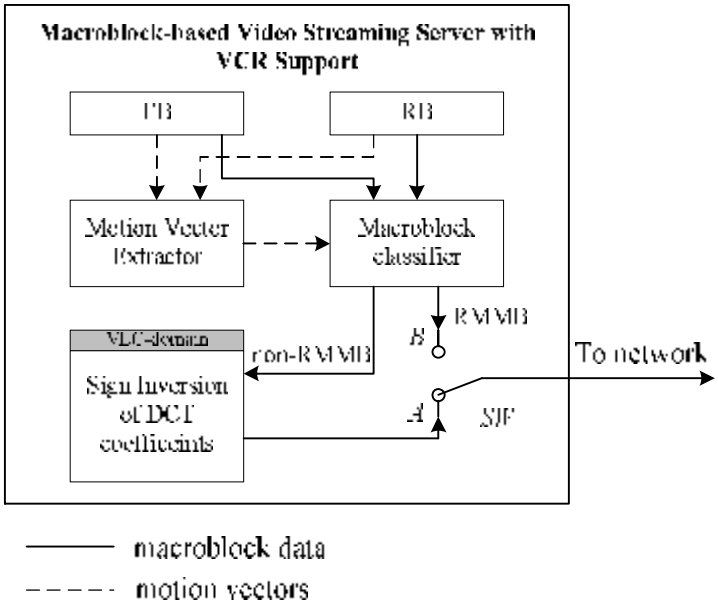


Figure 5.2. The proposed architecture for the video streaming system with VCR functionality.

5.2.2 Classification of MBs at the Switching Point

In order to reduce such drift, in contrast to the frame-based selection scheme used in the original dual-bitstream architecture, a MB-based selection scheme is adopted at the switching point. Various types of MBs are handled differently. The MB classifier in the proposed server is shown in Figure 5.2. 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 input to a MB classifier for identifying the types of MBs. Two types of MBs are now defined. For illustration, we use the example in Figure 5.1 again to give a clearer account for defining MBs in the requested frame, 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 reference mismatched MB (RMMB) if the MB in frame *n* of the FB, $MB_{(k,l)}^{FB_n}$, $MB_{(k,l)}^{R_{n-1}}$ having a motion compensated MB, $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$, with different spatial position of $MB_{(k,l)}^{R_{n-1}}$. Otherwise, it is defined as a non-RMMB. In non-RMMBs, reference mismatch does not exist and the reason will be discussed in the following section. For example, in Figure 5.1, 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 zero MV MB and $MB_{(0,0)}^{FB_{n-1}}$ is classified as a non-RMMB. On the other hand, since $mv_{(1,0)}^{FB_n}$ is a non-zero MV, $MB_{(1,0)}^{R_{n-1}}$ is classified as a RMMB. In this chapter, our proposed algorithm works at the level of MBs. The server processes those RMMBs as same as the original dual-bitstream system, the switch *SW* is connected to position B. The server gets the data from the RB and the drift problem mentioned in (5.4) cannot be avoided. As it will be described in details, for non-RMMBs, the server uses only the MPEG data from the FB to reconstruct the requested MB in order to avoid the reference mismatch. Note that the server processes each non-RMMB in the compressed domain such that a complete decoding and encoding are not required at the server.

5.2.3 Sign Inversion Technique for non-RMMBs

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 non-RMMB, we are interested in obtaining $MB_{(k,l)}^{R_{n-1}}$ from the FB. If this can be done, no drift will occur since both reference frame and the prediction error come from the FB. We now present in details how a non-RMMB can be reconstructed from the FB provided that frame *n* of the FB is available at the decoder. Rearranging (5.1), we obtain an expression for $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$

$$MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n}) = MB_{(k,l)}^{FB_n} - e_{(k,l)}^{FB_n} \tag{5.5}$$

When the requested $MB_{(k,l)}^{R_{n-1}}$ is found to be a non-RMMB, its corresponding MB in frame *n* of the FB, $MB_{(k,l)}^{FB_n}$, is coded with zero MV. 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

$$MB_{(k,l)}^{FB_{n-1}} = MB_{(k,l)}^{FB_n} + \tilde{e}_{(k,l)}^{FB_n} \tag{5.6}$$

where $\tilde{e}_{(k,l)}^{FB_n} = -e_{(k,l)}^{FB_n}$. Note that the client machine has 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. In order to reconstruct $MB_{(k,l)}^{R_{n-1}}$, which is identical to $MB_{(k,l)}^{FB_{n-1}}$, in the backward-play operation, (5.6) indicates that, for a

non-RMMB, the only data that the server needs to send is the quantized DCT coefficients of $\tilde{e}_{(k,l)}^{FB_n}$. In the following discussions, we describe how to compute these quantized DCT coefficients of $\tilde{e}_{(k,l)}^{FB_n}$ from the existing MPEG video stream in the server.

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

$$DCT(\tilde{e}_{(k,l)}^{FB_n}) = -DCT(e_{(k,l)}^{FB_n}) \tag{5.7}$$

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

$$Q[DCT(\tilde{e}_{(k,l)}^{FB_n})] = -Q[DCT(e_{(k,l)}^{FB_n})] \tag{5.8}$$

From (5.8), $Q[DCT(\tilde{e}_{(k,l)}^{FB_n})]$ 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. $Q[DCT(\tilde{e}_{(k,l)}^{FB_n})]$ is then transmitted to the client by switching *SW* to position A. From the above derivation, we can conclude that the server only needs to invert the sign of all DCT coefficients for each non-RMMB 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 MBs classified as non-RMMBs. Since both the reference frame and the inverted DCT coefficients are from the FB, the reconstructed pixels of the non-RMMB are identical to corresponding

pixels of the MB in the FB. The problem of reference mismatch can be avoided and no drift is introduced in non-RMMBs.

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 [24-26], as demonstrated by some typical examples as shown in Table 5.1 which shows the distribution of non-RMMB for various sequences, including “Claire”, “Grandma”, “Salesman”, “Carphone”, “Table Tennis” “Foreman”, and “Football”. These sequences have been selected to emphasize different amount of motion activities. It is clear that over 90% and 39% of the MBs are non-RMMBs for Claire and Football sequences respectively. By inverting the sign of all DCT coefficients in the server, the sequence containing more non-RMMBs can alleviate the mismatch significantly.

Table 5.1. Percentage of non-RMMB for various sequences.

Claire	Grandma	Salesman	Carphone	Table Tennis	Foreman	Football
91.33%	86.78%	64.20%	59.27%	51.93%	45.03%	39.40%

Consider the same example of backward playback. 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 algorithm can be processed in a recursive way to reduce the drift problem. The MB in frame *n-2*, $MB_{(k,l)}^{R_{n-2}}$, is treated as non-RMMB when $mv_{(k,l)}^{FB_{n-1}} = (0,0)$ and its spatial corresponding $MB_{(k,l)}^{R_{n-1}}$ is also a non-RMMB. Figure 5.3 shows a scenario when frame *n-2* is requested. In Figure 5.3, $MB_{(0,0)}^{R_{n-2}}$ is a non-RMMB since $mv_{(0,0)}^{FB_{n-1}}$ is equal to zero and $MB_{(0,0)}^{R_{n-1}}$ is classified as a non-RMMB. Even though $mv_{(1,0)}^{FB_{n-1}}$ is also equal to zero, $MB_{(1,0)}^{R_{n-2}}$ is still treated as a RMMB. The reason behind is that $MB_{(1,0)}^{R_{n-1}}$ is a RMMB in which the drift has already introduced in this reconstructed MB. It is useless to employ the technique of sign inversion of DCT coefficients in $MB_{(1,0)}^{R_{n-2}}$.

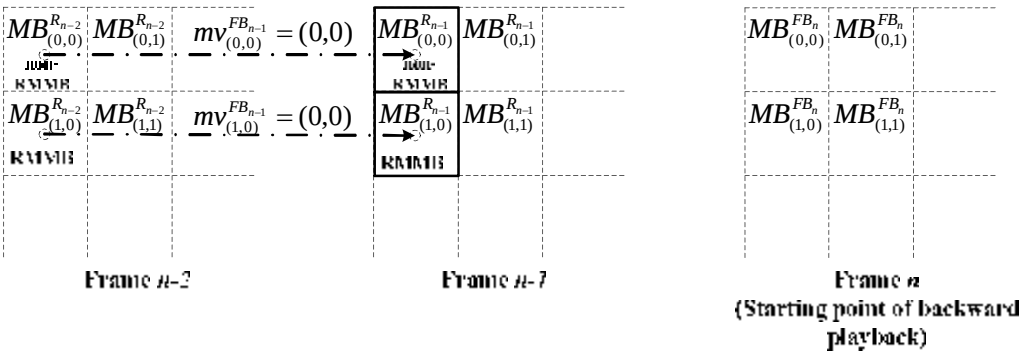


Figure 5.3. Definition of the non-RMMB and RMMB at frame *n-2* when a user requests backward playback is at frame *n*.

5.2.4 VLC-domain Technique for Sign Inversion in non-RMMBs

In the server, the sign inversion of DCT coefficients requires additional

variable length decoding and re-encoding. To reduce the computational load of the server, we propose to compute the newly quantized DCT coefficients $Q[DCT(\tilde{e}_{(k,l)}^{FB_n})]$ in the VLC domain. In MPEG video encoding, DCT coefficients representing high spatial frequencies are almost always zero, whereas low-frequency coefficients are often non-zero. To exploit this behavior, the DCT coefficients are arranged qualitatively from low to high spatial frequency following the zig-zag scan order. This zig-zag scan approximately orders the coefficients according to their probability of being zero. With zig-zag ordering, many DCT coefficients are zero in a typical 8×8 block. In this case, better coding efficiency is obtained when codewords are defined by combining the length of the zero coefficient run with the amplitude of the nonzero coefficient terminating the run.

Each nonzero sequence of DCT coefficients is then coded in the LAST-RUN-LEVEL symbol structure with different variable length codes (VLCs). LAST indicates the last nonzero coefficient of the zig-zag scan order; RUN refers to the number of zero coefficients before the next nonzero coefficient; LEVEL refers to the amplitude of the nonzero coefficient. Table 5.2 illustrates this. The trailing bit of each VLC is the '*s*' bit that codes the sign of the nonzero coefficient. If '*s*' is 0, the coefficient is positive; otherwise it is negative. To

convert $Q[DCT(\tilde{e}_{(k,l)}^{FB_n})]$ from $Q[DCT(e_{(k,l)}^{FB_n})]$, the server just parses the FB and inverts all '*s*' bits of VLCs in each non-RMMB, as shown in Figure 5.4. On the other hand, LAST-RUN-LEVEL combinations that are not in the Table 5.2 are coded using a 6-bit "Escape" code followed by a 1-bit flag for LAST, 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 5.3. In this case, the 12-bit FLC for LEVEL is converted into its 2's complement. The bit manipulation of VLCs in non-RMMB is summarized in Figure 5.4. 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.

Table 5.2. VLC table for RUN-LEVEL combinations. The sign bit 's' is '0' for positive and '1' for negative.

Variable length codes	Last	Run	Level
10 s	0	0	1
:		:	:
0000 0100 000 s	0	0	12
110 s	0	1	1
: :		: :	: :
0000 0101 0111 s	0	26	1
0111 s	1	0	1
:		:	:
0011 11 s	1	1	1

: :		: :	: :
0000 01011111s	1	40	1
0000 011	Escape		

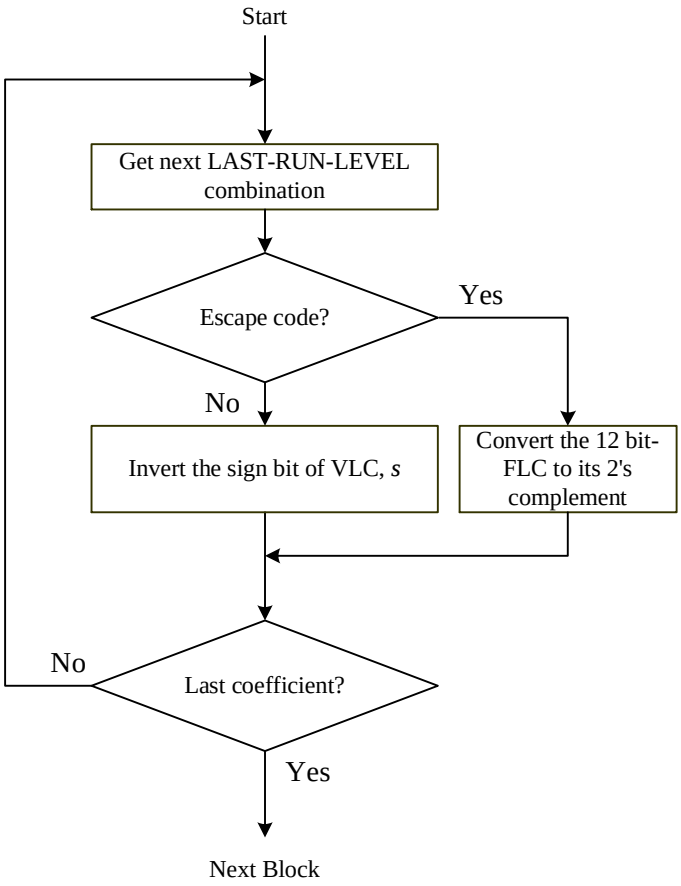


Figure 5.4. Execution flow of the server during bit manipulation of VLCs in non-RMMB.

Table 5.3. FLC table for RUNS and LEVELS. It is used following the escape code of a VLC.

Fixed Length codes	Run
0000 00	0
0000 01	1
0000 10	2
: :	: :
:	:

Fixed Length codes	Signed_level
1000 0000 0001	-2047
1000 0000 0010	-2046
:	:
1111 1111 1111	-1
0000 0000 0000	Forbidden

:	:		
:	:	0000 0000 0001	+1
:	:		
:	:	:	:
:	:	:	:
1111 11	63	0111 1111 1111	+2047

5.3 Experimental Results

A large amount of experimental works has been conducted to evaluate the performance of the proposed MB-based algorithm when applied to the dual-bitstream streaming system [42] with VCR support. MPEG-4 encoder [58] was used in this simulation. Various video sequences, as tabulated in Table 4.1, with different spatial resolutions and motion characteristics were tested. The frame rate of these sequences was 30 frames/s with the GOP length of 14. All the sequences were encoded at two different bitrates. Each test sequence has been encoded into two bitstreams, FB and RB, and I-frames in the RB are interleaved between I-frames in the FB. In this section, comparisons of the original dual-bitstream system and the proposed MB-based dual-bitstream system are provided to illustrate their performances.

For the original dual-bitstream system, P-frames of the FB may be used to approximate P-frames of the RB at the moment of a backward-play operation requested by a user. For example, if the user requests a backward-play operation at frame *n*. Since frame *n* always has a short distance (d_c) to frame

n-1, bitstream switching is required at the switching point *n*. This approximation, however, will lead to reference mismatch which affects the reconstruction quality of the next requested frame, frame *n-1*, as illustrated in Figure 5.5. In this figure, the PSNR comparison between the system with and without using the proposed MB-based algorithm for the “Salesman” sequence encoded at 3Mb/s at all possible switching points are simulated. The PSNR of the frame at the switching point is degraded seriously, about 2dB to 3dB, of the original system. As shown in Figure 5.5, when the server performs a P-to-P or P-to-I approximation by using the proposed MB-based algorithm, there is around 1dB improvement. Table 5.4 shows the average PSNR of all possible switching points. We show that the proposed MB-based dual-bitstream system outperforms the original one in all sequences. Especially, the proposed algorithm gains up to 1.34dB in the “Claire” sequence since over 90% MBs are classified as non-RMMBs. The results are more significant for non-RMMBs as shown in Table 5.5 in which the detailed comparisons of the average PSNR for non-RMMBs only are tabulated. This result is expected since a sign inversion technique should not introduce any drift to non-RMMBs. On average, the PSNR performance of non-RMMBs improves 1.7dB as compared with the original dual-bitstream system.

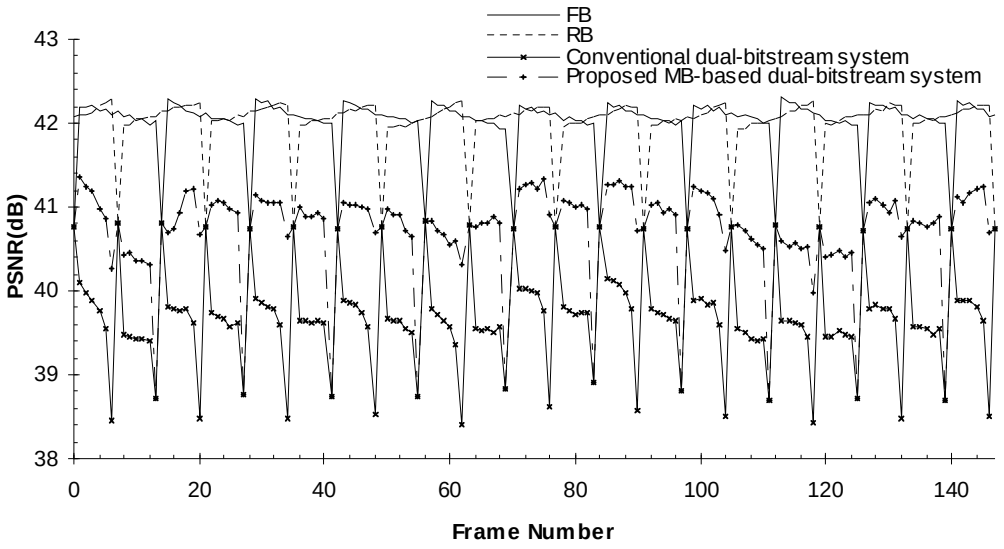


Figure 5.5. PSNR performances of the original system and the proposed MB-based system for the "Salesman" sequence encoded at 3.0 Mb/s due to reference mismatch at all possible switching points from the FB to RB at the moment of a backward-play operation requested by a user.

Table 5.4. Overall average PSNR of all possible switching points.

Sequences	Bitrate	Original dual-bitstream system	MB-based dual-bitstream system	Gain
Salesman	1.5M	36.62	37.58	0.96
	3M	39.58	40.87	1.29
Football	1.5M	28.91	29.40	0.49
	3M	32.58	33.45	0.87
Table Tennis	1.5M	31.97	32.43	0.46
	3M	36.71	37.94	1.22
Foreman	64K	25.64	25.88	0.24
	128K	27.55	28.11	0.56
Carphone	64K	28.70	29.05	0.36
	128K	31.71	32.34	0.63
Claire	64K	34.08	34.81	0.73
	128K	39.65	40.99	1.34
Grandma	64K	32.11	32.48	0.37
	128K	35.10	35.82	0.72

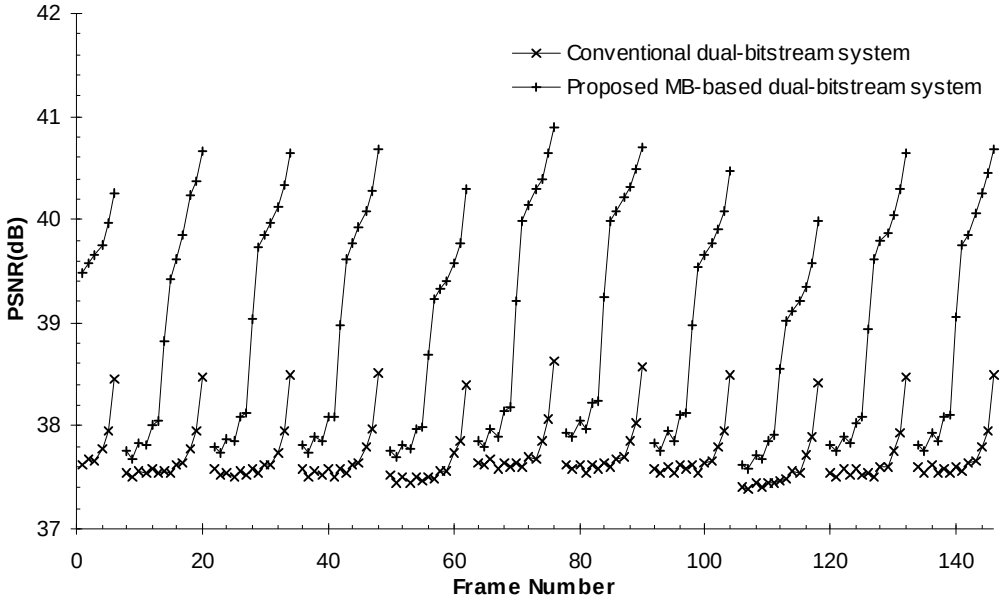
Table 5.5. Average PSNR of all possible switching points for non-RMMBs.

Sequences	Bitrate	Original dual-bitstream system	MB-based dual-bitstream system	Gain
Salesman	1.5M	36.54	38.10	1.56
	3M	39.58	41.80	2.22
Football	1.5M	30.08	31.69	1.61
	3M	33.36	36.06	2.71
Table Tennis	1.5M	32.06	33.41	1.35
	3M	36.85	39.93	3.08
Foreman	64K	25.19	25.78	0.58
	128K	27.20	28.59	1.39
Carphone	64K	28.42	29.04	0.62
	128K	31.49	32.61	1.12
Claire	64K	32.51	35.08	2.57
	128K	37.81	41.34	3.52
Grandma	64K	31.65	32.04	0.39
	128K	34.65	35.42	0.77

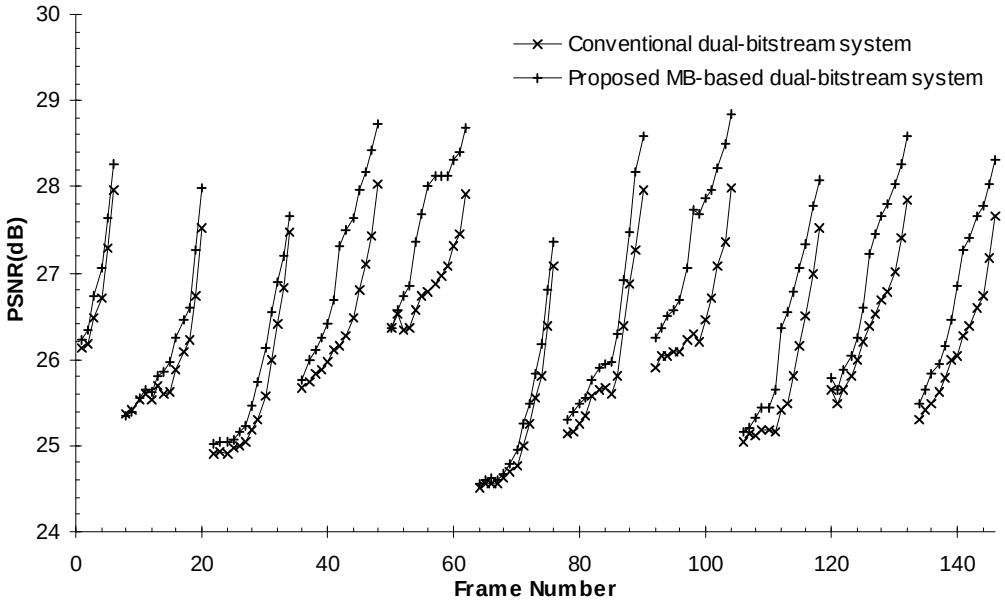
Furthermore, the quality degradation will not only be confined to the frame at the switching point but can propagate and be accumulated in the subsequent P-frames. Such drift will last until the next I-frame in the RB. In Figure 5.6, we have realized the effect of drift by using the original algorithm and our proposed MB-based system for “Salesman”, “Foreman” and “Football” sequences which have different levels of motion activities. In order to show the worst situation due to drift for both algorithms, the longest propagation of drift has been simulated in which the switching point occurs at the start of each GOP in the RB. For example, as shown in Figure 5.7, a user requests a backward-play

operation at frame 21. In this case, the switching point is at frame 21, the drift problem caused by P-to-I approximation will propagate for reconstructing frames within the GOP in the RB, frames 20 to 8, during backward playback. In Figure 5.6(a), it can be seen that the proposed system has a remarkable PSNR improvement over the original one for the “Salesman” sequence encoded at 3.0 Mb/s. It is due to the reason that the “Salesman” sequence contains more non-RMMBs in which the technique of sign inversion can be employed and drift can be reduced. Besides, the PSNR of the requested frames after bitstream switching drops gradually as the distance from the switching point increases. It is because the number of RMMBs increases and the drift could be accumulated. Note that motion-compensated prediction is not used in I-frames and it means that there is no inter-frame dependency between the last frame of the GOP and the first frame of the next GOP of the FB. In this case, no non-RMMB exists in the last frame of the GOP in the FB. As a consequence, the non-RMMB does not exist in I-frames. As shown in Figure 5.6(a), the PSNR of frames 13, 27, 41, etc drops dramatically since frames 14, 28 and 42 of the FB are intra-coded. All MBs of the subsequent frames that follow an I-frame of the FB in reverse order are defined as RMMBs. Thus, the technique of sign inversion cannot be applied in these frames. Let us use the

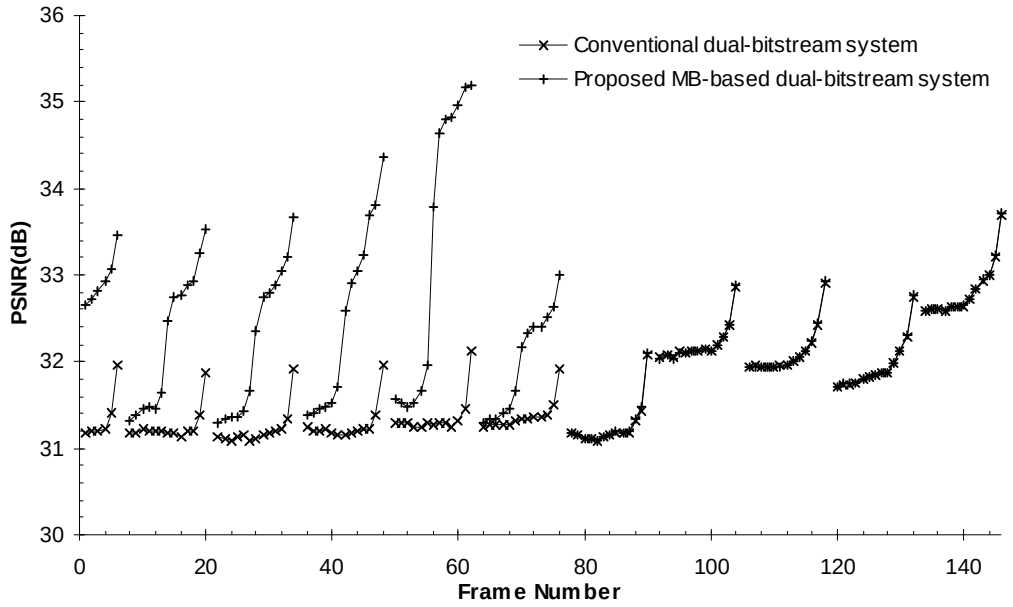
example in Figure 5.7 again, in which a sign inversion technique can be applied from frames 20 to 14 (the first segment) and cannot be employed from frames 13 to 8 (the second segment). Table 5.6 further shows the average PSNR of the first segment of both algorithms. We see that the proposed sign inversion technique can greatly reduce the drift. For the second segment in which the non-RMMB does not exist, the average PSNR of the proposed dual-bitstream system still improves due to the good quality of the last frame (in the RB) in the first segment. For sequences containing high motion activities such as “Football” and “Foreman”, the proposed algorithm still has a small PSNR improvement, as shown in Figure 5.6(b), Figure 5.6(c), Table 5.6 and Table 5.7. This further demonstrates the effect of the proposed MB-based algorithm when applied to the MPEG video system with VCR functionality.



(a) the "Salesman" sequence encoded at 3.0 Mb/s



(b) the "Foreman" sequence encoded at 128 Kb/s



(c) the "Football" sequence encoded at 3.0 Mb/s

Figure 5.6. The PSNR performances of the original dual-bitstream system and the proposed MB-based dual-bitstream system in the case when a user requests a forward-play operation to a backward-play operation at the start of each GOP in the RB until the next I-frame.

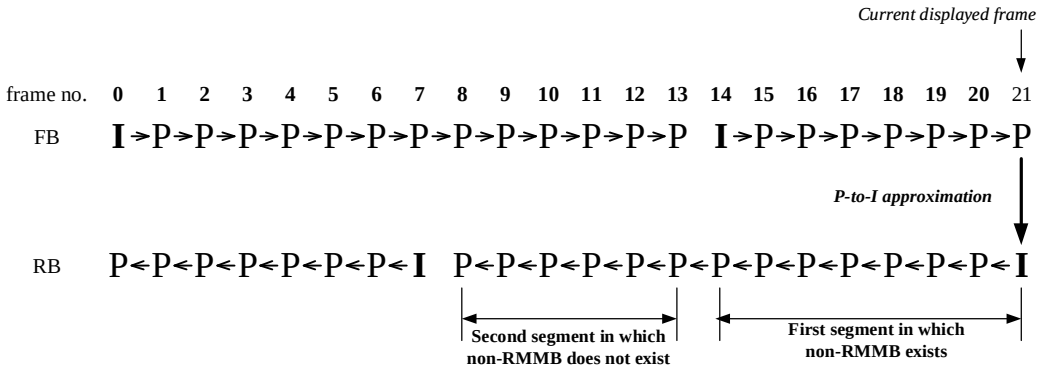


Figure 5.7. A user requests a backward-play operation at frame 21 in which a sign inversion technique can be applied from frames 20 to 14 (first segment) and cannot be employed from frames 13 to 8 (second segment).

Table 5.6. Average PSNR comparison of the first segment when the client requests backward-play operation at the start of each GOP in the RB.

Sequences	Bitrate	Original dual-bitstream system	Macroblock-based dual-bitstream system	Gain
Salesman	1.5M	34.90	36.64	1.74
	3M	37.80	39.85	2.05
Football	1.5M	28.03	28.74	0.70
	3M	31.78	32.76	0.98
Table Tennis	1.5M	30.74	31.78	1.04
	3M	35.32	37.05	1.73
Foreman	64K	25.03	25.46	0.43
	128K	26.54	27.31	0.77
Carphone	64K	27.90	28.49	0.59
	128K	30.67	31.53	0.87
Claire	64K	33.06	34.53	1.47
	128K	38.63	40.48	1.85
Grandma	64K	31.55	32.31	0.76
	128K	33.85	35.46	1.62

Table 5.7. Average PSNR comparison of the second segment when the client requests backward-play operation at the start of each GOP in the RB.

Sequences	Bitrate	Original dual-bitstream system	Macroblock-based dual-bitstream system	Gain
Salesman	1.5M	34.54	35.04	0.50
	3M	37.54	37.90	0.36
Football	1.5M	27.51	27.72	0.21
	3M	31.56	31.69	0.13
Table Tennis	1.5M	30.22	30.59	0.38
	3M	35.12	35.35	0.24
Foreman	64K	23.98	24.23	0.25
	128K	25.53	25.79	0.25
Carphone	64K	27.00	27.28	0.28
	128K	29.84	30.14	0.30
Claire	64K	32.43	33.29	0.86
	128K	38.05	38.61	0.56
Grandma	64K	31.14	31.59	0.46
	128K	33.42	34.17	0.76

5.4 Chapter Summary

In this chapter, we have addressed several issues in implementing an MPEG video streaming system with VCR functionality. The dual-bitstream system was proven to be an efficient approach by storing the FB and RB in the server. This approach reduces the requirement of network bandwidth significantly, however, the mismatch between the encoding of FB and RB incurs the problem of drift. Therefore, we propose an efficient MB-based selection scheme for the dual-bitstream system. The proposed scheme is motivated by

the center-biased motion vector distribution of real-world video sequences. With the motion information, the video streaming server organizes the MBs in the requested frame into two categories – non-RMMB and RMMB. Then it selects the necessary MBs adaptively, processes them in the compressed domain and sends the processed MBs to the client machine. For non-RMMBs, we have proposed a technique of sign inversion of DCT coefficients to ensure that both the reference frame and the prediction error are come from the same bitstream which can avoid the mismatch problem. Since the non-RMMB is manipulated on the VLC domain, it does not complicate the server's implementation. Furthermore, since the mismatch problem can be prevented, the visual quality of the non-RMMB during backward playback is exactly the same as that of forward playback. Simulation results show that, with our proposed MB-based solution, the drift problem due to the P-to-P or P-to-I approximation for backward playback can be alleviated significantly.

Chapter 6 – Redundancy Reduction for the Dual-Bitstream System

6.1 Introduction

The frame-based and MB-based approaches mentioned in Chapters 4 and 5 can alleviate the drift problem due a P-to-P or an I-to-P approximation in the dual-bitstream system. They are very suitable for video applications that require high-quality video browsing. Example applications include high-definition TV and studio video editing. In contrast, for fast browsing video on nowadays mobile phones with only a small display, the drift becomes unnoticeable to human eyes due to the fast changes of the content displayed. Therefore, the high-quality video browsing is not necessary in some applications.

In this chapter, a simplified RB is introduced to reduce the extra storage required for the RB. It is useful for low-cost video servers with limited storage capacity. In this simplified RB, we suggest reusing some MB data from the FB by exploiting their redundancy, and then propose a novel MB-selection strategy to adaptively select the appropriate MBs from the two bitstreams.

6.2 Simplified RB (SRB) in the Dual-bitstream System

Although the dual-bitstream system can provide an effective way to support VCR operations for MPEG video, it requires additional storage for the RB. It is found that the MB-based algorithm proposed in Chapter 5 can be further extended in order to reduce the storage requirement of the RB. The proposed algorithm attempts to exploit redundancy in some MBs found between the two bitstreams. For convenience of our discussion, the MB viewpoint of the dual bitstreams in Figure 5.1 is redrawn with some modifications, as shown in Figure 6.1. Again, $MB_{(k,l)}^{FB_n}$ and $MB_{(k,l)}^{RB_n}$ represent the reconstructed MBs at the k^{th} row and l^{th} column of frame n in the FB and RB respectively. As compared with Figure 5.1, non-RMMB and RMMB are now replaced by SMB (skipped MB) and non-SMB (non-skipped MB), respectively. These symbols are more suitable to discuss the redundancy reduction of the FB and RB in this chapter.

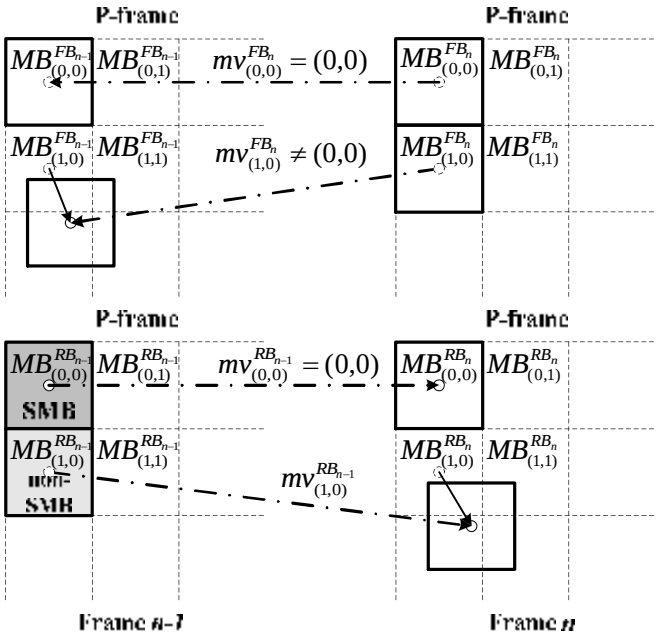


Figure 6.1. MB viewpoint of the proposed dual-bitstream system and the definition of the SMB and non-SMB.

To reduce the temporal redundancy in coding video sources, block motion-compensated prediction is used in which the previously transmitted and decoded frame serves as the prediction for the current frame. In Figure 6.1, The prediction error in the FB, $e^{FB_n}_{(k,l)}$, can be written as

$$e^{FB_n}_{(k,l)} = MB^{FB_n}_{(k,l)} - MCMB^{FB_{n-1}}(mv^{FB_n}_{(k,l)}) \tag{6.1}$$

where $MCMB^{FB_{n-1}}(mv^{FB_n}_{(k,l)})$ stands for the motion-compensated MB of $MB^{FB_n}_{(k,l)}$ with the motion vector $mv^{FB_n}_{(k,l)}$ pointed to frame *n-1* in the FB.

In the RB, the prediction error, $e^{RB_{n-1}}_{(k,l)}$, is given by

$$e^{RB_{n-1}}_{(k,l)} = MB^{RB_{n-1}}_{(k,l)} - MCMB^{RB_n}(mv^{RB_{n-1}}_{(k,l)}) \tag{6.2}$$

Here, $MCMB^{RB_n}(mv^{RB_{n-1}}_{(k,l)})$ represents the motion-compensated MB of $MB^{RB_{n-1}}_{(k,l)}$

with the motion vector $mv_{(k,l)}^{RB_{n-1}}$ pointed to frame *n* of the RB. This prediction error is DCT-transformed. The DCT coefficients are then quantized, variable-length encoded and stored in the video server.

In a video sequence, frames at the same time instant in the FB and RB are perceptually similar to each other. They actually represent the same content and have similar color, texture, and objects, but the only difference is the coding directions, as described in (6.1) and (6.2). This means if the RB is encoded completely as a separate bitstream from the FB, a considerable amount of redundancy exists. To generate the RB in the video server with limited storage capacity, the strategy is to reuse the MB data as much in the FB as possible. To do so, a special measure is taken to encode some MBs in the RB which can utilize the MB data in the FB. In the proposed technique, all coefficients of these MBs are not necessary to be encoded and they are defined as skipped MBs (SMBs) in the RB. $MB_{(k,l)}^{RB_{n-1}}$ is classified as a SMB if the MB in frame *n* of the FB, $MB_{(k,l)}^{FB_n}$, is coded with zero MV. Otherwise, it is classified as a non-SMB. For illustration, we use the example in Figure 6.1 again to give a clear account of the definition of the SMB. In this figure, since the motion vector , $mv_{(0,0)}^{FB_n}$, of $MB_{(0,0)}^{FB_n}$ is equal to zero, $MB_{(0,0)}^{RB_{n-1}}$ is classified as a SMB. On the other hand, since $MB_{(1,0)}^{FB_n}$ is coded with non-zero MV, $MB_{(1,0)}^{RB_{n-1}}$ is classified

as a non-SMB.

When $MB_{(k,l)}^{RB_{n-1}}$ in the RB is defined as a SMB, its corresponding MB in frame *n* of the FB, $MB_{(k,l)}^{FB_n}$, is coded without motion compensation. This implies that the spatial position of $MB_{(k,l)}^{FB_{n-1}}$ in the FB is the same as that of $MB_{(k,l)}^{FB_n}$. In this case, $MCMB^{FB_{n-1}}(mv_{(k,l)}^{FB_n})$ is equal to $MB_{(k,l)}^{FB_{n-1}}$, and (6.1) can be rewritten as

$$e_{(k,l)}^{FB_n} = MB_{(k,l)}^{FB_n} - MB_{(k,l)}^{FB_{n-1}} \tag{6.3}$$

In order to reuse the MB data as much in the FB as possible during encoding $MB_{(k,l)}^{RB_{n-1}}$ of the RB, its motion vector is enforced to zero as well, i.e., $mv_{(k,l)}^{RB_{n-1}} = 0$. Such arrangement is to ensure $MCMB^{RB_n}(mv_{(k,l)}^{RB_{n-1}})$ is equal to $MB_{(k,l)}^{RB_n}$ such that (6.2) becomes

$$e_{(k,l)}^{RB_{n-1}} = MB_{(k,l)}^{RB_{n-1}} - MB_{(k,l)}^{RB_n} \tag{6.4}$$

As mentioned before, frame *n* of the FB and RB share the same video content. Because of this, pixels of $MB_{(k,l)}^{FB_n}$ and $MB_{(k,l)}^{RB_n}$ are similar and it is reasonable to approximate $MB_{(k,l)}^{RB_n}$ by $MB_{(k,l)}^{FB_n}$ during various VCR operations. That is,

$$MB_{(k,l)}^{RB_n} \approx MB_{(k,l)}^{FB_n} \tag{6.5}$$

Similarly,

$$MB_{(k,l)}^{RB_{n-1}} \approx MB_{(k,l)}^{FB_{n-1}} \tag{6.6}$$

By putting (6.5) and (6.6) into (6.4), it can be rewritten as

$$e_{(k,l)}^{RB_{n-1}} = MB_{(k,l)}^{FB_{n-1}} - MB_{(k,l)}^{FB_n} \tag{6.7}$$

From (6.3) and (6.7), we get

$$e_{(k,l)}^{RB_{n-1}} = -e_{(k,l)}^{FB_n} \tag{6.8}$$

By applying the DCT to (6.8) and considering that the DCT is an odd transform, we can obtain $e_{(k,l)}^{RB_{n-1}}$ in the DCT domain according to the following equation,

$$DCT(e_{(k,l)}^{RB_{n-1}}) = -DCT(e_{(k,l)}^{FB_n}) \tag{6.9}$$

Then its quantized DCT coefficients, $Q[DCT(e_{(k,l)}^{RB_{n-1}})]$, are written as

$$Q[DCT(e_{(k,l)}^{RB_{n-1}})] = -Q[DCT(e_{(k,l)}^{FB_n})] \tag{6.10}$$

$Q[DCT(e_{(k,l)}^{RB_{n-1}})]$ is the quantized DCT coefficients to be encoded in the RB. However, $Q[DCT(e_{(k,l)}^{FB_n})]$ is already available in the FB. From (6.10), $Q[DCT(e_{(k,l)}^{RB_{n-1}})]$ can be extracted directly from the FB by simply inverting the signs of all quantized DCT coefficients in $Q[DCT(e_{(k,l)}^{FB_n})]$. Therefore, the server can store the simplified RB (SRB) instead of the RB. SRB is the one that video data about the SMBs are not encoded and the quantized DCT coefficients are taken from the FB during VCR operations. In other words, the data in these MBs are shared among the FB and SRB. Therefore, the storage requirement of the SRB can be reduced remarkably.

6.3 Architecture of Video Steaming Server with the support

of SRB

The above section only addresses how to eliminate the redundancy between the dual bitstreams. When a VCR operation requests a particular frame from the SRB, the server adopts a MB-selection strategy which needs to extract the appropriate MBs from the FB in order to insert the shared data into the SRB prior to transmission. Figure 6.2 shows the proposed architecture of a video streaming server. When a frame from the SRB is requested, the appropriate motion vectors are extracted from the FB. The server does not need to do anything for non-SMBs, and the switch *SW* is connected to position B. For a SMB, the server uses only the data from the FB to reconstruct the requested MB in the SRB by switching *SW* to position A. In each SMB, the corresponding VLC codewords are extracted from the FB. Afterwards, these VLC codewords are undergone VLC decoding to reconstruct the quantized DCT coefficients. From (6.10), the signs of all coefficients are inverted to form the desired coefficients, which are encoded to its final VLC codewords for the SMB. The non-SMBs in the SRB are then integrated with these VLC codewords before transmitting to the network. Note that the server only needs to perform variable length decoding and encoding, and a complete decoding and encoding are not required at the server. It only causes a slightly increase

in the server complexity. Note that the VLC-domain technique for sign inversion mentioned in Section 5.2.4 can also be used in the streaming server to further reduce its complexity.

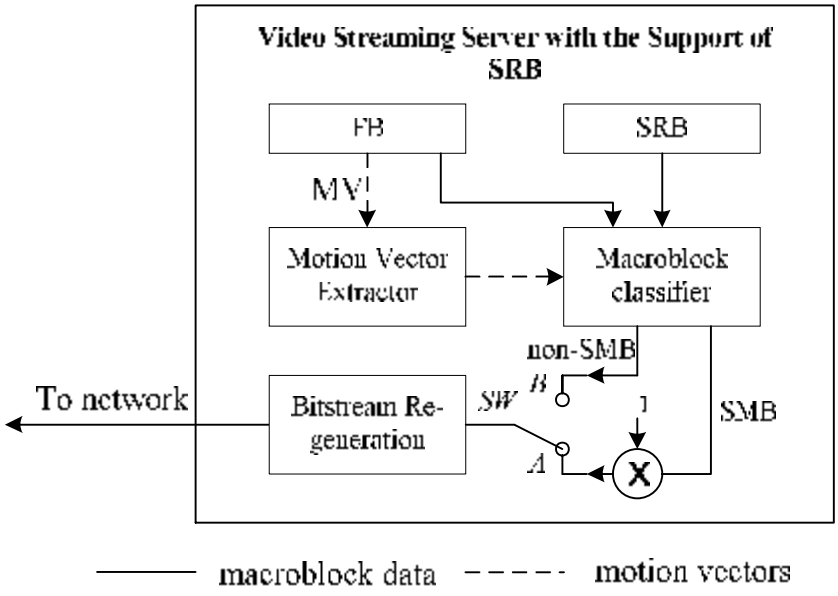


Figure 6.2. The proposed architecture for the dual-bitstream video streaming scheme with VCR functionality.

6.4 Experimental Results

This section contains the results of some simulations performed with the proposed SRB when applied to the dual-bitstream streaming system with VCR support. Again, MPEG-4 encoder [58] was used to encode the same set of the video sequences as tabulated in Table 4.1. These sequences cover various spatial resolutions and motion characteristics, they were encoded at two different bit rates with the frame rate of 30 frames/s. For all sequences, the GOP length was set to 14. Each test sequence was encoded into two

bitstreams, the FB and the SRB (or RB), and I-frames in the SRB (or RB) are interleaved between I-frames in the FB. The SRB (or RB) can be obtained by re-encoding the FB in reverse order. Note that the generation of the SRB (or RB) is done offline.

In Table 6.1, we show the bitstream size and the average PSNR value for each test sequence that was encoded into the RB and SRB at two different bit rates. In this table, ΔPSNR and ΔSIZE represent a PSNR change and percentage change in the bitstream size of the SRB when compared to the original RB. A positive value means an increment whereas a negative value means a decrement. It can easily be seen that the required storage in the server of the proposed SRB is much fewer that that of the original RB in both bit rates. The results are more significant for the sequences “Salesman”, “Claire”, and “Grandma” as shown in Table 6.1. In these sequences, the size of the SRB can be reduced by 40-48% and 28-40% as compared to the original RB at high bit rate and low bit rate, respectively. It is due to the reason that these sequences contain more SMBs in which the redundancy to be exploited between the two bitstreams becomes more significant. For sequences containing high motion activities such as “Football”, “Table Tennis”, “Foreman” and “Carphone”, there still have good savings, as tabulated in Table 6.1.

Besides, this table signifies that the size of the SRB can be reduced more remarkably for sequences encoded at high bit rate. The reason behind is that, at low bit rate, a considerable percentage of DCT blocks have a significant amount of zero elements in MBs of the original RB. In these MBs, the encoder needs to allocate fewer bits for encoding the residuals. If those MBs are considered as SMBs in the proposed SRB, it cannot achieve as much saving of bits as the case in video sequences encoded at high bit rate.

Table 6.1. Average PSNR and bitstream size for various sequences.

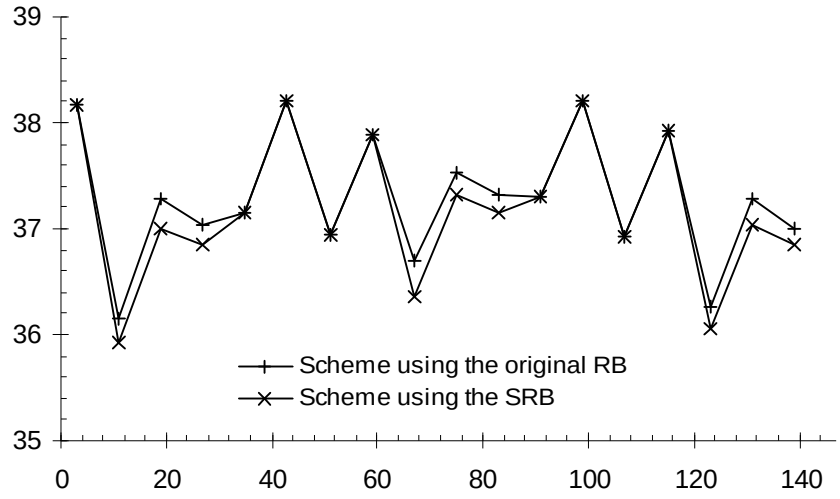
Sequences	Bit rate of FB	FB		RB		SRB		ΔPSNR (dB)	ΔSize
		PSNR (dB)	Size (KB)	PSNR (dB)	Size (KB)	PSNR (dB)	Size (KB)		
Salesman	3M	42.171	2407.074	40.600	2498.703	40.265	1292.170	-0.334	-48.29%
	1.5M	37.690	1087.164	36.842	1157.830	36.711	687.312	-0.131	-40.64%
Football	3M	34.318	2554.604	32.825	2781.006	32.705	2144.181	-0.119	-22.90%
	1.5M	30.653	1168.521	28.992	1346.715	28.876	1118.515	-0.116	-16.94%
Table Tennis	3M	38.907	2248.761	37.094	2362.411	36.644	1786.607	-0.449	-24.37%
	1.5M	33.927	1149.868	32.693	1263.781	32.377	1000.946	-0.316	-20.80%
Foreman	128K	29.019	107.365	27.682	121.813	27.541	96.051	-0.141	-21.15%
	64K	26.204	56.651	25.324	60.575	25.239	51.715	-0.084	-14.63%
Carphone	128K	33.036	107.729	31.799	119.139	31.643	86.188	-0.156	-27.66%
	64K	29.416	58.930	28.474	63.978	28.352	51.616	-0.122	-19.32%
Claire	128K	40.673	103.487	39.973	110.903	39.648	66.702	-0.325	-39.86%
	64K	34.496	54.969	33.772	58.391	33.553	41.635	-0.219	-28.70%
Grandma	128K	36.322	107.430	35.843	116.936	35.726	68.207	-0.118	-41.67%
	64K	32.350	55.304	31.944	57.712	31.877	41.461	-0.067	-28.16%

Table 6.2. Average PSNR of all possible requested frames with respect to all starting point for various sequences.

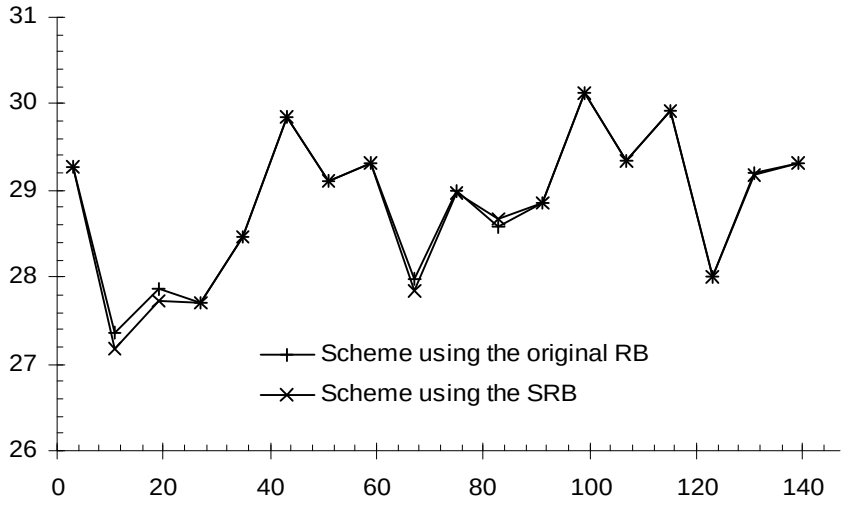
Sequences	Bit rate of FB	RB (dB)	SRB (dB)	ΔPSNR (dB)
Salesman	3M	40.308	40.271	-0.037
	1.5M	36.703	36.679	-0.024
Football	3M	32.311	32.329	0.018
	1.5M	28.426	28.436	0.010
Table Tennis	3M	36.654	36.517	-0.137
	1.5M	32.332	32.235	-0.097
Foreman	128K	27.503	27.535	0.032
	64K	25.351	25.360	0.009
Carphone	128K	31.688	31.706	0.018
	64K	28.483	28.482	-0.001
Claire	128K	40.000	39.978	-0.022
	64K	33.910	33.886	-0.024
Grandma	128K	35.905	35.897	-0.008
	64K	32.047	32.050	0.003

The average PSNR values of the RB and SRB are also shown in Table 6.1. They show that the average PSNR values are slightly degraded by about 0.067 dB to 0.449 dB for the SRB. The degradation is due to the approximation in (6.5) and (6.6). In fact, this small degradation reflects the quality of the reconstructed frames during backward playback. In other VCR operations, the quality degradation is also negligible as shown in Figure 6.3. It illustrates that the PSNR comparison for decoding the requested frames by using the RB and SRB when the fast backward operation with a speed-up factor of 8 is issued at the end of the sequences. In this Figure, the “Salesman” and “Carphone” sequences were encoded at 1.5Mbits/s and 64Kbits/s, respectively. When the

server performs bitstream switching by using the proposed SRB, there is a slight PSNR drop in some requested frames. However, for devices with only a small display, this negligible degradation is not significant visually in the fast-backward mode since the fast display speed will mask out most of the spatial distortion. We have also done exhaustive simulation on the PSNR performances of all possible combinations of requested frames and start frames (the frame in which a user issues a VCR operation). Table 6.2 lists out the average PSNR values of these possible combinations when the sequences are coded at high bit rate and low bit rate. In fact, these values indicate the average PSNR performances of the random access mode for using the RB and SRB. Random access is an important operation for providing VCR capability in a video browsing system. From the statistics as shown in Table 6.2, it can easily be seen that the proposed SRB in the dual-bitstream system can achieve almost the same quality as compared to the scheme using the RB during random access. Therefore, the SRB can reduce the storage requirement significantly as well as keeping the reconstruction quality of various VCR browsing operations.



(a)



(b)

Figure 6.3. PSNR performances by using the original RB and the SRB in the fast-backward mode with a speed-up factor of 8 for the (a) "Salesman", and (b) "Carphone" sequences.

6.5 Chapter Summary

In this chapter, an efficient technique for reducing the storage requirement of the server has been proposed to eliminate the possible redundancy between the dual bitstreams. The proposed technique exploits a large amount of

zero-MV MBs existed in real-world video sequences. With the motion information, the video streaming server classifies some MBs as skipped MBs (SMBs). A SMB is the one that the information about the MB is not necessary to be stored in the server and it is taken directly from the FB. By sharing the data between the dual bitstreams, a new and simplified RB (SRB) is used instead of the RB in the dual-bitstream system. Simulation results show that, with our proposed SRB, the dual-bitstream system reduces the storage requirement of the server with just a slight drop in PSNR for various VCR operations.

Chapter 7 – Conclusions and Future Directions

7.1 Conclusions of the Present Works

In this thesis, we have investigated some techniques for facilitating digital video browsing capability on the MPEG video streaming system. These techniques can enhance the dual-bitstream system, which had been known to be an efficient approach to enable quick and user-friendly browsing of video contents with minimum requirements on the network bandwidth and the decoder complexity.

For the dual-bitstream system, the server stores both the forward-encoded bitstream (FB) and the reverse-encoded bitstream (RB). The idea is to switch frames between the FB and RB by minimizing the number of transmitted frames for any speed-up factors. Unfortunately, two technical challenges have not yet been well resolved. First, the extra RB in the dual bitstreams increases the storage requirement of the server. Second, the frame in one bitstream may not be exactly identical to the frame in another bitstream. If one of these frames is used as the reference for a frame in the other bitstream, it induces the drift problem. The detailed analysis of the drift problem has been made in Chapter 3. Results of our study also indicated that the quality degradation due

to the drift is very serious. This makes a hurdle for interactive browsing operations with high-quality guarantee, which are desirable features in video-on-demand and high-definition TV. A further need for reducing drift errors may arise.

Therefore, in this thesis, the major objective is to develop efficient techniques in the dual-bitstream system for supporting interactive browsing operations in digital video. Effort has been made to provide solutions for the drift and storage problems of the dual-bitstream system. In Chapter 4 and Chapter 5, we have proposed two solutions to deal with the drift problem arising from bitstream switching. These solutions are operated either on frame level or MB level. They give a new direction for the implementation of the dual-bitstream system with VCR functionality. Afterwards, in Chapter 6, a MB-level solution has been designed to tackle the storage problem in the video server. The proposed MB-based technique could reduce the redundancy between the FB and RB. This solution is conducive to video browsing applications in which the server has only limited storage capacity.

The design of the original dual-bitstream structure is based on the MPEG video coding structure with conventional I/P-frames. In Chapter 4, it has provided a new direction for the utilization of SI/SP-frames in the dual-bitstream

structure. The SP/SI-frames are designed in the newest H.264 coding standard to support seamlessly bitstream switching so as to accommodate the bandwidth variation in video streaming. By making use of the concept from SP/SI-frames, we have replaced the original P/I pairs in the dual bitstreams with the SP/SI-pairs. This new arrangement could totally resolve the drift problem for fast-forward/backward operations with speed-up factors larger than *N/4* and random access. The reason is that the frame-selection scheme selects d_{FB} and d_{RB}, and initiates the decoding from the nearest I-frame in the case of a VCR operation with a large speed-up factor (larger than *N/4*). Thus a possible switching always occurs at P/I pairs and the I-to-P approximation is required such that a P-frame in one bitstream is approximated by I-frame in another bitstream. In our new dual bitstreams, the use of SP/SI pairs could efficiently eliminate the mismatch errors. We have then formulated how to encode SP/SI pairs and given the complete encoding and decoding structures in Chapter 4. The derivation in Chapter 4 has also proven that drift-free browsing could be achieved by adopting SP/SI-frames in the dual bitstreams. It has been found that the proposed dual-bitstream structure really achieves the same reconstructed frames as the original frames in the server during video browsing.

However, due to the decrease in coding efficiency, the dual bitstreams with SP/SI-frames affects the performance of normal playback. Therefore, the content in Chapter 5 is another contribution of the thesis where a novel MB-based solution in the server has been proposed for the realization of VCR operations. In our study, it has been found that different MBs in one particular frame have different properties on their compressed domain during video browsing. Therefore, by using the motion information, a MB-selection scheme at the server has been designed in which MBs in the FB and RB are classified into two types. They are reference-mismatched MBs (RMMBs) and non-RMMBs. After the classification of MBs with our proposed server, non-RMMBs are manipulated by the sign inversion technique to reverse the motion compensation process from using the previously decoded MBs as the reference in order to obtain the requested MB. Since the sign inversion technique is operated on the compressed domain, complete decoding and encoding are not required in the server. The additional computation requirement for the server is then limited. Experimental results have confirmed that over 90% and 39% of the MBs are classified as non-RMMBs for sequences containing a low and high amount of motion activities respectively. Consequently, the average PSNR performance of non-RMMBs improves 1.7dB

as compared with the original dual-bitstream structure.

The solutions suggested in Chapter 4 and Chapter 5 could successfully alleviate the drift. They are important to high-quality video streaming service. In Chapter 6, we have considered the scenario in which the video streaming server has limited storage space and the video browsing is taken place on mobile devices with only a small display. In this case, the effect of drift becomes invisible in such a small display under fast browsing. Since the original dual-bitstream system stores an additional RB, it doubles the storage requirement for the server. It is not desirable for low-cost video servers with limited storage capacity. Therefore, a simplified RB has been contrived in order to minimize the extra storage required for the RB. In the SRB, some MBs are allowed to be skipped and reuse the MB data from the FB. This arrangement can exploit the redundancy between the FB and the RB. For those MBs in the FB with zero MVs in frame *n+1*, we can always find the prediction errors of the collocated MB in frame *n* by using the sign inversion technique. Therefore, the corresponding MBs in the RB are not necessary to be coded. These MBs are defined as skipped MBs (SMBs). Due to the fact that the amount of MBs having zero MVs is large, the experimental results have show that the storage savings in the SRB could achieve up to 48% with small quality degradation.

In conclusion, we can expect the amount of video contents available to grow as the widespread adoption of Internet video streaming and the rapid development of playback devices. Efficient video browsing for digital video becomes indispensable. In our present work, a number of techniques have been investigated and they can enhance the dual-bitstream system in different aspects. We believe that these techniques in cooperation with the dual-bitstream system will play a vital role in the future market of the video streaming server with VCR support.

7.2 Future Directions

In this thesis, we have proposed several techniques to resolve the problems of video browsing in the dual-bitstream system. Compressing video is an active research topic, and different video coding techniques are designed in a way that only minimum processing resources are needed if a compressed video is decoded for normal playback in a pre-determined order. For browsing video, the challenge is therefore how to efficiently decode the video sequence in any other order. With the successful techniques described in this thesis and proven by a wide range of experimental works, we give some opinions on the trend for the future development of our related studies.

7.2.1 Adaptive Macroblock-selection Scheme for the Dual-bitstream System with SP/SI-frame Coding

The proposed dual-bitstream structure with SP/SI-frames can completely eliminate drift errors caused by any I-to-P approximation when a user requests a fast-forward/backward operation. This new bitstream arrangement has an excellent switching performance, but it causes quality degradation of the FB which directly affects normal playback. On the other hand, the proposed MB-based algorithm solves the P-to-P approximation when the bitstream switching occurs at a P-frame. This MB-based solution has no influence on normal playback though it cannot totally remove the drift. To further improve the video browsing capability in video streaming, how to keep the qualities of both VCR operations and normal playback is the main concern of our further work. To achieve this, it is possible to integrate the MB-based algorithm into the dual bitstreams with SP/SI-frames in order to balance the quality of normal playback and the switching performance.

To illustrate the idea more efficiently, let us use Figure 7.1 as an example in which the MB-based algorithm is adaptively integrated into the dual bitstreams with SP/SI-frames. As shown in Figure 7.1, there are two types of MB pairs at the switching point (frame *n*) – an SP/SI MB pair and a P/I MB pair. The SP/SI

MB pair is good for bitstream switching while the P/I MB pair can keep the quality of normal playback. A control scheme should be involved in determining whether a particular MB is coded as an SP/SI MB pair or P/I MB pair at each switching point. To evaluate the quality of normal playback by using SP/SI MB and P/I MB pairs, the sum of absolute difference (SAD) between the pixels of original MB (O_n) and the pixels of SP/SI MB $(SPMB^r_{RB,n} / SIMB^r_{FB,n})$ or P/I MB $(PMB^r_{RB,n} / IMB^r_{FB,n})$ can be used, and they are computed as

$$SAD^{SP/SI}_{FB} = \sum \left| SIMB^r_{FB,n} - O_n \right| \tag{7.1}$$

and

$$SAD^{P/I}_{FB} = \sum \left| IMB^r_{FB,n} - O_n \right| \tag{7.2}$$

respectively. $SAD^{SP/SI}_{FB}$ and $SAD^{P/I}_{FB}$ represents the reconstructed errors of normal playback using the SP/SI MB and P/I MB pairs, respectively. If the quality of backward playback is also taken into account, (7.1) and (7.2) can be rewritten as

$$SAD^{SP/SI}_{FB+RB} = \sum \left| SIMB^r_{FB,n} - O_n \right| + \sum \left| SPMB^r_{RB,n} - O_n \right| \tag{7.3}$$

and

$$SAD^{P/I}_{FB+RB} = \sum \left| IMB^r_{FB,n} - O_n \right| + \sum \left| PMB^r_{RB,n} - O_n \right| \tag{7.4}$$

respectively.

On the other hand, for the P/I MB pair, the drift errors during bitstream switching can be computed by considering the difference between the IMB and PMB in the FB and RB respectively. For simplicity, the difference can be approximated by the SAD again, and it can be formulated as

$$SAD_{FB\to RB}^{P/I} = \sum \left| IMB_{FB,n}^{r} - PMB_{RB,n}^{r} \right| \tag{7.5}$$

Note that the SP/SI MB pair does not suffer any drift problem. To balance the playback quality and switching performance, we can make use of (7.3) to (7.5) to design a control scheme for determining the appropriate MB type in the dual bitstreams. One possible control scheme is defined as follows.

$$SAD_{FB+RB}^{SP/SI} < SAD_{FB+RB}^{P/I} + \lambda \cdot SAD_{FB\to RB}^{P/I} \tag{7.6}$$

where λ is the weighting factor of the switching performance. Further investigation should be needed in order to find a way to determine λ. If (7.6) holds true, then the SP/SI MB pair is used; otherwise the P/I MB pair is selected. By using this control scheme, the optimal MB types in the dual bitstreams can be found with the consideration of the playback quality and switching performance at every switching point.

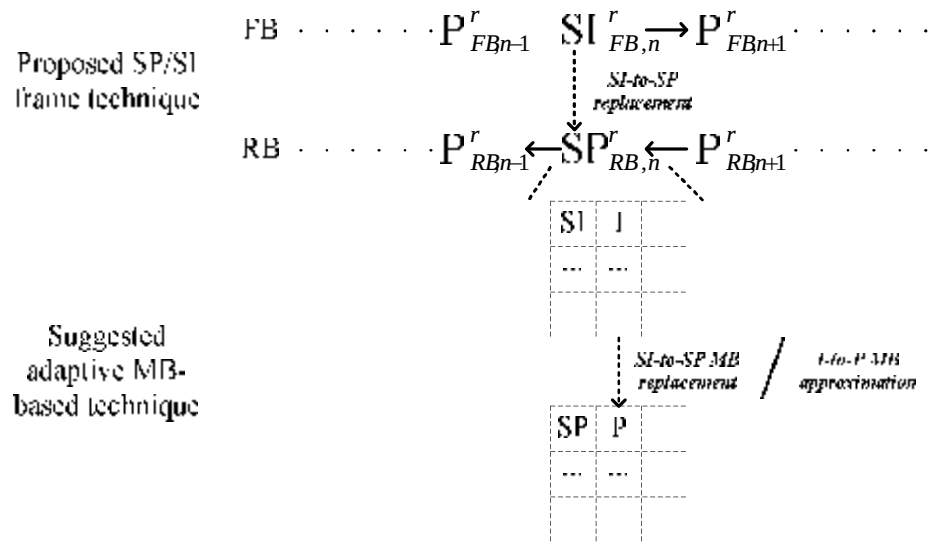


Figure 7.1. Integration of the MB-based solution into the dual bitstreams with SP/SI-frames.

7.2.2 The Use of Multiple Reference Frames in the Dual-bitstream Structure in H.264

In the emerging H.264 standard, it allows to use multiple reference frames for motion prediction in order to achieve higher coding efficiency. This new feature provides a great challenge to perform video browsing on the H.264 encoded bitstream since the use of multiple reference frames introduces much more dependency among frames. On browsing digital video, more frames are then necessary to be transmitted to the client.

Figure 7.2 illustrates an example when four reference frames are used for motion prediction. When a user requests a random-access operation to frame 8, frame 4 to frame 7 in the FB must be available on the frame buffer prior to

decoding frame 8. The original frame-selection scheme selects frame 7 in the RB to approximate frame 7 in the FB, but it is not sufficient to decode frame 8. Therefore, frame 4 to frame 6 in the RB are used to approximate frame 4 to frame 6 in the FB as well. In this case, the server needs to send three more frames as compared with the case of using a single reference frame. Therefore, motion estimation and compensation with multiple reference frames severely complicates the browsing operations. This is crucial issue that should be investigated before they can be put into practical for browsing H.264 video.

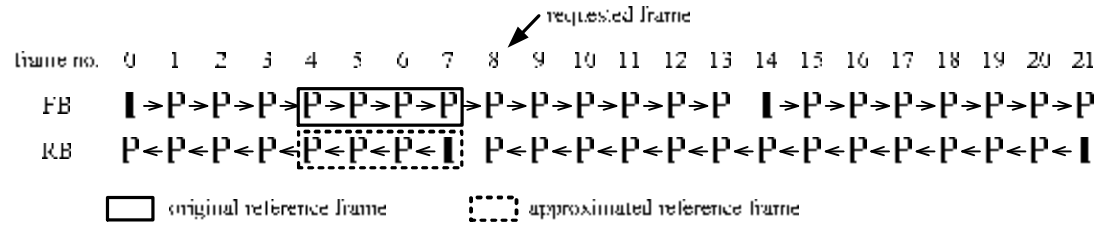


Figure 7.2. The dual-bitstream structure with multiple reference frames.

References

[1] H.J. Stuttgen, “Network evolution and multimedia communication,” IEEE Multimedia, Vol. 2, pp. 42-59, Fall 1995.

[2] Dapeng Wu, Yiwei Thoms Hou, Wenwu Zhu, Ya-Qin Zhang, Peha, J. M., “Streaming video over the Internet: approaches and directions”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 3, March 2001, pp. 282-300.

[3] Dapeng Wu, Yiwei Thoms Hou and Ya-Qin Zhang, “Transporting real-time video over the Internet: challenges and Approaches,” Proceedings of the IEEE, Vol. 88, No. 12, December 2000, pp. 1855-1877.

[4] L. Chiariglione, “The development of an integrated audiovisual coding standard: MPEG,” Proceedings of IEEE, Vol. 83, Feb. 1995, pp. 151-157.

[5] 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.

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

[7] ITU-T Rec. H.263, “Video Coding for Low Bitrate Communication,” 1997.

[8] ISO/IEC 14996-2, “Coding of Audio-visual Objects – Part 2: Visual,” 2001.

[9] ISO/IEC 14996-10 and ITU-T Rec. H.264, “Advanced Video Coding,” 2003.

[10] T.D.C Little and D. Venkatesh, “Prospects for interactive video-on-demand,” IEEE Multimedia, Vol. 13, August 1994, pp. 14-24.

[11] A. Ganjam and H. Zhang, “Internet multicast video delivery,” Proceedings of the IEEE, Vol. 93, No. 1, January 2005, pp. 159-170.

[12] J. M. McManus and K.W. Ross, “Video-on-demand over ATM: constant-rate transmission and transport,” IEEE Journal on Selected Areas in Communications, Vol. 14, No. 6, August 1996, pp. 1087-1098.

[13] T. Hanamura, W. Kameyama, and H. Tominaga, “Hierarchical coding scheme of video signal with scalability and compatibility,” Signal Processing: Image Communication, Vol. 5, February 1993, pp. 159-184.

[14] G. Morrison and I. Parke, “A spatially layered hierarchical approach to video coding,” Signal Processing: Image Communication, Vol. 5, December 1993, pp. 445-462.

[15] D. Wilson and M. Ghanbari, “Optimization of two-layer SNR scalability for MPEG-2 video,” Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 4, pp. 2637-2640, 1997.

[16] R. Aravind, M. R. Civanlar and A. R. Reibman, “Packet loss resilience of MPEG-2 scalable video coding algorithms,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 5, October 1996, pp. 426-435.

[17] Weiping Li, “Overview of Fine Granularity Scalability in MPEG-4 Video Standard,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 3, March 2001, pp.301-317.

[18] Feng Wu, Shipeng Li, and Ya-Qin Zhang, “A Framework for Efficient Progressive Fine Granularity Scalable Video Coding,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 3, March 2001, pp. 332-344.

[19] Hsiang-Chun Huang, Chung-Neng Wang, and Tihao Chiang, “A Robust Fine Granularity Scalability Using Trellis-Based Predictive Leak,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No. 6, June 2002, pp. 372-385.

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

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

[22] SonicBlue Inc. [Online]. Available: http://www.replay.com/

[23] TiVo Inc. [Online]. Available: http://www.tivo.com/

[24] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, “A novel unrestricted center-biased diamond search algorithm for block motion estimation,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 8, August 1998, pp. 369-377.

[25] Lai-Man Po and Wing Chung Ma, “A novel four-step search algorithm for fast block motion estimation,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, June 1996, pp. 313-317.

[26] Renxiang Li, Bing Zeng, and Ming L. Liou, “A new three-step search algorithm for block motion estimation,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 4, August 1994, pp.438-442.

[27] R. Srinivasan and K. R. Rao, “Predictive coding based on efficient motion estimation,” IEEE Transactions on Communications, Vol. 33, No. 9, September 1985, pp. 1011-1015.

[28] Yui-Lam Chan and Wan-Chi Siu, “New adaptive pixel decimation for block motion vector estimation,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 1, February 1996, pp. 113-118.

[29] Yui-Lam Chan and Wan-Chi Siu, “An efficient search strategy for block motion estimation using image features,” IEEE Transactions on Image

Processing, Vol. 10, No. 8, August 2001, pp. 1223-1238.

[30] N. Farber and B. Girod, “Robust H.263 compatible video transmission for mobile access to video servers,” Proceedings of IEEE International Conference on Image Processing, Vol. 2, October 1997, pp. 73–76.

[31] Bo Xie and Wenjun Zeng, “Source characteristics based fast bitstream switching,” Proceedings of IEEE International Conference on Multimedia and Expo, 2003, ICME2003, Vol.1, July 2003, pp. 521-524.

[32] Bo Xie and Wenjun Zeng, “Two fast bitstream switching algorithms for real-time adaptive multicasting of video,” IEEE International Conference on Communications 2004, Vol. 3, June 2004, pp. 1283-1287.

[33] Bo Xie and Wenjun Zeng, “Rate-distortion optimized dynamic bitstream switching for scalable video streaming,” Proceedings of IEEE International Conference on Multimedia and Expo 2004, ICME2004, Vol. 2, June 2004, pp. 1327-1330.

[34] Bo Xie and Wenjun Zeng, “On the rate-distortion performance of dynamic bitstream switching mechanisms,” Proceedings of IEEE International Conference on Multimedia and Expo 2005, ICME2005, Vol. 2, July 2005, 4 pp. 13.

[35] Marta Karczewicz and Ragip Kurceren, “The SP- and SI-Frames Design

for H.264/AVC," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 7, July 2003, pp. 637-644.

[36] Xiaoyan Sun, Shipeng Li, Feng Wu, Jacky Shen and Wen Gao, "The Improved SP Frame coding Technique for the JVT Standard," Proceedings of IEEE International on Conference Image Processing 2003, ICIP2003, September 2003, pp. 297-300.

[37] Ragip Kurceren and Marta Karczewicz, "Synchronization-Predictive Coding for Video Compression: The SP Frames Design JVT/H.26L," Proceedings of IEEE International on Conference Image Processing 2002, ICIP2002, September 2002, pp. 497-500.

[38] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, Ajay Luthra, "Overview of the H.264/AVC video coding standard," IEEE Transactions on Circuits and Systems for Video Technology, July 2003, Vol. 13, No. 7, pp. 560-576.

[39] ITU-T Recommendation H.264 / ISO/IEC 11496-10, "Advanced Video Coding", Final Committee Draft, Document JVT-F100, December 2002.

[40] Iain E G Richardson, "H.264 and MPEG-4 Video Compression," John Wiley & Sons, September 2003, ISBN 0-470-84837-5.

[41] N. Omoigui, L. He, A. Gupta, J. Grudin, and E. Sanocki,

"Time-compression: system concerns, usage, and the benefits," Proceedings of ACM SIGHI Conference, May 1999, pp. 136-143.

[42] 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, March 2001, Vol. 11, No. 3, pp. 415-425.

[43] Shih-Yu Huang, "Improved techniques for dual-bitstream MPEG video streaming with VCR Functionalities," IEEE Transactions on Consumer Electronics, Vol. 49, No. 4, November 2003, pp. 1153-1160.

[44] Arch C. Luther, "Principles of digital audio and video," Artech House, 1997.

[45] Hervé Benoit, "Digital television : MPEG-1, MPEG-2 and principles of the DVB system," Oxford; Boston: Focal Press, 2002.

[46] Vasudev Bhaskaran, "Image and video compression standards : algorithms and architectures," Kluwer Academic Publishers, 1997.

[47] Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "Low-complexity and high quality frame-skipping transcoder," Proceedings of IEEE International Symposium on Circuits and Systems 2001, Sydney, Australia, May 6-9, 2001, pp. 29-32.

[48] Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, "New architecture for dynamic frame-skipping transcoder," IEEE Transactions on Image

Processing, Vol. 11, No. 8, August 2002, pp. 886-900.

[49] Kai-Tat Fung, Yui-Lam Chan and Wan-Chi Siu, “Low-complexity and high-quality frame-skipping transcoder for continuous presence multipoint video conferencing,” IEEE Transactions on Multimedia, Vol. 16, No. 1, February 2004, pp. 31-46.

[50] C. H. Huang, K. C. Yang, and J. S. Wang, "A low-cost unrestricted fast playback scheme for video streaming," IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 52, No. 7, July 2005, pp. 384-388.

[51] M. S. Chen and D. D. Kandlur, “Downloading and stream conversion: supporting interactive playout of videos in a client station,” Proceedings of the International Conference on Multimedia Computing and Systems, May 1995, pp. 73-80.

[52] S. Cen, “Reverse playback of MPEG video,” U.S. Patent 5739862, April 14, 1998.

[53] S. J. Wee and B. Vasudev, “Compressed-domain reverse play of MPEG video streams,” Proceedings SPIE Conference on Multimedia Systems and Applications, November 1998, pp. 237-248.

[54] S. J. Wee, “Reversing motion vector fields,” Proceedings IEEE International Conference on Image Processing 1998 (ICIP98), October

1998, pp. 209-212.

[55] Y.-P Tan, Y. Liang, and J. Yu, “Video transcoding for fast forward/reverse video playback,” Proceedings of IEEE International Conference on Image Processing, ICIP2002, September, 2002, pp. 713-716.

[56] Chang-Hong Fu, Yui-Lam Chan and Wan-Chi Siu, "Fast motion estimation and mode decision for H.264 reverse transcoding," Electronics Letters, Vol. 42, No. 24, November 2006, pp. 1385-1386.

[57] Chang-Hong Fu, Yui-Lam Chan and Wan-Chi Siu, “Efficient Reverse-Play Algorithms for MPEG Video Streaming with VCR Support,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 16, No. 1, January 2006, pp. 19-30.

[58] IS0/IEC JTC1/SC29/WG11 N3908, “MPEG-4 Video Verification Model Version 18.0,” 2001.

Available: http://ip.hhi.de/suehring/tml/download

[59] JM 7.6 Software [Online].

Available: http://www.stanford.edu/~esetton/H264_2.htm.