The Hong Kong Polytechnic University

Department
of
Electronic and Information
Engineering

Error Resilient Support in Video Proxy Over Wireless Channels

by

CHEUNG Hoi Kin

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

November 2006

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)

_____ _____ _____ (Name of student)

Abstract

The challenge of robust transmission is to protect compressed video data against hostile channel conditions while bringing little impact on bandwidth usage. There have been many error resilient video coding schemes proposed in the literature that attack the transmission error problem from different angles. One of the most common schemes is reference picture selection (RPS). So far, various RPS techniques have been investigated widely for use in real-time encoding. However, the RPS has not been examined in transmitting an already encoded MPEG bitstream, which have emerged as one of the indispensable video services over the Internet and 3G wireless networks nowadays. One straightforward approach in adopting the RPS scheme in an already MPEG encoded bitstream is to decode all the P-frames from the previously nearest I-frame to the current transmitted frame which is then re-encoded with a new reference; this can create undesirable complexity in the video server/proxy as well as introduce re-encoding errors.

Thus, in this thesis, some novel techniques are suggested for an effective implementation of RPS in an already MPEG encoded bitstream with the minimum requirement on the server/proxy complexity. All the proposed techniques will manipulate data in the MPEG compressed domain such that the computational burden of the video server/proxy can be greatly reduced. By effectively utilizing the new compressed-domain techniques, we develop an extremely efficient structure to handle various types of macroblocks in the video streaming server. The server or proxy classifies macroblocks in the requested frame into two categories – a macroblock without motion compensation (non-MC macroblock) and a motion-compensated macroblock (MC macroblock). Two novel macroblock-based techniques are used to manipulate different types of

macroblocks in the compressed domain and the server then sends the processed macroblocks to the client machine.

For non-MC macroblocks, we propose to reuse motion vectors and prediction errors from the existing and already encoded MPEG bitstream in order to avoid the additional computational burden in the server/proxy. A smart quantization and dequantization pair is proposed to maintain the reconstructed quality when the RPS scheme is applied to this scenario. The server also makes use of some new compressed-domain shifting and cropping operators to handle MC-macroblocks for the purpose of further reducing its computational complexity. Experimental results show that, as compared to the RPS scheme using the conventional technique, the new approach can reduce the required server/proxy complexity significantly.

It is exciting to report in this thesis that significant improvements in terms of server/proxy complexity and quality of reconstructed video can be achieved by employing our compressed-domain techniques. Undoubtedly, these techniques could become the trend for the research in adopting RPS in any already encoded MPEG video.

Acknowledgment

I would like to take this opportunity to express my sincere gratitude to my supervisor, Dr. Y.L. Chan, and co-supervisor, Professor W.C. Siu, for his continuous encouragement, guidance and care during the period that I worked on this thesis. They gave me information suggestions and valuable advice contributing to every success of my research. More importantly, I am deeply impressed with their hard working style and their willingness to devote to the advancement of science and research. This gives me a clear image of a great researcher should be and have inspired me to work hard on the thesis. It is beyond doubt that this will continuously influence my future research and career.

I would again like to express my sincere thank to Dr. Bonnie Law, Mr. Rainbow Fu, Mr. Bill Ip, Miss K.M. Au, Mr. Patrick Chan, Mr. K.C. Hui, Miss K.L. Lau, Miss Karen Wong, Mr. Michael Yiu and Dr. K.T. Fung. The sharing of ideas and experience with them has greatly contributed to make every success of my work. It has been a wonderful time to me in these years to work with them.

Meanwhile, I am glad to express my gratitude to the Department of Electronic and Information Engineering and the Centre of Multimedia Signal Processing for providing me a comfortable working environment, and a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (PolyU 5216/03E) for their generous financial support.

Above all, I am deeply grateful to my family for their constant love, encouragement and support. Without their understanding and patience, it is impossible for me to complete this research study.

Table of Contents

CERTIFICATE OF ORIGINALITY ..... **I**

ABSTRACT ..... **II**

ACKNOWLEDGMENT ..... **IV**

TABLE OF CONTENTS ..... **VI**

LIST OF FIGURES ..... **VIII**

LIST OF TABLE ..... **X**

ABBREVIATIONS ..... **XI**

LIST OF PUBLICATIONS ..... **XIII**

Accepted papers ..... *xiii*
Submitted papers ..... *xiii*

CHAPTER 1 ..... **1**

INTRODUCTION **1**

1.1 OVERVIEW 1
1.2 DIGITAL VIDEO AND COMPRESSION 2
1.3 ERROR RECOVERY FOR DIGITAL VIDEO TRANSMISSION 5
1.4 MOTIVATION AND PROJECT OBJECTIVES 8
1.5 ORGANIZATION OF THE THESIS 10

CHAPTER 2 ..... **12**

REVIEWS ON VIDEO COMPRESSION AND ERROR RECOVERY SYSTEMS **12**

2.1 INTRODUCTION 12
2.2 VIDEO COMPRESSION FUNDAMENTALS 13
2.2.1 Representation of Video Information *13*
2.2.2 Video Compression Techniques *16*
2.3 ERROR CONTROL IN VIDEO COMMUNICATIONS 25
2.3.1 Spatial and Temporal Error Propagation *26*
2.3.2 Error Detection *28*
2.3.3 Conventional Error Correction Techniques *29*
2.3.4 Error concealment at the decoder *31*
2.3.5 Error-resilient coding *34*
2.3.6 Encoder-Decoder Interactive Error Control *39*
2.4 ERROR RECOVERY SCHEMES FOR ALREADY MPEG ENCODED BITSTREAMS 45
2.5 CHAPTER SUMMARY 48

CHAPTER 3 ..... **50**

THE STRAIGHTFORWARD ERROR RESILIENCE SYSTEM FOR ALREADY MPEG ENCODED BITSTREAMS **50**

3.1 INTRODUCTION 50
3.2 STRAIGHTFORWARD APPROACH FOR ALREADY MPEG ENCODED BITSTREAMS ... 51

3.3 COMPLEXITY MEASUREMENT 56
3.4 CHAPTER SUMMARY 58

CHAPTER 4 60

THE PROPOSED TECHNIQUES FOR MACROBLOCKS WITHOUT MOTION COMPENSATION 60

4.1 INTRODUCTION 60
4.2 THE PROPOSED MB BASED ALGORITHM 60
4.2.1 Direct addition of quantized DCT coefficients for non-MC MBs 62
4.2.2 Modified Quantizer-dequantizer Pair for Non-MC MBs 66
4.3 CHAPTER SUMMARY 74

CHAPTER 5 76

THE PROPOSED DCT-DOMAIN OPERATORS FOR MOTION-COMPENSATED MACROBLOCKS 76

5.1 INTRODUCTION 76
5.2 MOTION VECTOR RE-COMPOSITION FOR MC MBS 76
5.3 NEW PREDICTION ERRORS IN MC MBS 80
5.4 DCT-DOMAIN OPERATORS FOR DRS 83
5.4.1 Shifting Operator 84
5.4.2 Chopping Operator 90

5.6 REGION TRACKING FOR $\overline{\text{DR}}$S 93
5.6 CHAPTER SUMMARY 96

CHAPTER 6 98

EXPERIMENTAL RESULTS AND DISCUSSIONS 98

6.1 INTRODUCTION 98
6.2 SIMULATION CONDITIONS 99
6.3 COMPUTATIONAL COMPLEXITY OF THE VIDEO SERVER 99
6.4 QUALITY PERFORMANCE 109

CHAPTER 7 122

CONCLUSIONS AND FUTURE DIRECTIONS 122

7.1 CONCLUSIONS OF THE PRESENT WORK 122
7.2 FUTURE DIRECTIONS 126
7.2.1 The Proposed RPS Scheme with Varying Quantization Factors. 127
7.2.2 RPS in already H.264 encoded bitstreams 129

REFERENCE 131

List of Figures

Figure 2.1 Various video representation formats, (a) 4:2:0, (b) 4:2:2, and (c) 4:4:4....15
Figure 2.2 A video picture with homogenous, edge and texture regions. 17
Figure 2.3 Block diagram of a generic video encoder. 17
Figure 2.4 Block diagram of a generic video decoder. 18
Figure 2.5 Zig-Zag sequence of DC and AC coefficients in an 8x8 DCT block....21
Figure 2.6 Typical real-time video communications system. 25
Figure 2.7 RPS using NACK mode. 44
Figure 2.8 RPS using ACK mode. 44
Figure 3.1 Failure of RPS operation due to adequate frame buffers. 52
Figure 3.2 Server-client model for transmitting an MPEG bitstream....54
Figure 3.3 Architecture of adopting pixel-domain RPS in the already MPEG encoded bitstream....55
Figure 3.4 Switch positions during re-encoding process. 55
Figure 3.5 Server complexity for CIF video sequences (GOP length = 15)....58
Figure 4.1. RPS in a non-MC MB. 61
Figure 4.2. RPS in a non-MC MB. 63
Figure 4.3. Decision levels (vertical line) and representation levels (dot) of the quantizer (a) with a dead zone and (b) without a dead zone. Both of them have a quantization step size of *L/2*. 71
Figure 4.4. The proposed MB-based techniques when the round-trip delay (RTD) corresponds to the duration of encoding time for two frames. 73
Figure 5.1. Motion vector re-composition in a MC MB. 77
Figure 5.2. Motion vectors of the four overlapping MBs with $MB^{n}_{(k,l)}$ 78
Figure 5.3. FDVS in a MC MB. 79
Figure 5.4. Block order. 81
Figure 5.5. Block level in frame *n-1*. 84
Figure 5.6. Contributions from the four overlapping blocks of (a) $r_1^{n-1,DR}$, (b) $r_2^{n-1,DR}$, (c) $r_3^{n-1,DR}$, and (d) $r_4^{n-1,DR}$. 85
Figure 5.7. Example showing matrix multiplication to extract a sub-block and translate it to the appropriate location. 87
Figure 5.8. Block level in frame *n*. 91
Figure 5.9. The situation of MC MBs adopting our proposed DCT-domain techniques (a) with region tracking and (b) without region tracking....95
Figure 6.1 The first frames of various sequences. (a) Mother and Daughter, (b) Salesman, (c) Calender, (d) Foreman, (e) Football, (f) Table Tennis, and (g) Carphone. 100
Figure 6.2. Number of TM operations for DCT-domain shifting and chopping operators with (a) zero vertical displacement, and (b) zero horizontal displacement. 105
Figure 6.3. The PSNR performances of the conventional technique and the proposed MB-based techniques in the case when frame 2 is corrupted and the round-trip delay (RTD) corresponds to the duration of encoding time for three frames. (a) the "Mother and daughter" sequence encoded at 537 kbits/s, (b) the "Salesman" sequence encoded at 1.14 Mbits/s, (c) the "Football" sequence encoded at 2.25 Mbits/s, (d) the "Calendar" sequence encoded at 1.64 Mbits/s, (e) the "Table

tennis" sequence encoded at 1.44 Mbits/s, (f) the "Foreman" sequence encoded at 1.12 Mbits/s, and (g) the "Carphone" sequence encoded at 457 kbits/s.....116
Figure 7.1. Representation level of quantization Q_s^{-1} with quantization factor = 6...128
Figure 7.2. Representation level of quantization Q_s^{-1} with quantization factor = 9...128
Figure 7.3. Representation level of quantization Q_s^{-1} with quantization factor = 3...128

List of Table

Table 4.1. Percentage of the non-MC MB for various sequences. 66
Table 4.2. PSNR performances after the corrupted frame for the direct addition technique and the conventional RPS using the re-encoding technique (only non-MC MBs). 69
Table 5.1. Matrices s_i^{ver} and s_i^{hor} for $r_1^{n-1,DR}$ 87
Table 5.2. Matrices s_i^{ver} and s_i^{hor} for $r_2^{n-1,DR}$ 89
Table 5.3. Matrices s_i^{ver} and s_i^{hor} for $r_3^{n-1,DR}$ 90
Table 5.4. Matrices s_i^{ver} and s_i^{hor} for $r_4^{n-1,DR}$ 90
Table 5.5. Chopping matrices $c_{h_i}^{row}$ and $c_{w_i}^{col}$ for $e_{(k,l),i}^{n,DR}$ 92
Table 6.1. The required numbers of TM operations for $mv_{(k,l)}^{n}$ with different combinations of horizontal (Δx) and vertical (Δy) displacements when the shifting and chopping operators are used. The maximum displacement of Δx and Δy are confined to ±8 pixels. 104
Table 6.2. Savings of using the proposed MB-based techniques as compared with the conventional re-encoding technique in RPS. 108
Table 6.3. Average PSNR performances for non-MC MBs of the succeeding frame after the corrupted frame by using different techniques in RPS. 117
Table 6.4. Average PSNR performances for all MBs of the succeeding frames after the corrupted frame by using different techniques in RPS. 118
Table 6.5. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD=1). 119
Table 6.6. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD =2). 120
Table 6.7. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD =3). 121

Abbreviations

ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
MPEG	Moving Picture Experts Group
JPEG	Joint Picture Expert Group
RPS	Reference picture selection
MCP	Motion-compensated prediction
MC MB	Motion compensated macroblock
non-MC MB	Non-motion compensated macroblock
DA	Direct addition
FDVS	Forward dominant vector selection
DR	Dominant region
$\overline{DR}$	Non-dominant region
LR	Lower right
LL	Lower left
UR	Upper right
UL	Upper left
MB	Macroblock
DCT	Discrete cosine transform
IDCT	Inverse discrete cosine transform
QP	Quantization parameter
1-D	One dimensional
2-D	Two dimensional
I-frame	Inter frame

P-frame	Predictive frame
B-frame	Bi-directional predictive frame
GOP	Group of picture
FEC	Forward error correction
ARQ	Automatic retransmission request
MDC	Multiple description coding
AIR	Adaptive intra-refresh
RTD	Round trip delay
ACK	Positive acknowledgement signal
NACK	Negative acknowledgement signal
VoD	Video on demand
FCS	Feedback control signaling
GPRS	General packet radio services
RTD_dist	Round trip delay distance

List of Publications

Accepted papers

1. Hoi-Kin Cheung, Yui-Lam Chan and Wan-Chi Siu, "Reference Picture Selection in an Already MPEG Encoded Bitstream", Proceedings of the International Conference on Image Processing (ICIP2005), Vol. 1, pp. 793-796, September 2005, Italy.

2. Hoi-Kin Cheung, Yui-Lam Chan and Wan-Chi Siu, "Robust and Fast Global Motion Estimation for Arbitrarily Shaped Video Objects in MPEG-4" Proceedings of the International Conference on Signal Processing (ICSP'2004), pp. 1163-1166, August-September 2004, Beijing, China.

Submitted papers

1. Hoi-Kin Cheung, Yui-Lam Chan and Wan-Chi Siu, "Efficient Reference Picture Selection Algorithm for an Already MPEG Encoded Bitstream", submitted for possible publication in IEEE Transactions on Circuits and Systems for Video Technology (Revised).

Chapter 1

Introduction

1.1 Overview

The popularity of digital video has been growing enormously in recent years. Nowadays, not only traditional video cameras can shoot video scenes, but also almost all mobile phones and digital cameras have the ability of capturing digital video clips. These digital devices change our habit from photo taking to video shooting. The digital video clips can then be easily shared with others through the Internet or mobile phone networks. Besides, the continued advance in high speed Internet and mobile communications networks enable more flexible for video transmissions. Hence, text and still images cannot satisfy people's desire anymore. We are looking forward to having high quality movie entertainment services on personal computers, televisions, or even mobile phones on the street. Owing to the enormous amount of video data to be stored, or transmitted through the bandwidth-limited networks, video compression is necessary to reduce the storage and channel bandwidth requirements. Many video compression schemes have been developed to fulfill this objective. The schemes aim at removing data redundancy. However, some of these techniques adopted in the video compression scheme are extremely vulnerable to transmission errors. Moreover, one inherent problem of any communications system, especially for wireless networks, is that they suffer from channel noise and information lost. Therefore, if one want to enjoy video streaming service through various networks,

an effective error recovery scheme is necessary. Apart from real-time video encoding in which data arrives from a live source at a fixed frame must be compressed in real time and transmitted at that speed, videos in archive such as movie and TV series on discs or tapes are commonly requested for playback. To save storage space in a server, they are encoded and stored in different kinds of reliable media without any special measure for handling errors and losses during transmission. Therefore, an effective error recovery scheme specially designed for already encoded video bitstream transmission is also very important.

In this chapter, the fundamental of the digital video and the motive for video compression will be introduced. An overview of some existing compression schemes will then be summarized. As the transmission of digital video is becoming very popular, we will discuss the problems of sending digital video through error prone networks. Some existing error recovery schemes for real-time video encoding and transmission will also be briefed. Finally, the motivation, objectives and the organization of this thesis will be presented.

1.2 Digital Video and Compression

Traditionally, analog video is captured by a number of light sensitive films moving in a fast speed sequentially whilst each film captures an instant time of the scene. During video playback, pictures of recorded films are projected on screen sequentially with the same speed as recording. To ensure smooth motion, about 25-30 frames per second (frame rate) must be used if the scene is captured by a camera and not generated synthetically. In the modern world, as a result of advances in new digital technologies, analog video is converted to digital form,

called digital video, and stored in storage devices such as hard disks, DVD, VCD, etc. Digital video is actually formed by a sequence of digital pictures which are created in the form of a two dimensional matrix of individual picture elements called pixels. Advantages of digital video over the traditional one include no fading along time so that quality can be preserved virtually forever. It is more flexible on storage form and more convenient for transmission through different types of networks. With the help of digital video processing tools, digital video can be easily modified or edited. Video enhancement, restoration and object segmentation are also typical applications of digital video processing. It also allows limitless innovative special effects on digital video.

However, digital video induces large data rate. To deal with the huge data size of digital video, research has made significant progress in lossy compression techniques in order to reduce the data size for efficient transmission and storage. The interest in compression for digital video has been shown in international efforts for video compression at different target bit rates or applications. Developing an international standard requires collaboration among many parties from different countries with different infrastructures and commercial interests, and an organization that can support the standardization process as well as enforce the standards. There are two major teams of developing digital video coding standards. They are the ISO/IEC MPEG (Moving Picture Experts Group) and ITU-T VCEG (Video Coding Experts Group). MPEG is a Working Group of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). The goal of MPEG is to develop standards for compression, processing and representation of moving pictures and audio. It has been responsible for the successful MPEG-1 (ISO/IEC 11172) [1] and

MPEG-2 (ISO/IEC 13818) standards [2], which have given rise to widely adopted commercial products and services, such as VCD, DVD, digital television, MP3 players, etc. On the other hand, the VCEG is a working group of the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) which leads to a series of important standards for video communications over telecommunication networks and computer networks. Starting from the first H.261 [3] videoconferencing standard and following on with the very successful H.263 [4], two other standardizations, informally known as H.263+ and H.263++, have been created to extend the capabilities of H.263. The most powerful and advanced video compression standard at this moment is H.264 or known as MPEG-4 Part 10 [5] which is developed by the joint effort of the ITU-T VCEG and the ISO/IEC MPEG. All the standards have the common endeavor to reduce the storage requirement of digital video. It is possible since the uncompressed digital video always contains a large amount of redundancy. Certain information in digital video is insensitive to human eyes. Thus, various compression schemes preserve the non-redundant information but discard the insignificant information. In general natural video, scene and objects in the video content changes smoothly and gently over time, so successive frames are often highly correlated. In other words, typical video signals contain a large amount of temporal redundancy. The redundancy can be removed to achieve compression of video data. Removing this redundant information and representing video data in a more efficient from can significantly reduce the capacity required to store or transmit video. Motion estimation and compensation algorithm is widely used in most of the modern video compression standards. This algorithm 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 past frame, or called 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 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.

1.3 Error Recovery for Digital Video Transmission

The aforementioned video coding techniques are used to reduce the bandwidth requirements and enable the transmission of video information over band-limited communications channels. However, the process of removing temporal redundancy means that the coded current frame depends on its reference frame. If any part of coded data is lost or corrupted, the decoded sequence may be significantly distorted. In other words, the encoded video bitstream is extremely delicate to errors. Even a single bit of error in the video bitstream can cause tremendous video quality degradation during decoding. Besides, the inherent problem with any communications system, especially for wireless networks, is that information may be lost, disordered or altered during transmission. Transmitting coded video data through networks are extremely unreliable. Transmission errors can roughly be categorized into two types, random bit errors

and erasure errors. Random bit errors, including bit inversion, bit insertion and bit deletion, are mainly caused by the non-ideal of physical channels. For instance, signal in both the wired and wireless network connection can be affected by strong electromagnetic interference or hardware electrical noise, which results in altering the transmitting bits. Depending on the coding schemes and the affected information content, impact of random bit errors can range from negligible to objectionable. In case of fixed length coding, very limited influence would occur since a single bit error is restricted in one particular code word only. On the contrary, when variable length coding (VLC) is adopted, even a single bit error can lead to loss of synchronization and affect the decoding of consequent code words. Unfortunately, almost all video compression standards adopt the higher coding efficient entropy coding such as Huffman and Arithmetic coding, which is a kind of VLC. In this case, the video decoder is unable to detect corrupted incoming bits immediately until an invalid code word is detected. Note that the invalid code word does not exist in the code word table. As a result, errors will be spread and this phenomenon is called error propagation, which causes difficulties to the error recovery process. The error propagation will be stopped until the next synchronization code word. In some cases, even after the synchronization code word is located, the decoded information may still be unusable since its temporal or spatial location has already been lost. On the other hand, the impact of erasure errors, which has similar effect as random bit errors in VLC, is much more destructive than random bit errors due to the loss or damage of a contiguous segment of bits. This type of error can be caused by packet drop or loss in routing systems due to traffic congestion, system failures for a short time or signal fading in wireless networks. The signal strength of a wireless network is highly depended on receiver location, weather conditions,

vehicle moving speeds, etc. Therefore, transmitting encoded video data through wireless networks is unreliable.

Apart from VLC, the adoption of predictive coding in the video compression scheme is another key factor that threatens the video quality when transmission errors exist. The predictive coding encodes the current frame by using its previously decoded frame as a reference. It means that the corrupted area in the decoded frame can propagate to other decoded frames if they are temporally predicted from the corrupted frame. This temporal error propagation will not stop until the next synchronization frame (I-frame). Note that an I-frame is coded independently without predicting from other video frames. Although its coding efficiency is lower than those of the predictive frame (P-frame) and the bi-directional predictive frame (B-frame), it can terminate temporal error propagation from the preceding frames and can be acted as an access point for video playback. Hence, the robustness to transmission errors of the encoded video bitstream is highly depended on the number of frames between every two I-frames, which is called the Group of Picture (GOP) length. In low bit-rate video applications such as video streaming over wireless networks, it is common to have lengthy GOP with the aim of reducing the bandwidth requirement. Once transmission errors exist, they remain visible on decoded video for a long period of time. The degradation is annoying to end-users and definitely unacceptable.

Over the past decade, many error recovery schemes tackling the video transmission over error prone networks have been proposed in the literature. Our work focuses on techniques of the error tracking [6] and reference picture selection (RPS) schemes [7-9], which are adopted in the H.263 Annex N [4] and

the MPEG-4 standard [10]. The error tracking scheme attempts to stop error propagation within a GOP by adopting intra-refresh, while RPS employs only an uncorrupted previous frame as a reference. These schemes are well designed for the circumstance of real-time video transmission. For the error-tracking scheme, the encoder collects the information of the channel condition and feedback from the client. The information of the channel condition can indicate the current traffic status and the error rate of the network. Meanwhile the encoder can locate the temporal and spatial occurrence of an error by using the feedback signal from the client. The encoder then evaluates the extent of propagated errors and decides which MBs should be intra-coded. However, intra-coding is very inefficient in terms of compression ratio and thus it results in rising of the bandwidth requirement. Instead of using intra-refresh, the RPS scheme encodes the video frame by adaptively selecting its reference, which is sure to be correctly reconstructed at the decoder side. Similar to the intra-refresh approach, by analyzing the feedback information from the decoder, the encoder can detect the temporal and spatial occurrence of an error and its extent at the decoder. Hence, the most suitable reference frame can then be determined for encoding the next video frame. More details on the existing error recovery schemes will be discussed in Chapter 2.

1.4 Motivation and Project Objectives

The mentioned techniques are intended for real-time video transmission. Nowadays, many video services and applications, such as video on demand (VoD), digital TV, and distance learning, will use pre-encoded video bitstreams for storage and transmission rather than real-time encoded video. In coding of

live video for transmission, the video system might use feedback from the decoder to encoder to adapt its coding strategies in the presence of noise. In this way, the error tracking and reference picture selection schemes are accomplished by the source encoder.

However, for pre-encoded video, the lack of flexibility makes it difficult to adapt the coding strategies for the bitstream. For instance, the error tracking scheme adopts intra-mode coding of MBs to stop temporal error propagation. Based on the feedback signal, the server can identify the temporal and spatial occurrence of an error. It then finds out the extent of propagated errors and converts the affected inter-coded MBs to intra-coded MBs. To do this, the pre-coded video bitstream is fully decoded and it includes the processes of motion compensation, dequantization, inverse transformation, and variable length decoding. The coefficients of those intra-coded MBs to be transmitted are then transformed, requantized, scanned and variable-length encoded to create the new intra-coded MBs. These processes can be implemented in the server or proxy. In the following, for the sake of simplicity, the words “video server” and “video proxy” represent the same thing. This approach will complicate the video server. On the other hand, RPS can also stop temporal error propagation by allowing the encoder to select one of several previously decoded frames as the reference for motion compensation. In this approach, a feedback sent from the decoder to the video encoder signifies that a given frame has been damaged by transmission errors. To stop temporal error propagation, the server will detect which frame uses the corrupted frame as a reference frame. Then, it will re-encode the affected frame with a different and uncorrupted reference frame. The server needs to decode all the P-frames from the previously nearest I-frame to the

affected frame which is then re-encoded with a new reference; this can create undesirable complexity in the server as well as introduce re-encoding errors. In this thesis, we propose a compressed-domain scheme for transplanting RPS to the pre-encoded video bitstream in order to reduce the computational requirement of the server and the quality degradation of the reconstructed video arising from re-encoding.

1.5 Organization of the thesis

Prior to embarking on the description of the main research in this thesis, a review of error control techniques for video communications is given in Chapter 2. This review has been confined to methods which can be categorized into the conventional error correction, error concealment coding, error resilient coding and interactive error control coding techniques. Of course, some general introductory video coding materials are included for the purpose of clarifying certain definitions employed in later chapters.

Chapter 3 mainly discusses the impact of directly adopting RPS in an already MPEG encoded bitstream on server complexity. We argue that straightforward implementation requires much higher server complexity for these decoding and re-encoding processes. Besides, the video quality suffers from its re-encoding process, which introduces additional degradation. In Chapter 4, two new techniques, called direct addition and modified quantization/dequantization pair, are proposed to perform RPS in the already MPEG encoded bitstream. It is found that the performance of RPS used in the pre-encoded video bitstream could be substantially improved since the proposed techniques are mainly

operated on the quantized DCT domain in order to alleviate the computational requirement of the server and the quality degradation of the reconstructed video arising from re-encoding. Chapter 5 investigates some DCT-domain operators to further improve the performance of RPS in the already MPEG encoded bitstream by adopting forward dominant vector selection (FDVS) with the new DCT-domain operators. After describing the proposed compressed-domain techniques in the RPS scheme, various comparisons of these techniques are undertaken in Chapter 6 to give a clear evaluation of adopting different techniques in RPS.

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 this area.

Chapter 2

Reviews on Video Compression and Error Recovery Systems

2.1 Introduction

Recent advances in compression and networking technologies have stimulated the popularity of digital video applications. Such applications include video telephony, video conferencing, video-on-demand, video surveillance, video storage, etc. Due to the enormous size of video data, video compression techniques are becoming extremely essential in order to allow digital video to be transmitted over most networks and handled by processors. For effective video communications, reduction of video data size is only one of the necessary steps. Besides, handling errors and losses in a communications network is also crucial to the success of video communications applications. The extensive use of predictive and variable-length techniques in video coding makes compressed video especially vulnerable to transmission errors, and successful video communications in the presence of channel errors needs careful designs of the encoder and decoder with error control and recovery schemes. In the past decade, numerous techniques have been proposed for this purpose. In this chapter, we will review both video compression techniques and error recovery schemes.

This chapter is organized in the following order. First of all, we will start introducing the digital video components. It is the most basic element building up

every digital picture and video. Then, a review of compression techniques adopted in the existing video coding standards will be given. After that, we will discuss the effects of errors occurring in coded video data or MPEG video. We will also review techniques for error detection in a video communications system. Then, we will present various approaches that have been developed for error control in video communications. It includes error-concealment and error-resilient coding. Techniques that rely on encoder and decoder interaction will also be discussed. The final section will review some new techniques for adopting error recovery schemes in an already MPEG encoded bitstream.

2.2 Video Compression Fundamentals

Digital video is video information that is stored and transmitted in digital form. However, the bit rate of digital video is very huge when an analog video signal is converted into digital form at an equivalent quality. This bit rate is too high for most networks and processors to handle. The solution is to compress the digital video information, and to store or transmit it in a compressed form. Compression techniques for digital video have been continually improving over the last two decades. International standards now provide standard techniques for digital video coding. In the following sections, some video compression techniques will be given.

2.2.1 Representation of Video Information

Natural video scene is composed of multiple objects with their own characteristic shape, depth, texture and illumination [5]. To human observer, a real scene appears continuously at temporal and spatial dimensions. To capture a real scene into digital form, it is accomplished by sampling the natural scene temporally and spatially. At every interval instant of time, a scene is captured and sampled using a light sensitive device (e.g. CCD or CMOS sensor array). This is a silicon chip in digital cameras consisting of a two-dimensional gird of light-sensitive cells. The number of cells determines the spatial resolution and the dimension of video pictures. In principle, a video picture is a rectangle with the sampling points arranged in a rectangular grid. This basic element of a picture, the sampling point, is generally abbreviated to “pixel”. By placing pixels close together, the observer will perceive a continuous picture. Thus, the more the number of pixels it has, the higher the resolution a video picture it represents. Consequently, a smoother picture as well as better quality of video can be obtained. Besides, the temporal resolution of video is another important parameter affecting the overall quality critically. To represent a moving video, it is done by taking a rectangular “snapshot” at periodic time intervals. A higher temporal sampling rate (frame rate) gives apparently smoother motion in the video scene. But it needs more pixels to be captured and stored. A typical frame rate is about 30 frames per second (fps) in order to provide smooth motion. On the contrary, the frame rate less than 10 fps is sometimes used for very low-bit-rate video communications. A low frame rate leads to what is called flicker which is caused by the previous image fading from the eye retina before the next picture is displayed. Especially, ignoring flicker and artificial effect may become visible in fast-motion regions of the video scene.

As mentioned before, a pixel is the most fundamental element constructing a picture and video. For every pixel in video pictures, the visual information is described by either the grayscale intensity level or color-space components. The level is typically sampled into the range from 0 to 255 intensity steps in digital video. In grayscale picture, 0 represents a black pixel while 255 represents a white pixel. Color pictures, on the other hand, are represented by at least three components per pixel, which provide extra color information. Different color spaces such as *RGB*, *CMY*, *YCrCb*, etc. can be employed to represent color accurately. The most typical color space used in video coding standards is *YCbCr* where *Y* represents the luminance (brightness) level of the picture and *CbCr* are the chrominance (color) information. Since the human visual system is less sensitive to chrominance than to luminance, the *Cr* and *Cb* components can be represented with a lower spatial resolution than the *Y* component in *YCbCr* 4:2:0 format, as shown in Figure 2.1 (a). The advantage of the 4:2:0 format is that the amount of data required to represent the chrominance information is reduced, but no obvious effect on visual quality will show. For high-quality video, the higher chrominance resolution color space (*Y, Cb, Cr*) 4:2:2 and 4:4:4 can be adopted as shown in Figure 2.1(b) and (c) respectively.

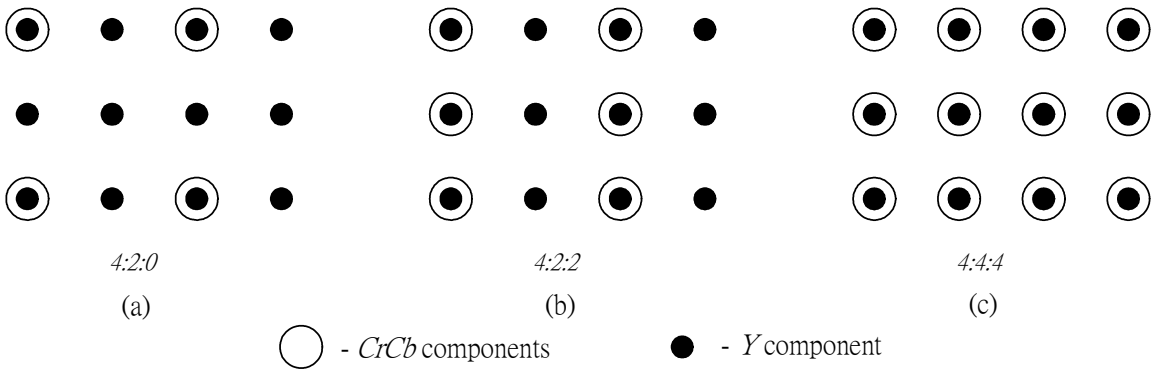


Figure 2.1 Various video representation formats, (a) 4:2:0, (b) 4:2:2, and (c) 4:4:4.

2.2.2 Video Compression Techniques

Digital video has a substantial amount of data. There is a need to reduce this data rate such that it can be integrated into the existing communications systems. This means video compression is necessary. A number of video coding techniques has been developed for the past decades. All of the advanced and efficient video coding techniques are built according to the characteristics of natural video and human visual perception. They exploit some of the inherent redundancy in still images and moving sequences in order to provide significant data compression.

Most sensory signals contain a substantial amount of redundant information [11]. In general, real and natural video is smooth spatially and temporally. A video picture may have plain (homogeneous) areas, texture patterns, object edges and curves (Figure 2.2). In the homogeneous area of a picture, the values of neighboring pixels are often very similar. The homogeneous area often dominates the whole picture. Technically, it is called spatial redundancy and can be removed for the purpose of compression. In the temporal domain, objects in the natural video scene move gently. Motion can be classified as two types, the global motion due to the camera movement and the local motion due to the objects' self-actions. Thus, contents in one picture are most likely to be found in the next or previous video pictures. A high correlation or similarity between frames of a video sequence introduces temporal redundancy, which is especially high in the stationary part of a video sequence such as static background. In this case, the amount of information can be reduced considerably by exploiting the similarity between frames.

Video coding standards exploit both temporal and spatial redundancy to achieve compression and share a number of common techniques. Most video coding standards assume a model that employs transformation, quantization, entropy coding and block-based motion estimation and compensation. In the following, we review the main components of this model. Figure 2.3 and Figure 2.4 show the block diagrams of the encoding and decoding systems respectively.

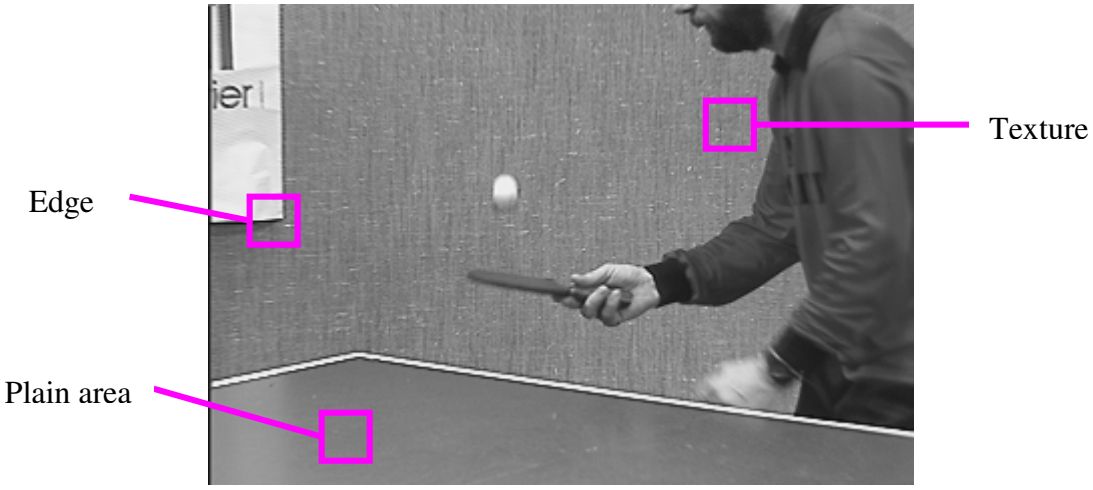


Figure 2.2 A video picture with homogenous, edge and texture regions.

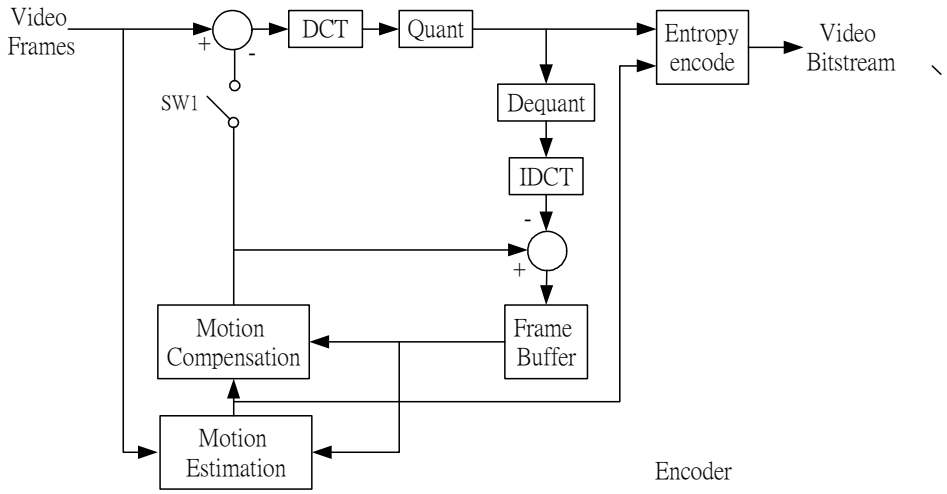


Figure 2.3 Block diagram of a generic video encoder.

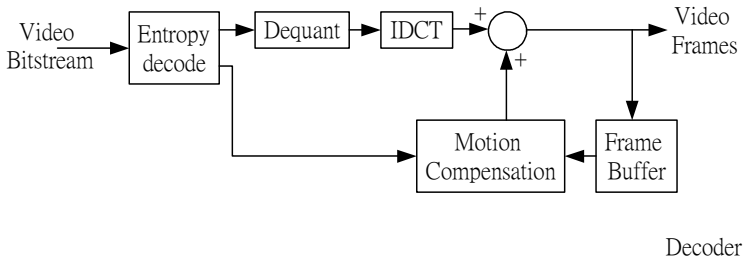


Figure 2.4 Block diagram of a generic video decoder.

To remove spatial redundancy, discrete cosine transform (DCT) is the most popular technique utilized in a video encoder, which aims at compacting spatial signal into fewer coefficients for efficient compression. On the other hand, predictive coding including motion estimation and compensation attempts to reduce temporal redundancy by exploiting the similarities between neighboring frames. The processes of motion estimation and compensation are to predict the current frame by using its previous frame as a reference so that lesser bit is required to represent a new frame. Before the bitstream is sent to the decoder, lossless entropy coding is used to remove statistical redundancy in the data and produce a compressed bitstream that may be transmitted or stored.

Three major types of frames exist in various standards. They are intra-frames (I-frames), predicted frames (P-frames) and bidirectional-frames (B-frames). I-frames are acted as an access point for video playback as it is encoded without predicting from other frames. In another words, I-frames do not remove any temporal redundancy. Consequently, the level of compression obtained with I-frames is relatively small. In principle, it would appear to be the best if the number of I-frames is limited to the first frame relating to a new scene in a movie. However, I-frames must be used at regular intervals in order to allow for the possibility of the contents of an encoded I-frame being corrupted during

transmission. Since P-frames are predicted from an I-frame, a complete movie would be lost due to corruption of its reference, which is totally unacceptable. Therefore, I-frames are inserted into the compressed bitstream relatively frequently. The group of frames between successive I-frames is known as a group-of-pictures (GOP). In contrast, the encoding of a P-frame attempts to reduce temporal redundancy by predicting a preceding I-frame or a preceding P-frame. As indicated in Figure 2.3, 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 acted as a reference by other frames.

The encoding of an I-frame is almost identical to that of the JPEG algorithm, which is the image compression standard developed by the Joint Picture Expert Group. The techniques employed in JPEG are to exploit spatial redundancy and statistical redundancy of an image. The first step is to divide an image into a number of blocks of 8x8 pixels. A block of pixels is transformed into another domain to produce a set of transform coefficients, which are then coded and transmitted. The most popular transform used in video coding standards is discrete cosine transform (DCT). After transformation, the quantization process is used to discard unimportant transform coefficients, which is the only

irreversible process during encoding. Zig-Zag scanning is then used to group non-zero quantized coefficients together prior to entropy encoding. The block diagram in Figure 2.3 shows the production of an I-frame in which the switch *SW1* is open.

Generally, natural video scene is composed of one or more foreground objects and a background. Typically, pixel values in a frame are often similar. The DCT is a transformation tool converting the original spatial signal into frequency domain. In the viewpoint of video or image compression, the DCT can help separating the image into spectral sub-bands of different importance with respect to the human visual system [11]. It can also pack the energy of the signal into a few of coefficients [11]. In a real picture, an image block can often be represented with a few low-frequency DCT 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 upper left 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 upper left to lower right of the 8x8 block. Hence the upper left part of the transformed block contains the low frequency coefficients while the lower right part of the block contains the high frequency coefficients as shown in Figure 2.5. Both coefficients are then quantized using uniform or non-uniform quantizers, each with a different stepsize. The stepsizes for different coefficients are specified in a quantization weighting matrix. The matrix is designed according to the visual sensitivity to different frequency coefficients. In the quantization weighting matrix, the higher orders of DCT coefficients are

quantized with coarser quantization stepsizes than the lower frequency ones due to the insensitivity of the human visual system to high frequency distortions [5]. Such arrangement can have further bandwidth compression. Besides, one can further scale the matrix to adjust the stepsizes in order to achieve the desired bit rate. The scale factor is called the quantization parameter (QP) in the standard video coder.

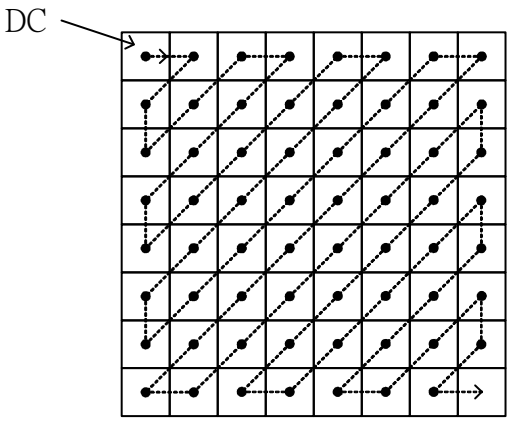


Figure 2.5 Zig-Zag sequence of DC and AC coefficients in an 8x8 DCT block.

For converting the quantized DCT coefficients into their binary representations, the quantized DCT coefficients are converted into a one-dimensional (1-D) array following the zig-zag order, as depicted in Figure 2.5. This scan arranges the low-frequency coefficients before the high-frequency coefficients. Since many quantized coefficients become zeros in a typical image block, it is not efficient to encode the coefficient individually. Instead, after the zig-zag scan, it is easy to code the resulting row of numbers efficiently with run/level and variable length coding techniques. In various video-coding standards, they define a number of tables with code words to be used for specific patterns in a coefficient data

sequence. The philosophy is to use very short codes for patterns that occur very often in the sequence and vice versa. It is noted that the encoder treats DC coefficients separately. The DC coefficient is a measure of the average pixel values over an 8×8 block. Therefore, it is the largest and the most important coefficient, its resolution is kept as high as possible during quantization. Due to the small physical area covered by each block, the DC values of adjacent blocks are very similar. To further improve the coding efficiency, the DC coefficient of a block is predicted from the DC value of its previously encoded block, and then the DC prediction error is quantized and encoded using entropy coding. The prediction can even be performed from either the block above or the block to the left in some recent video coding standards. The direction of the prediction is adaptive and is selected based on comparison of horizontal and vertical DC gradients of surrounding blocks.

The coding of P-frames exploits the strong correlation between the consecutive frames by using motion estimation and compensation. The complexity is remarkably increased as compared with that of I-pictures since the motion estimation process demands huge computational requirement. Given a current frame, it is firstly divided into a number of non-overlapped blocks. In most video coding standards, the block size for motion estimation may not be the same as that for DCT. Typically, motion estimation is performed on a larger block known as a macroblock (MB). Now, 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 preceding I- or P-frame (reference frame). If a close match is located, 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. The predicted MB is obtained from the reference frame based on the motion vector using motion compensation. Then, the prediction error 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. When the switch in SW1 is close in Figure 2.3, the encoding process of a P-frame is activated. Figure 2.4 also depicts the corresponding decoder.

Each MB has one motion vector. It consists of two motion vector components, the horizontal one first, followed by the vertical one. They are coded independently. Since motion vectors tend to be highly correlated from MB to MB, motion vectors are coded differently with respect to previously decoded motion vectors in order to reduce the number of bits required to represent them. Each component of the differential motion vector is coded by using a variable length code word. To enhance the efficiency of motion compensation, half-pixel or quarter-pixel motion estimation is adopted in various video coding standards. As mentioned above, those MBs that are coded with motion estimation and compensation is known as an inter-MB. In this case, the encoder requires fewer bits to code the prediction error block than the original image block. On the other hand, an intra-MB is defined as a MB that is coded directly using transform coding only. It is noted that not all the MBs in P-frames are coded using motion estimation and compensation. In general, an inter-MB has advantage in terms of coding efficiency as compared with an intra-MB. However, there are some reasons for using intra-MBs in a P-frame. First, for scene change or violent

motion in scene, the prediction error of the inter-MB may not be less than the original pixel block of the intra-MB. Hence, intra-MBs may be coded at lower bit rates. Second, intra-MBs have a better error resilience to channel errors. If channel errors exit, the error propagates into subsequent frames. If that part of the picture is not updated, the error can persist for a long time. In principle, the intra/inter coding decision depends on whether motion-compensated prediction can substantially reduce the prediction error. The straightforward way to do this is to code the MB as intra-MB and compare it with the total number of bits required when coded as inter-MB with the same quantizer parameter. The mode that needs fewer bits is chosen.

The major difference between coding B and P-frames is that both past and future frames can be acted as references for inter-MBs. 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 combined motion compensated predictions. Since the process of motion estimation performs twice, the encoding of B-frames 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.

2.3 Error Control in Video Communications

Figure 2.6 shows a real-time video communications system. An input video is first compressed by a video source encoder to the desired data rate. The data may be transmitted directly to the network if it can guarantee error-free transmission. Otherwise, a channel encoding process is necessary in order to protect the compressed video from transmission errors. At the receiver side, the received bitstream is undergone channel decoding and its output is then input to the video decoder to reconstruct the original video.

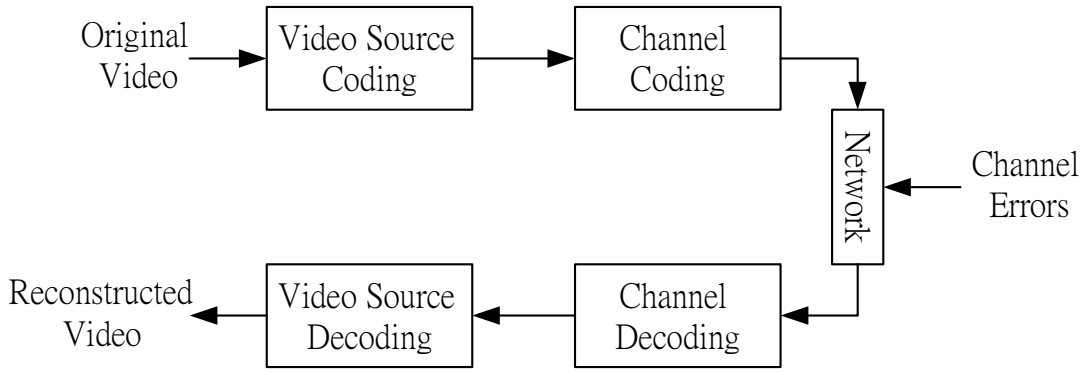


Figure 2.6 Typical real-time video communications system.

Error handling in video communications is a very challenging task. The reason is that compressed video bitstreams are extremely vulnerable to transmission errors due to the extensive use of temporal predictive coding and variable-length coding in the source encoder. To combat against transmission errors during video communications, the channel encoder adds redundancy to the compressed bitstream such that it is possible to detect and correct errors. Thus, in the real-time video communications system, the source encoder reduces

redundancy of the video source as much as possible for a given distortion. On the other hand, the channel encoder adds redundancy to the compressed bitstream to enable the correction of transmission errors. They work in different directions. Therefore, the challenge of robust transmission is to protect compressed video data against hostile channel conditions while bringing little impact on bandwidth efficiency.

2.3.1 Spatial and Temporal Error Propagation

Due to the use of predictive coding and VLC, the effect of transmission errors of coded video is never local. Errors occurring within a coded video picture can propagate spatially and temporally when the sequence is decoded. Even a single bit of error can ruin the whole picture or sequence of pictures.

There are three major reasons that cause spatial error propagation. First, a single bit error can desynchronize the coded information with the VLC bitstream. In this situation, DCT coefficients and motion vectors after the error are undecodable until the next slice. Since MBs in a slice are coded in order from left to right, the MBs to the right of the erroneous MBs may also be corrupted in the decoded frame. Second, due to the use of differential coding in DC coefficients, an error in a DC coefficient will incur all subsequent DC coefficients in the slice to be wrongly decoded. Since DC coefficients carry the most important visual information, conspicuous artifact is spread to the right of the erroneous MBs. Third, similar to the previous case, motion vectors are encoded differentially with respect to the motion vectors in the previously encoded MB. An error in a single

motion vector will make all subsequent motion vectors to be wrongly decoded and it will cause spatial error propagation again. It is noted that an incorrect motion vector in the particular MB means that it may be predicted from the wrong area within a reference frame. In principal, the error should only be propagated within a single slice since a resynchronization codeword is inserted in the video bitstream at the start of each slice. However, transmission errors such as packet loss may cover more than one slice. In this case, the lost packet will result in observably large erroneous area in the decoded frame.

Moreover, the use of motion estimation and compensation in video coding standards pose an even more severe problem, namely temporal error propagation. This predictive technique adopted in P or B-frames removes temporal redundancy in successive frames by encoding only residual errors between a current frame and a motion-predicted frame. In MPEG, I- and P-frames are used as reference frames for P and B-frames. Therefore, any corruption in a reference frame can propagate to its subsequent frames. In other words, the MB in each P or B-frame that is predicted from the erroneous area of the reference frame will also be incorrectly decoded. Spreading of the corrupted area may increase or decrease in size due to the process of motion compensation. Generally, motion compensation always causes the corrupted area to spread out since MBs in the surrounding area are predicted from the corrupted area in the reference frame. On the contrary, some of the distortion may be removed in subsequent predicted frames when marcoblocks are predicted from error-free areas. Besides, some MBs in P- and B-frames may be coded as intra-MB and these MBs will not be affected. The temporal error propagation will be completely stopped at the beginning of the next GOP since it

is started with an independently coded I-frame. Actually, the extent of temporal propagation that is affected by the original transmission error depends on the picture type of the erroneous frame. In the worst case, when an error is occurred in an I-frame, frames in the same GOP may all be affected. The video degradation will be extremely severe. For P frames, all P- and B-frames in the GOP after the erroneous frame may be affected. The number of affected frames, of course, depends on the position of the erroneous P-frame in the GOP. In contrast, no temporal error propagation is occurred for an erroneous B-frame as it is not used as a reference for the others.

2.3.2 Error Detection

Before any error control and recovery scheme can be carried out, it is necessary to detect whether and where a transmission error exists in the video bitstream. One possible way to perform error detection is to check whether the received code word is in the decoding table. If this is the not case, a transmission error is detected at the decoder since not all the possible code words are legitimate [12]. However, this approach cannot recognize an error immediately until an illegal code word is detected. It makes the decoder difficult to perform appropriate action at once. An alternative for detecting lost or damaged packets in transport level is to add header information such as check sum and/or packet sequence number fields into the packet header. Although this method is more reliable, it expenses additional channel bandwidth. Another approach [13-14] for error detection without increasing channel bandwidth is by exploiting characteristics of natural video signals. Due to the smoothness property of video signals, the gradient of pixel values is used for detecting the erroneous data. Once a sharp

change in pixel gradient is discovered, the area is regarded to be damaged. Obviously, both aforementioned techniques can be used together in practical systems. In this thesis, we will put more focus on how to recover a corrupted video bitstream. We assume that error detection is applied before any action of error control and the locations of errors are also known to the decoder on the rest of this thesis.

2.3.3 Conventional Error Correction Techniques

In general, video data can be regarded as conventional data transmitting over networks. Hence some conventional data correction techniques can be extended directly to a compressed video bitstream. For instance, forward error correction (FEC) is well known for both error detection and correction in data communications [15]. In the H.261 standard [3], an 18-bit FEC code is inserted to each 493 bits of video data to form a frame for error detection and correction with the expense of coding efficiency. The FEC code is able to correct a single bit error and to detect two bit errors in each frame. The same mechanism can also be employed in H.263 [4]. Unfortunately, the error burst is usually longer than two bits in wireless networks or the Internet. This FEC method is not very successful for transporting H.263 video bitstream over these types of networks. Hence it is seldom used. Besides, it is much more difficult to apply FEC for packet-based transmission since several hundred bits must be recovered when a packet loss occurs.

An alternative is to use the upper-layer transport protocol. This transport protocol can exercise error control in the form of automatic retransmission request (ARQ) which requests retransmission upon the detection of lost or overly delayed packets. This is done in TCP. Retransmission has been considered very successful for non-real-time data transmission, but it is generally unacceptable for real-time video communications due to the long delay incurred. In order to shorten the delay, Zhu [16] proposed a system that carries out the retransmission without waiting for feedback acknowledgement. All video packets are transmitted more than one time from the server to increase the chance of safely arriving. In the decoder side, duplicated packets are discarded automatically. Although this approach can relieve the long delay for real-time applications, it increases network bandwidth requirement. Ohta [17] introduced an approach that is similar to the technique in TCP/IP. By inserting the sequence number to the subfield of the video packet header, lost of packet can then be detected easily. Besides, disordered packets can be re-arranged according to the sequent number. In short conclusion, both FEC and retransmission mechanisms attempt to detect, correct and if necessary, retransmit corrupted data. Such mechanisms either incur low coding efficiency or introduce the long delay. They cannot take the advantages of characteristics of video signals.

In the literature, there have been many techniques to attack the transmission error problem by considering the properties of video signals. In the following sections, we categorize these techniques into three classes. The classification depends on whether the encoder or decoder plays the major role or both are in cooperation. We can summarize the three classes as follows:

1. Error concealment at the decoder: the decoder plays the major role by concealing the effect of transmission errors upon detection of errors.
2. Error-resilient encoding: the encoder plays the major role by making the bitstream more resilient to minimize the effect of possible transmission errors.
3. Encoder-decoder interactive error control: both encoder and decoder are involved by working cooperatively so that the encoder can adapt its operations according to the loss condition sent from the decoder.

2.3.4 Error concealment at the decoder

The error recovery techniques described in the section 2.3.3 aim at recovering a video signal without any degradation. Alternatively, error concealment techniques can be used at the decoder to conceal the effect of transmission errors in order to make the decoded frame much less visually objectionable. Note that only the decoder performs the task of error concealment without relying on any additional information from the encoder.

Error concealment is possible since images of natural scenes contain mainly low-frequency components. It means that the color values of spatially and temporally adjacent pixels vary smoothly except in areas containing shape edges and large motion activities. By considering this smoothness property of image and video signals, all the concealment techniques make use of the intact pixels nearby spatially and temporally to perform some kind of spatial/temporal estimation and interpolation of the lost information. Therefore, the decoder can conceal the

artifacts caused by transmission errors such that the affected regions are less visible to human perception. This concealment technique is easy to implement since the encoder, which is usually the most expensive part of the system, is kept unchanged.

The simplest concealment approach for recovering a corrupted MB can be achieved by copying the spatially corresponding MB in the previously decoded frame. This approach works exceptionally well in the region with temporal stationary, but it gives unsatisfactory results if the scene contains much motion. To tackle the problem, a more effective solution suggested by Civanlar [17] is to use the MB in the reference frame pointed by the motion vector of the corrupted MB. It is widely accepted that the use of the motion-compensated MB for concealment can improve the PSNR of reconstructed frames by 1dB at a cell-loss rate of 10^{-2} for MPEG-2 coded video [18]. Due to its simplicity, this technique is adopted in the MPEG-2 standard that allows the encoder to send the motion vectors for intra-MBs. Loss of such MBs can be recovered by the motion-compensated MBs in the previous frame. However, the recovery performance by this approach is critically dependant on the availability of the motion vector. It assumes that the motion vector of the corrupted MB is available in all circumstances. If the motion vector is also missing, it must first be estimated from the motion vectors of its surrounding MBs. When a MB is damaged, its horizontal adjacent MBs are likely to be damaged since all macrblocks in the same row are put into the same packet. Hence, the average or median of the motion vectors is taken over the motion vectors above and below for reconstructing the missing motion vector [19-20]. This is based on the spatial smoothness property of motion vectors in natural video. Lam et al [21]

proposed to estimate the lost motion vector by using the boundary matching algorithm. The estimated motion vector is chosen to be the one with minimum variation among neighboring intact pixels. In [2], the authors proposed to estimate the lost motion vector by using the motion vectors that next to the vertices of the lost MB. A first-order plane is then instituted and it indicates the movement tendency in that small area. Then, the lost motion vector is calculated for each pixel by using interpolation.

Another approach is to interpolate pixels in a corrupted MB or block from pixels in adjacent decoded MBs or blocks in the same frame. Note that a packet loss typically causes the loss of all MBs in the same row. In this case, only available adjacent blocks (or macroblocks) for a corrupted block (or macroblock) are those above or below. Usually, only the boundary pixels in adjacent blocks (or macroblocks) are used for interpolation.

In [22], the author suggested an algorithm to recover the corrupted block by adaptively performing interpolation in both the spatial, temporal, and frequency-domains. The weighting factors used for different interpolation depend on the motion content and loss patterns of the damaged regions. However, only the pixels near the intact blocks can be concealed effectively using this method. The pixels in the middle of the corrupted block especially at the area with edges are often blurred. Zhu and Wang [23] later developed an algorithm to reduce the blurring artifacts by using a second-order smoothness criterion. *W. Kwok et al* [24] further improved the concealment quality by using an edge-adaptive smoothness measure. The proposed algorithm predicts the edge direction of the corrupted blocks and can preserve the smoothness of edge information of the

video content so that the recovered MB looks more natural. Conversely, uncorrected prediction can lead to apparent artifacts in the concealed regions. Tsai and Lee [25] proposed another adaptive algorithm which recovers the corrupted block in a P-frame by selecting one of the most suitable predefined techniques. The adaptive selection is based on the coding mode of the B-frame next to the erroneous P-frame. In the MPEG standard, it defines four coding modes of the B-frame: direct mode, interpolated mode, forward mode and backward mode. According to the coding mode in the collocated MB in the B-frame, the motion activity of the corrupted MB can be roughly predicted. Hence, the algorithm will select the most suitable predefined technique for concealment.

2.3.5 Error-resilient coding

As compared to the error concealment approach, only encoder plays the role in error-resilient coding. To combat transmission errors, the error-resilient coding approach inserts redundancy to make the video bitstream more robust. The design objective in error-resilient coding is to achieve the best rate-distortion measure for a given amount of redundancy under an assumed channel with transmission errors.

One major reason for the sensitivity of a compressed video bitstream to transmission errors is due to the use of variable-length code words. Any single bit error cannot only make this code word undecodable, it can lead to the loss of synchronization. One effective approach to stop spatial error propagation is to isolate the effect of a transmission error within a limited region. Both MPEG-4

and H.263 standards [4,10,26] add one unique resynchronization code word in the compressed video bitstream periodically. If errors exist in the bitstream, the decoder can resume proper decoding when it detects a resynchronization code word. Although synchronization of the decoder can be regained, the spatial or temporal location is already lost. This problem can be solved by adding a distinct resynchronization code word at a fixed interval of the bitstream [27-28]. This approach not only can recover the synchronization sooner, but also regain the correct location of the following coefficients. Sadka et al [29] adopted a technique that uses fixed-length coding for some critical header information so as to avoid the loss of such important information. In the MPEG standard, some side information such as spatial and temporal locations is attached after the resynchronization code word in order to locate where the decoded MB belong to. Obviously, insertion of resynchronization code words will reduce coding efficiency. The more frequent such resynchronization code words are, the more bits that will be used for them. However, it would also enable the decoder to regain synchronization more quickly and it implies a smaller region in the reconstructed frame will be affected.

In addition to the insertion of resynchronization code words to the compressed bitstream, some algorithms further add extra information along with the main bitstream as side information. One approach is intentionally keeping some redundancy in the stage of source coding such that better error concealment can be carried out at the decoder in the presence of transmission errors. For example, [30] suggested sending a weighted sum of block coefficients for assisting the interpolation process of error concealment at the decoder. In [31-32], the authors suggested finding two sets of motion vectors for each MB, each

of which refers to different previous frame. Since the probability that these two reference frames simultaneously experience loss is low, it is more likely to have successful decoding of the current frame. In the absence of errors, one of the motion vectors is useless. The disadvantage of sending extra information to assist error concealment, however, is the bitrate boost of the bitstream, that may not be allowed for some applications or networks. Data hiding is an alternative to carry side information without substantially increasing bit rate. This technique is accomplished by modifying the imperceptible digital data [33]. Nevertheless, hiding extra information will degrade the video quality which depends on the amount of alternation occurring in the original data.

Another scheme for providing error resilience in a compressed video bitstream is layered coding combined with transport prioritization. Layered coding, which is a special case of scalable coding adopted since the MPEG-2 standard [2], purposely divides the video bitstream into different layers according to their importance. The purpose of scalable coding facilitates decoders with different bandwidth capacities or decoding powers to access the same video at different quality levels. To act as an error-resilient tool, the base layer carries the most important information such as coding modes, motion information and basic residue signal for the video bitstream while the enhancement layer contains the refinement signal of the video bitstream. The base layer can provide the basic quality of the video signal while higher quality video can be reconstructed with the enhancement layer. Note that the enhancement layer must be decoded with the based layer bitstream. To combat channel errors, layered coding must be worked together with unequal error protection in the network system. Different protection and priority are applied on different layers; obviously, the base layer

should be adopted with higher degree of protection and priority as compared with the enhancement layer. In case of channel error occurred, the most important information is most likely to be received. However, the coding efficiency of layered coding is not as good as that of the conventional video coding with only a single layer. The reason is that similar side information such as header information and coding modes must be transmitted in each layer. Besides, the enhancement layer must select only the frames of the base layer as reference frames for temporal prediction so as to avoid error propagation. In this case, the prediction gain will be significantly reduced and hence the coding efficiency will be dropped. Layered coding with unequal error protection has also been investigated extensively for various networks. For example, in the Grand Alliance high-definition television broadcast system, FEC has been adopted to protect the base layer against channel errors. Kansari et al [34] applied both FEC and retransmission for the base layer protection in their proposed resilient system.

Layered coding provides error resilience for transmitting the base layer with an error-free channel. However it may not be viable to guarantee lossless transmission of the base layer in some applications. Multiple Description Coding (MDC) is another error-resilient technique to combat transmission errors. In contrast to layered coding, which contains different importance of the layers, MDC assumes that there are several independent channels with equal error rates between the encoder and decoder. Each channel may be temporarily failed but the probability that all channels are down at the same time is small. MDC then separates the video bitstream into multiple equally important sub-streams or descriptions, and sends through different paths of the network. Each

sub-stream can be decoded independently to reconstruct a basic quality of the video signal at the decoder and that incremental improvement is achievable with additional descriptions. Since the probability of having all the sub-streams corrupted simultaneously is lower than that of a single stream, it can guarantee an acceptable quality with a single description. The disadvantage of this scheme is to reduce the coding efficiency as compared to the conventional encoder with only single description since each description in MDC must carry sufficient information about the original signal and there may be overlap in the information contained in different descriptions. Although the MDC scheme requires working on networks supporting multiple channels, it can be achieved through virtual channel connections by using time interleaving, frequency division, etc. The MDC scheme can be classified into four categories. They are temporal, spatial, scalar quantization and transform scalable coding. In temporal scalable coding with two descriptions, each of the sub-streams carries alternative frames of the video sequence. Frames in one sub-stream will only perform prediction from the reconstructed frame of itself. Thus, a low temporal quality of video can be reconstructed even though one description is corrupted. This approach introduces a quite large bit-rate overhead since the prediction interval between a current frame and its reference frame is farther apart temporally. Spatial scalable coding [35-36], on the other hand, reduces the prediction interval. It obtains multiple equally important descriptions by splitting adjacent samples among several channels using an interleaving subsampling lattice. Then, they code the subimages independently. With the new slice group provided in the H.264 standard, Wang et al [37] proposed a novel MDC approach. They introduce three independent MC loops, which allow full control of redundancy and side quality. A full video quality of reconstruction, compared with single

bitstream system, can be obtained when all the descriptions are correctly received. In [38], Vaishampayan suggested a multiple-description scalar quantization. In this approach, two sub-streams are created by using two separate quantization processes to produce two coarse quality video bitstreams. The quantization factors of these two sub-streams are designed so that if both sub-streams are received, the reconstruction accuracy is the same as that of using a fine quantizer.

2.3.6 Encoder-Decoder Interactive Error Control

In the techniques discussed so far, their performances in terms of both coding efficiency and error robustness are often not optimal due to the rapidly fluctuations of the network condition. In the error-free condition, side information and redundancy, which are targeted for providing error resilience, in the video bitstream is completely wasteful. On the contrary, the encoder cannot adaptively increase the robustness of error resilience for securing video delivery over networks with high error rate. These are all the consequences of the stand-alone encoder working without noticing the statistics of loss information from the decoder. Therefore, if the decoder can inform loss information or feedback information to the encoder, better performance can be achieved. The cooperative process of error control between the encoder and decoder can be realized if a backward channel from the decoder to the encoder is available. The feedback information is not in the scope of the video syntax but it is typically transmitted in a different layer of the protocol stack where control information is exchanged. For instance, the H.245 control protocol [39] is used to report the

temporal and spatial locations of corrupted MBs. More likely, such feedback information is transmitted in the error-free condition.

By using the feedback information from the decoder, the encoder is able to adopt its coding strategy to avoid error propagation. One simple approach is to synchronize the reconstructed reference frame in both of the encoder and decoder [40]. Temporal error propagation is caused by the reference mismatch between the encoder and decoder. When the reconstruction frame at the decoder is corrupted, the next predictive frame will also be degraded since the encoder assumes that the reference frame is decoded correctly. To solve this mismatch, a technique called error compensation has been proposed in [40]. In this technique, when the encoder knows the temporal and spatial locations of corrupted MBs through the feedback channel, it produces the same error-concealed MBs of the decoder. Note that this error compensation technique assumes that the same error concealment procedure as that used in the decoder is also carried out at the encoder. Thus, error propagation can be eliminated by coding the next frame according to the error-concealed frame, which is now synchronized in both encoder and decoder. Hueda [41] proposed to uses a specially designed frame called a correction frame to recover from erroneous reconstruction at the decoder. After the encoder estimates the degradation of the decoded frame according to the feedback information from the decoder, the correction frame is created by computing the difference between the corrected frame and the degraded frame. However, the extra bandwidth is required.

Another way of taking advantage of an available feedback channel is to employ the adaptive intra-refresh (AIR) scheme which has been widely adopted in both

H.263 [4] and MPEG-4 standards [10,26]. AIR employs intra-mode coding of MBs to stop temporal error propagation. Based on the feedback signal, the video encoder can locate the temporal and spatial occurrence of an error. It then evaluates the extent of propagated errors. The encoder then encodes the affected MBs to intra-MBs instead of inter-MBs. To do this, the relationship from the erroneous potions of the previous frame to the current frame is broken. The drawback of AIR will increase the bandwidth requirement. To reduce the bit-rate increase caused by intra-MBs, algorithms in [42-43] have been proposed to perform AIR selectively in which only severely affected MBs are refreshed in intra mode in order to mitigate the increase in bit rate. Redmill et al [44] suggested a similar approach to alleviate the bit budget boost problem by exploiting the active and static regions. The effect of active regions is more objectionable when video data is corrupted as compared with the static regions. Thus, only active regions containing transmission error will be recovered by using AIR. For the static region, the conventional concealment algorithm will be carried out since it often works very well in such area. In [45], the authors suggested combining the AIR and synchronization marker placement algorithms to enhance the error resilience of the video coding system. Two cost functions were employed to optimize the intra/inter-mode selection and the synchronization markers placement.

Instead of switching to intra-mode coding of MBs at the encoder, the feedback information also allows the encoder to select one of several previously decoded frames as a reference frame for motion estimation. In general, this approach is called reference picture selection (RPS) or Newpred [46-47], which has been adopted in the Annex N of the H.263 standard [4] and supported in the MPEG-4 v2 standard [10,26]. Unlike AIR, which will increase the data rate significantly,

RPS can change the reference frames of encoding incoming frames dynamically at the encoder side when it is acknowledged that previously received frame is corrupted at the decoder. In order to prevent error propagation, the encoder evaluates the extent of propagated errors. Then, it chooses a new reference frame which is certain to be arrived safely at the decoder. Compared to AIR, the reduction in coding efficiency due to the use of an older reference frame is remarkably lower if it is not too far. In RPS, the feedback message indicates which frame is correctly received and which is not. It also determines which frame is available in the decoder. This information is necessary to allow the encoder to select an appropriate reference frame for coding subsequent frames. Therefore, RPS requires extra frame buffers in both encoder and decoder. The size of frame buffers is directly related to error-resilient power. For the system having larger buffer size, encoder will have more freedom to select an uncorrupted previous frame as a reference and result in higher error-resilient power. Apart from the buffer size, Round Trip Delay (RTD) is another critical factor that affects the performance of RPS. RTD is the elapsed time required for a frame to be transmitted to the decoder and an acknowledgement to be sent back from the decoder. A longer RTD just results in a later start of error recovery.

In MPEG-4 [10,26], RPS can be operated in two different modes - NACK and ACK, as shown in Figure 2.7 and Figure 2.8, respectively. They are designed for catering to different network conditions. The NACK mode sends only a negative acknowledgement signal (NACK) whenever the decoder spots corruption in received frame. For this mode, the encoder always utilizes the previous frame as a reference unless a NACK is received, as depicted in Figure 2.7. When a NACK is received in the encoder, the nearest correctly reconstructed frame in

the past will be selected for the reference of the next frame. Therefore, the maximum length of error propagation is equal to RTD. Note that the operation of the encoder is not changed in error-free transmission. It implies that no performance loss will be incurred for error-free transmission. However, temporal error propagation still exists for the period of one RTD in the NACK mode. Therefore, the NACK mode is preferred in low-error rate channels. On the other hand, the ACK mode is suitable for high-error rate channels. All of the correctly reconstructed video frames at the decoder side will acknowledge the encoder by sending positive acknowledgements (ACK). In the encoder, only the frames that have been confirmed to be correctly received will be employed as references, as depicted in Figure 2.8. Therefore, temporal error propagation can be stopped within one frame, although the coding efficiency is low during error-free transmission. To be more flexible, a strategy has been also investigated to adaptively perform switching between two modes. If consecutive times of NACK messages received by the encoder reach to a preset value, the encoder can assume the error rate of the current network is awful. Then, the ACK mode will be used for carrying heavier protection to the video bitstream. On the contrary, when the encoder receives consecutive times of ACK messages which is larger than a predetermined number, the encoder will switch back to the NACK mode for obtaining the best coding efficiency in the good channel condition.

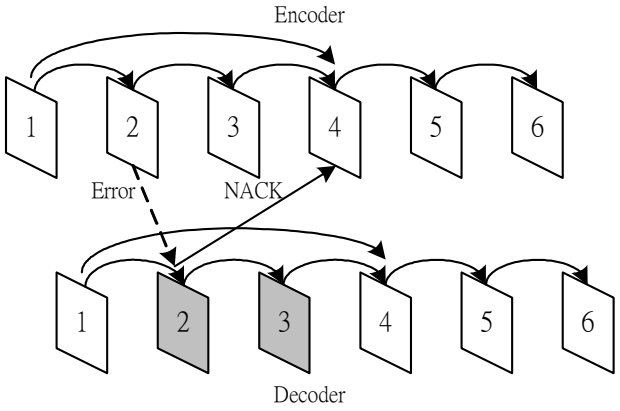


Figure 2.7 RPS using NACK mode.

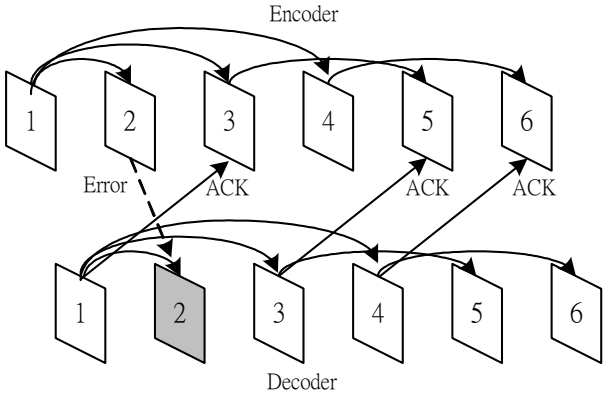


Figure 2.8 RPS using ACK mode.

In both ACK and NACK, the number of reference frames in the buffer can directly affect the ability of error resilience. Kimata et al [48] proposed an algorithm to maximize the usability of the limited buffer equipped in the decoder and encoder. Two new modes called ACK+mode and NACK+mode were introduced to free the un-usable or damaged information from the buffer and leave more room to correctly decoded pictures. Liang et al [49] also designed a method to optimize the usage of RPS according to the real-time channel condition. They use a rate-

distortion framework and a binary tree structure of error propagation estimation to calculate the optimal way of RPS. Lin et al [50] proposed a similar algorithm to optimize the assignment of the reference frame by introducing a cost function into the RPS scheme. In [51], the author mentioned an idea of combining MDC into RPS. The probability of having both of the channels in bad status simultaneously is lower than that of the single channel. Hence, this scheme is more robustness than the conventional RPS scheme. Since many interactive techniques for error recovery incur delay, any video packets that are received after the playout time is discarded and it will cause quality degradation. The algorithms proposed in [52-53] aimed at extending the arrival deadline of video packets even though they are received after the playout time. Late packets can recover the degraded previous frames in extra frame buffers. Eventually the proceeding frame is recovered from error and error propagation can be stopped. These algorithms provide an impressive result when they are cooperated with RPS.

2.4 Error Recovery Schemes for Already MPEG Encoded Bitstreams

Most of the existing interactive error recovery algorithms are targeted on real time video encoding. They are not so suitable for applications using an already MPEG encoded video bitstream. However, many recent video services and applications, such as video on demand (VoD), digital TV, and distance learning, will use pre-encoded video for storage and transmission rather than real-time encoded video. Such applications require an off-line encoder to generate a pre-encoded video bitstream. This approach is especially useful for production systems such as encoding a large video sequence into a storage device for

future delivery. In this off-line mode, the time-varying characteristics of the present communications networks pose particular difficulties for a pre-encoded video bistream to vary according to the channel conditions and produces acceptable quality. It is because the off-line encoder can only produce a single bitstream which does not adapt to the time-varying channel conditions at the time of transmission of the pre-encoded video bitstream. In other words, if a pre-encoded video bitstream is used, the lack of flexibility makes it difficult to change the resilience of the bitstream.

The problem of error resilience addressed above is very challenging. So far, only a limited number of error-resilient transcoding schemes [54-55] have been proposed to tackle the error-resilient problem when the pre-encoded video bitstream is transmitting over the channel in which part of the already transmitted bitstream has been corrupted. These transcoding schemes provide the necessary protection for transcoded video streams prior to their transmission. The authors in [54] suggested a rate-distortion framework with analytical models for error-resilient transcoding. The models are used to characterize how corruption propagates temporally and spatial in MPEG-encoded video subject to bit errors, and they optimize the combination of spatial error resilience, temporal error resilience, and transmission bitrate. In [55], an error-resilient transcoding scheme was proposed for a general packet radio services (GPRS) mobile access network. The transcoding process utilizes two error-resilience coding schemes – the adaptive intra refresh (AIR) [4,10,26] and reference picture selection (RPS) [46-47] schemes with feedback control signaling (FCS). The schemes can work together or independently. In the video server, it makes use of the feedback signal from the decoder which indicates which parts of the

decoded video frame were corrupted and had to be recovered. Both AIR and/or RPS are then used to add the necessary robustness to transcoded video data. For instance, AIR adopts intra-MBs to stop temporal error propagation. Based on the feedback signal, the video server can locate the temporal and spatial occurrence of an error. It then evaluates the extent of propagated errors. The error-resilient transcoder converts the affected inter-MBs to intra-MBs. To do this, the pre-encoded video bitstream is fully decoded and it includes the processes of motion compensation, dequantization, inverse transformation, and variable length decoding. The coefficients of those intra-MBs to be transmitted are then transformed, requantized, scanned and variable-length encoded to create the new intra-MBs. On the other hand, RPS adopted in the error-resilient transcoder can also stop temporal error propagation by allowing the transcoder to select one of several previously decoded frames as a reference for motion compensation. In this approach, a feedback sent from the decoder to the video server signifies that a given frame has been damaged by transmission errors. To stop temporal error propagation, the error-resilient transcoder will detect which frame uses the corrupted frame as a reference frame. Then, the transcoder will re-encode the affected frame with a different and uncorrupted reference frame. The approach used in [55] is to decode all the P-frames from the previously nearest I-frame to the affected frame which is then re-encoded with a new reference; this can create undesirable complexity in the video server as well as introduce re-encoding errors. In this thesis, we will propose efficient techniques on error-resilient video transcoding scheme for RPS to reduce the computational requirement of the video server and the quality degradation of the reconstructed video arising from re-encoding.

2.5 Chapter Summary

To resolve the difficulties of transmitting coded video data through error prone networks, many recently proposed error recovery schemes have been reviewed. We started this chapter by reviewing the basic principle of video compression, so as to understand the underlying aspect of error propagation (spatially and temporally) in the presence of transmission errors. The main reasons of spatial error propagation are due to the adoption of VLC and the predictive techniques in coding motion vectors and DC coefficients, while the temporal error propagation is due to the use of motion-compensated prediction. In general, some conventional data correction techniques can be applied directly to the compressed video bitstream. However, these techniques will introduce long delay. Therefore, we described various error recovery schemes that are designed to combat transmission errors in real-time video communications. These schemes can be classified into three categories by whether the encoder or decoder plays the major role or both are involved in cooperation. Schemes in each category contain different features and are suitable for different scenarios. All the error-concealment techniques employed at the decoder can recover the corrupted MBs by making use of the temporal and spatial smoothness property of the image and video signals. They are appropriate in any circumstances. However, the effectiveness of such techniques is limited by the available information. Error-resilient coding, on the other hand, achieves error resilience by adding a certain amount of redundancy in the coded bitstreams at the source coder. The techniques in this category are suitable for video broadcasting applications. They usually have better recovery power due to the extra information provided from the encoder. However, they consume extra bandwidth

for providing robust transmission. We also reviewed several interactive techniques. In this category, the decoder can inform the encoder which part of the coded information is lost by using a backward channel. The techniques in this category should give the best performance since the redundancy is added only when an error occurs. However, in applications such as broadcast, where no backward channel exists, none of the interactive techniques can be applied. Besides, all the existing interactive error recovery techniques are targeted on real time video encoding. Nowadays, many recent video services and applications, such as video on demand (VoD), digital TV, and distance learning, will use pre-encoded video for storage and transmission rather than real-time encoded video. So far, only a limited number of techniques have been proposed to tackle this problem. We did some reviews of these schemes at the end of this chapter. Results of our investigation indicate that these methods are still primitive, and there is plenty of room for improvement. Therefore, in the following chapters, we examine the use of compressed-domain processing to modify an already encoded bitstream for the purpose of improving its resilience prior to transmission over a noisy wireless channel.

Chapter 3

The Straightforward Error Resilience System for Already MPEG Encoded Bitstreams

3.1 Introduction

In Chapter 2, we have reviewed various error recovery schemes in the literature. All these schemes are used to protect digital video transmission over lossy networks. One of the most common schemes is reference picture selection (RPS), which has also been adopted in many video coding standards. Most of the earlier work on RPS has been studied for use in real-time encoding, but has not been examined in transmitting an already encoded bitstream over error-prone networks, which has emerged as one of the indispensable video services over the Internet and 3G wireless networks nowadays.

One straightforward approach to implement RPS in an already MPEG encoded bitstream is to decode all the P-frames from the previously nearest I-frame to the current transmitted frame which is then re-encoded with a new reference. However, this approach will induce a high complexity of the server and it is not desirable. In this chapter, we will present the impact of directly transplanting RPS in an already MPEG encoded bitstream on server complexity. Later, we will show that it is nearly impossible to directly adopt RPS in this application. Finally, simulation results for some testing video sequences will be given to confirm our discussion.

3.2 Straightforward approach for already MPEG encoded bitstreams

In RPS, temporal error propagation due to transmission errors in P-frames can be terminated by encoding the next frame adaptively with one of several previously decoded frames which has been correctly decoded. Both encoder and decoder sides are equipped with additional frame buffers to store several previous frames, because of round trip delay (RTD). Without RPS, only one frame (two frames are needed in case of B-frames) has to be stored. Generally, the error recovery power of RPS relies on the number of additional frame buffers at the encoder and decoder as well as RTD of the feedback signal. For instance, to stop error propagation, the possible range of previous frames that can be chosen for a reference is directly depended on the available frame buffers. It means that the larger the buffer size they can be used for storing the reconstructed frames, the more flexible the incoming frame can be selected for its reference, and as a result, the higher error-resilient power the RPS scheme has. Figure 3.1 shows an example to illustrate what happens to RPS for the situation with long RTD and shortage of frame buffers in the encoder and decoder. Assume that there are four frame buffers for both of the encoder and decoder. In the situation shown in Figure 3.1, due to traffic congestion or system failure, the encoder receives the delayed NACK signal of frame *2* right after frame *5* is delivered. At this moment, the four frame buffers equipped in the encoder and decoder contain the reconstructed frame *2* to frame *5*. The reconstructed frame *1* has already been dropped due to the first-in-first-out strategy of the buffer architecture. Then, the encoder estimates the extent of error propagation in order to determine the best reference for coding the next

frame. However, RPS fails in this situation since all the reconstructed frames in the decoder buffers are corrupted. Note that errors in frame *3* to frame *5* are due to error propagation from frame *2*. Hence, one possible approach to solve this problem is to increase the number of additional frame buffers in both of the encoder and decoder.

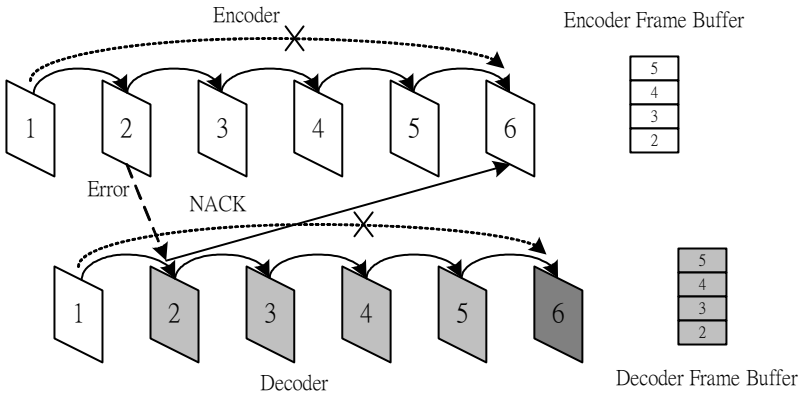


Figure 3.1 Failure of RPS operation due to adequate frame buffers.

However, in the circumstance of transmitting an already MPEG encoded bitstream, frame reconstruction is not necessary at the encoder since video data in the server has already been encoded. In error-free transmission, no frame buffer is necessary to be equipped in the encoder side, as shown in Figure 3.2. It means that pixel values of each frame are not directly available in the encoder. However, when RPS is adopted for combating transmission errors, the corresponding pixel values are required. Therefore, a special buffer arrangement is required for the encoder. Figure 3.3(a) shows the possible structure of adopting pixel-domain RPS in an already MPEG encoded bitstream. The switch SW_1 is used to enable the RPS operation in order to reflect the necessary action required when a frame is lost during error-prone transmission. The server or proxy is designed to receive an acknowledgement from the decoder. This acknowledgement signifies that a given frame has been corrupted

by transmission errors. In this case, SW_1 is connected to B_1 to activate the RPS operation. The operation requires to decode all the P-frames from the previously nearest I-frame to the affected frame that uses the corrupted frame as a reference. The affected frame is then re-encoded with a new reference. Figure 3.3(b) illustrates an example describing the RPS operation in this scenario. Since B-frames are not used as references for later frames, for the sake of simplicity, we focus our discussion on the case that the video bitstream contains I- and P-frames only. In this example, the server learns that a transmission error has occurred in frame *n-1* through a feedback channel, it requires modifying the reference frame of frame *n*. One straightforward approach is to re-encode frame *n* not relative to frame *n-1*, but to an older reference frame, frame *n-2*, which is known to be the last frame available without errors in the decoder. In other words, the server needs to decode frame *1* to frame *n* and perform re-encoding of frame *n* with frame *n-2* as the reference. In Figure 3.3(a), the frame selector is used to extract the side information such as all necessary headers, coding mode, zero run-length and encoded DCT coefficients for each MB of the necessary frames from the pre-encoded bitstream. A metadata file recording the location of each frame in the pre-encoded bitstream is generated off-line so that the server/proxy can locate the particular frame easily. At the moment of decoding frame *0*, switches SW_2, SW_3 and SW_4 are open such that the decoded frame *0* is stored in the buffer, FB_1, to reconstruct the next frame. During the decoding of frame *1* to frame *n-2*, SW_2 is close, and SW_3 and SW_4 remain open. When frame *n-2* is reconstructed, SW_4 is close and then the decoded version of frame *n-2* is copied from FB_1 to FB_2. SW_4 is open again during the decoding of frame *n-1* and frame *n*. At that moment, the reconstructed frame *n* and frame *n-2* are

stored in FB_1 and FB_2 respectively. The sever then re-encodes frame *n* with the new reference, frame *n-2*, by setting SW_3 to close. Figure 3.4 summarizes the switching positions during re-encoding frame *n*. Thus, straightforward implementation requires much higher server or proxy complexity for these decoding and re-encoding processes. Besides, the video quality suffers from its re-encoding process, which introduces additional degradation. This is due to the reason that both the degraded version of frame *n* and *n-2* are used for motion estimation and residual computation in order to generate the new prediction error which is then undergone the lossy re-quantization process before transmitting to the decoder. Since frame *n* is used as the reference frame for the subsequent transmitted P-frames, quality degradation propagates to later frames and it means that the reconstructed quality of the video sequence is degraded significantly when RPS is operated in the already encoded video bitstream.

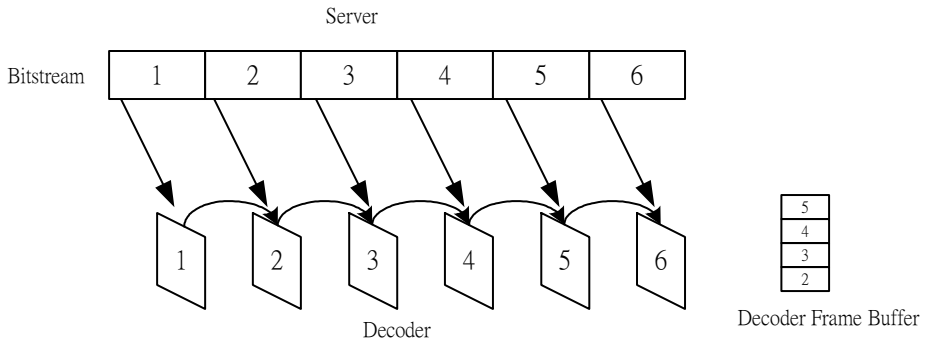


Figure 3.2 Server-client model for transmitting an MPEG bitstream.

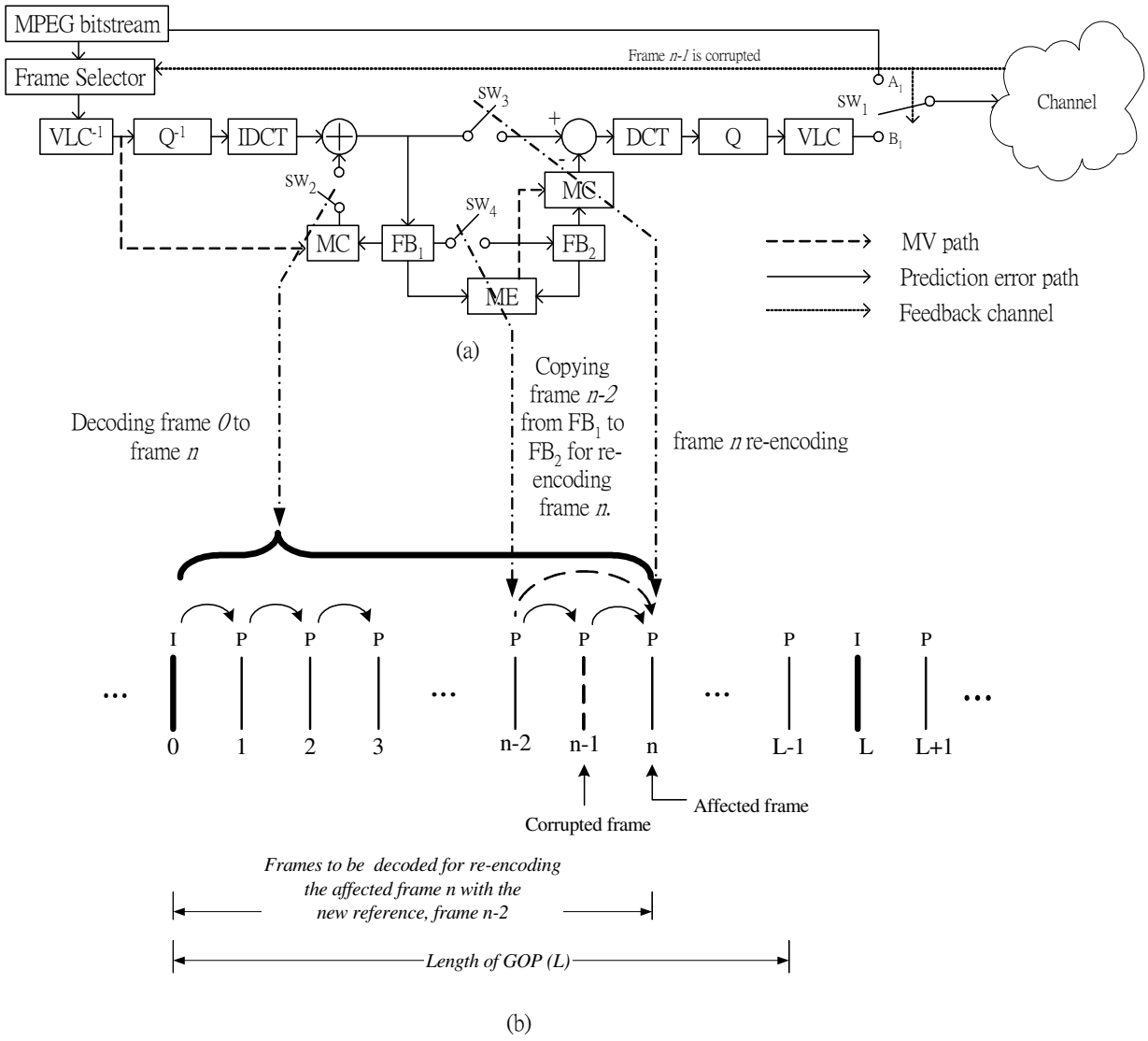


Figure 3.3 Architecture of adopting pixel-domain RPS in the already MPEG encoded bitstream.

SW_4	open	open	close	open	open
SW_3	open	open	open	open	close
SW_2	open	close	close	close	open
SW_1	B_1	B_1	B_1	B_1	B_1
Frame No.	*0*	*1 to n-2*	After frame *n-2* is reconstructed	*n-1,n*	Re-encoding Frame *n*

Figure 3.4 Switch positions during re-encoding process.

3.3 Complexity measurement

In order to measure the computational requirements of the server or proxy associated with the straightforward implementation of RPS in the already MPEG encoded bitstream, we introduce a cost that can directly reflect the computational burden of the server/proxy when the RPS technique is adopted. It is reasonable to approximate the cost to the required number of MBs to be decoded and re-encoded in the server/proxy. In the following, the meaning of the proxy is interchangeable with that of the server since the proxy is always implemented in the server. For the RPS scheme using the re-encoding technique mentioned in Section 3.2, the server requires decoding all MBs in P-frames from the previously nearest I-frame to the current transmitted frame, and re-encode all the MBs in the current frame with a new reference. The major steps in decoding a single MB are two-dimensional inverse DCT (2D-IDCT), dequantization and motion compensation operations while the re-encoding process involves two-dimensional DCT (2D-DCT), quantization, motion estimation and compensation operations. Note that forward 2D-DCT requires nearly the same computational operations as inverse 2D-DCT in the decoding process. Therefore, the main difference in computational complexity between the decoding and re-encoding processes is the motion estimation operation. It actually depends on the motion estimation algorithm to be used for the re-encoding process. In general, measuring the difference between the costs of these two processes would not be an easy task. In spite of these, it is possible to assume their complexities are the same for the sake of simplicity. Taking the above consideration, we can approximate the computational requirements of the straightforward RPS

implementation in the already MPEG encoded bitstream by using the total number of MBs to be decoded and re-encoded, and it is given by

$$\text{Complexity} = H \times V \times (n+1) / 16 \times 16 \qquad (3.1)$$

where H x V is the size of the frame and n is the number of decoded frames.

The computational requirement of this straightforward implementation is depicted in Figure 3.5 where the test video stream used for simulation is any sequence with a CIF format. The sequence is encoded at a frame-rate of 30 fps and the GOP length is *15* with an I-P structure. The complexity of every frame is plotted when the straightforward RPS is carried out at the particular frame. This figure shows that the complexity grows almost linearly as the corrupted frame is further apart from the previous I-frame, and it leads to a significant increase in the server complexity. On average, the operation of RPS is required at the middle of the GOP. From Figure 3.5, it still induces heavy computational burden to the server. Currently, there is no similar approach that is specially designed for applying RPS in this scenario, we will take this conventional approach as the reference for our proposed algorithms in this thesis.

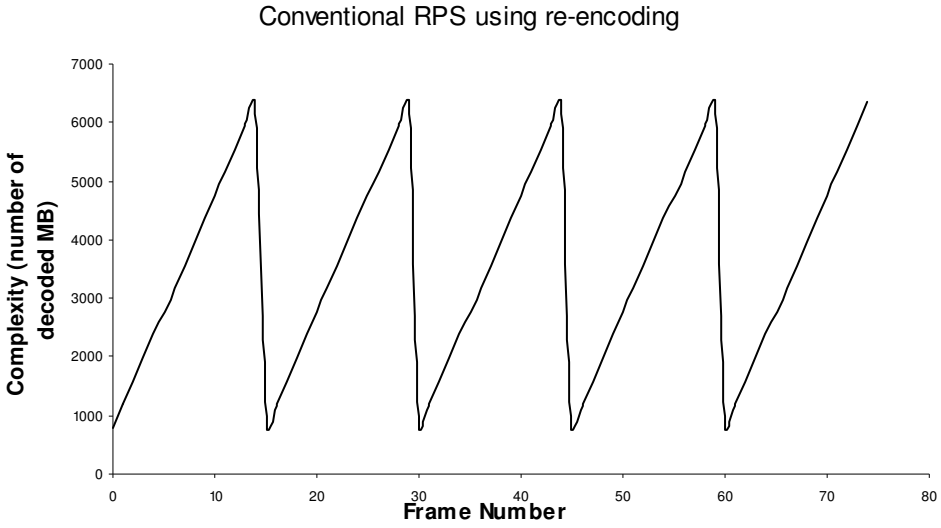


Figure 3.5 Server complexity for CIF video sequences (GOP length = 15)

3.4 Chapter summary

The RPS scheme is an efficient error-resilient approach for real time encoding scenario. However, no existing RPS algorithm, which is specially designed for using RPS in transmitting an already MPEG encoded bitstream, has been found in the literature. In order to carry out RPS for combating the channel errors in this circumstance, a straightforward approach is suggested in this chapter. Due to the adoption of predictive coding in the MPEG bitstream, performing RPS in the already MPEG encoded bitstream is inherently difficult and impractical. The necessary operations of the decoding process along with the exhaustive re-encoding process lead to heavy computational burden to the server and quality degradation. Eventually, visible delay and artifact may result in the decoder. One example has been taken for illustrating the complex processes in the real situation. In order to measure the computational complexity of this

straightforward implementation, we have also introduced a simple method which uses the number of decoded MBs as an approximation of the server complexity.

Chapter 4

The Proposed Techniques for Macroblocks without Motion Compensation

4.1 Introduction

In the last chapter, we have shown the impact of adopting RPS in an already MPEG encoded bitstream on the server complexity. The straightforward implementation of RPS in this application requires much higher server complexity for decoding and re-encoding processes. Besides, the video quality suffers from its re-encoding process, which introduces additional degradation. Since the newly reconstructed frame will be used as a reference frame for the following transmitted P-frames, quality degradation propagates to later frames and it means that the reconstructed quality of the video sequence is degraded significantly. In this chapter, we propose two possible MB-based solutions to allow the server to select one of several previously and correctly decoded frames of the decoder as a reference frame for re-encoding. They are designed to support RPS in an already encoded video bitstream.

4.2 The Proposed MB Based Algorithm

For our new MB-based techniques, two types of MB are now defined. For illustration, let us use the example in Figure 3.3(b) again. Assume that the server receives an acknowledgment signifying frame *n-1* has been damaged by transmission errors. The server needs to re-encode frame *n* with the new

reference, frame *n-2*. The situation in MB level is depicted in Figure 4.1. We assume that $MB^n_{(k,l)}$ represents the MB at the k^{th} row and l^{th} column of frame *n*. $MB^n_{(k,l)}$ is defined as a non-motion compensation (non-MC) MB if the MB is coded without motion compensation. Otherwise, it is defined as a motion-compensated (MC) MB. For example, in Figure 4.1, since the motion vector of $MB^n_{(0,1)}$, $mv^n_{(0,1)}$, is zero, it means that $MB^n_{(0,1)}$ is a non-MC MB. On the other hand, $MB^n_{(1,1)}$ is categorized as a MC MB. In this thesis, our proposed scheme works at the MB-level. Since the server will process non-MC MBs in the compressed domain, a complete decoding and re-encoding process is not required for these non-MC MBs such that the increase in computational burden of the server is greatly reduced with the minimal re-encoding errors.

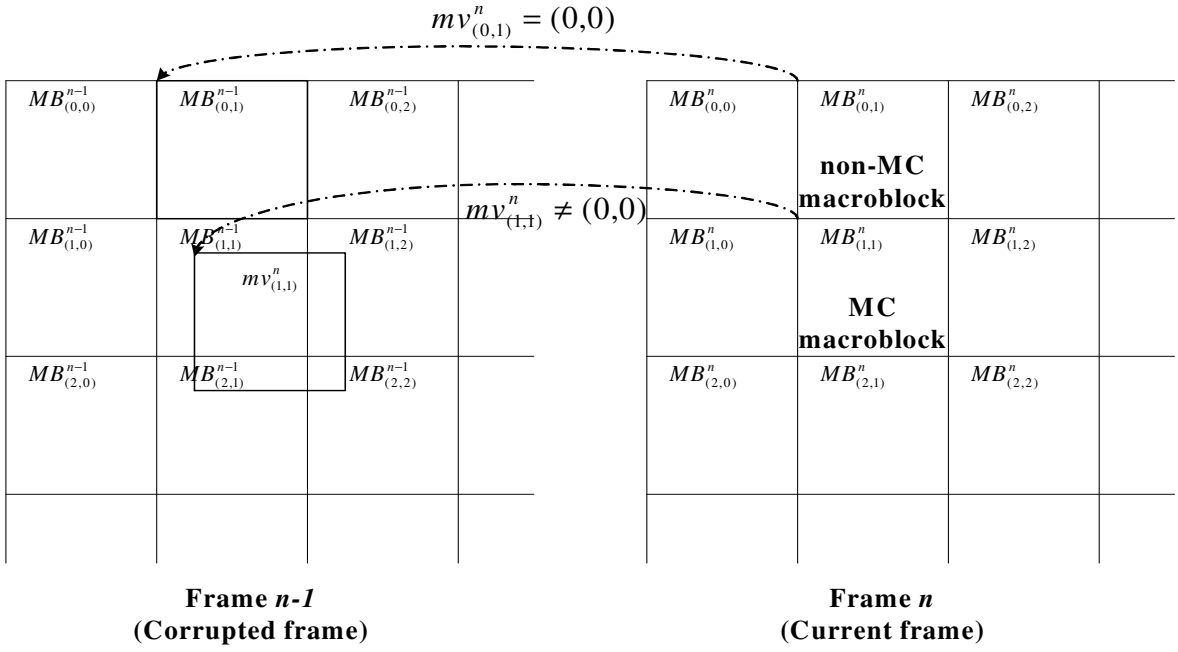


Figure 4.1. RPS in a non-MC MB.

4.2.1 Direct addition of quantized DCT coefficients for non-MC MBs

Block motion-compensated prediction (MCP) is the type of prediction used in various MPEG standards [4-5,26]. As mentioned in Chapter 2, this prediction type is what gives the MPEG codecs the advantage over pure still-frame coding methods. In MCP, the previously transmitted and decoded frame serves as the prediction for the current frame. The difference between the prediction and the actual current frame is the prediction error. The coded prediction error is added to the prediction to obtain the final representation of the current reconstructed frame. For decoding, each MB in frame *n*, $MB^n_{(k,l)}$, is reconstructed according to MCP and it is given by

$$MB^n_{(k,l)} = MC^{n-1}(mv^n_{(k,l)}) + e^n_{(k,l)} \tag{4.1}$$

where $MC^{n-1}(mv^n_{(k,l)})$ stands for the motion-compensated MB of $MB^n_{(k,l)}$ which is translated by the motion vector, $mv^n_{(k,l)}$, in the previously reconstructed frame *n-1* and $e^n_{(k,l)}$ is the prediction error between $MB^n_{(k,l)}$ and its motion-compensated MB, $MC^{n-1}(mv^n_{(k,l)})$.

In Figure 4.2, a situation in which frame *n-1* is corrupted is illustrated. In order to stop error propagation by operating RPS in the already encoded video bitstream, frame *n-2*, which is the last frame available at the decoder without errors, should be selected as the reference frame for frame *n*. The decoder then reconstructs each $MB^n_{(k,l)}$ according to frame *n-2* which is given by

$$MB^n_{(k,l)} = MC^{n-2}(\hat{mv}^{n}_{(k,l)}) + \hat{e}^n_{(k,l)} \tag{4.2}$$

where $\hat{mv}^{n}_{(k,l)}$ and $\hat{e}^{n}_{(k,l)}$ are the new motion vector and prediction error of $MB^{n}_{(k,l)}$ by using frame *n-2* as the reference, respectively. For the server, it needs to compute $\hat{mv}^{n}_{(k,l)}$ and then encode $\hat{e}^{n}_{(k,l)}$ in the quantized DCT domain. To calculate $\hat{e}^{n}_{(k,l)}$, all the related previous MBs in P-/I-frames need to be decoded by the server in the conventional system, as mentioned in Chapter 3. It becomes impractical when the GOP size is large.

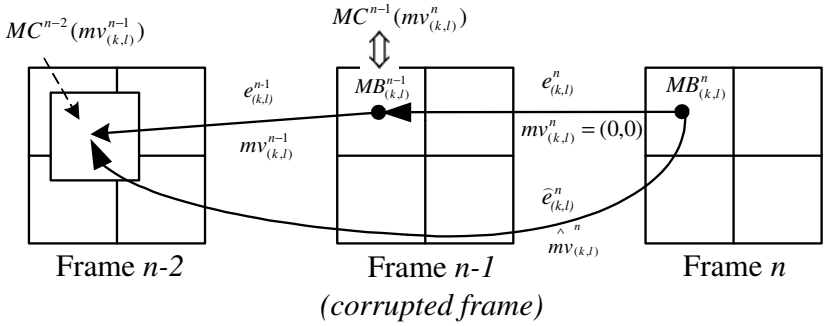


Figure 4.2. RPS in a non-MC MB.

However, as shown in Figure 4.2, if the current MB $MB^{n}_{(k,l)}$ is found to be a non-MC MB, then $\hat{mv}^{n}_{(k,l)}$ can be computed by adding $mv^{n}_{(k,l)}$ and $mv^{n-1}_{(k,l)}$ together, as indicated below,

$$\hat{mv}^{n}_{(k,l)} = mv^{n}_{(k,l)} + mv^{n-1}_{(k,l)} \tag{4.3}$$

Besides, since the motion vector $mv^{n}_{(k,l)}$ is zero, $\hat{mv}^{n}_{(k,l)}$ can be further simplified as

$$\hat{mv}^{n}_{(k,l)} = mv^{n-1}_{(k,l)} \tag{4.4}$$

Putting (4.4) into (4.2), we have

$$\hat{e}^{n}_{(k,l)} = MB^{n}_{(k,l)} - MC^{n-2}(mv^{n-1}_{(k,l)}) \tag{4.5}$$

For $MB^{n}_{(k,l)}$, the spatial position of its motion-compensated MB, $MC^{n-1}(mv^{n}_{(k,l)})$, in frame *n-1* is the same as that of $MB^{n-1}_{(k,l)}$, as depicted in Figure 4.2. Hence, for this specific case, $MC^{n-1}(mv^{n}_{(k,l)})$ is equal to $MB^{n-1}_{(k,l)}$, and (4.1) can be rewritten as

$$MB^{n}_{(k,l)} = MB^{n-1}_{(k,l)} + e^{n}_{(k,l)} \tag{4.6}$$

In the original MPEG bitstream, the reconstructed MB $MB^{n-1}_{(k,l)}$ is given by

$$MB^{n-1}_{(k,l)} = MC^{n-2}(mv^{n-1}_{(k,l)}) + e^{n-1}_{(k,l)} \tag{4.7}$$

After inserting (4.7) in (4.6), the prediction error between $MB^{n}_{(k,l)}$ and $MC^{n-2}(mv^{n-1}_{(k,l)})$ in (4.5), $\hat{e}^{n}_{(k,l)}$, can be written as

$$\begin{aligned} \hat{e}^{n}_{(k,l)} &= MB^{n}_{(k,l)} - MC^{n-2}(mv^{n-1}_{(k,l)}) \\ &= e^{n}_{(k,l)} + e^{n-1}_{(k,l)} \end{aligned} \tag{4.8}$$

Note that both $\hat{e}^{n}_{(k,l)}$ and $\hat{e}^{n-1}_{(k,l)}$ are pixel-domain values. Only the quantized DCT coefficients of $e^{n}_{(k,l)}$ and $e^{n-1}_{(k,l)}$ are available in the already MPEG encoded bitstream. Let them be $E^{n}_{(k,l)}$ and $E^{n-1}_{(k,l)}$, and they can be written as

$$e^{n}_{(k.l)} = DCT^{-1}\left(Q_L^{-1}\left(E^{n}_{(k,l)}\right)\right) \tag{4.9}$$

and

$$e^{n-1}_{(k.l)} = DCT^{-1}\left(Q_L^{-1}\left(E^{n-1}_{(k,l)}\right)\right) \tag{4.10}$$

where $DCT^{-1}(\ )$ is an inverse DCT operator and $Q_L^{-1}(\ )$ is an inverse quantization operator with the quantization step size *L*. Substituting (4.9) and (4.10) into (4.8), we obtain

$$\hat{e}^{n}_{(k.l)} = DCT^{-1}\left(Q_L^{-1}\left(E^{n}_{(k,l)}\right)\right) + DCT^{-1}\left(Q_L^{-1}\left(E^{n-1}_{(k,l)}\right)\right) \tag{4.11}$$

From (4.11), $\hat{e}^{n}_{(k,l)}$ can be computed by adding the decoded version of $E^{n}_{(k,l)}$ and $E^{n-1}_{(k,l)}$. To stop the temporal error propagation in the decoder, the encoded form

of $\hat{e}^n_{(k,l)}$, $Q_L\left(DCT\left(\hat{e}^n_{(k,l)}\right)\right)$, is transmitted from the server. To do so, $\hat{e}^n_{(k,l)}$ needs to undergo transformation and quantization. Although the server only needs to decode quantized DCT coefficients of two MBs for each non-MC MB with the new reference instead of decoding all its related previous MBs in P-/I-frames, the additional quantization process can lead to requantization errors, but this can be avoided if $Q_L\left(DCT\left(\hat{e}^n_{(k,l)}\right)\right)$ is computed in the quantized DCT domain, which will be discussed in the followings.

By applying DCT to (4.11) and taking into account the linearity of DCT, (4.11) can be written as

$$DCT\left(\hat{e}^n_{(k,l)}\right) = Q_L^{-1}\left(E^n_{(k,l)}\right) + Q_L^{-1}\left(E^{n-1}_{(k,l)}\right) \tag{4.12}$$

Note that dequantization is not a linear operation because of the existence of the dead zone. However, it is reasonable to approximate $Q_L^{-1}\left(E^n_{(k,l)} + E^{n-1}_{(k,l)}\right) \approx Q_L^{-1}\left(E^n_{(k,l)}\right) + Q_L^{-1}\left(E^{n-1}_{(k,l)}\right)$ to avoid the process of requantization. Thus, we obtain the final expression of $Q_L\left(DCT\left(\hat{e}^n_{(k,l)}\right)\right)$, as shown below,

$$Q_L\left(DCT\left(\hat{e}^n_{(k,l)}\right)\right) \approx E^n_{(k,l)} + E^{n-1}_{(k,l)} \tag{4.13}$$

Equation (4.13) implies that the newly quantized DCT coefficients $Q_L\left(DCT\left(\hat{e}^n_{(k,l)}\right)\right)$ can be computed in the quantized DCT domain by adding $E^n_{(k,l)}$ and $E^{n-1}_{(k,l)}$. Both of them can be extracted from the MPEG video bitstream in the server by performing entropy decoding only. The server combines $E^n_{(k,l)}$ and $E^{n-1}_{(k,l)}$ together which is then entropy encoded. Since no complete decoding process is

necessary during the generation of $Q_L\left(DCT\left(\widetilde{e}^{n}_{(k,l)}\right)\right)$, the computational complexity required for the server is very limited.

For a real world image sequence, the block motion field is usually gentle, smooth, and varying slowly. As a consequence, the distribution of motion vectors is center-biased [2,18,21], as demonstrated by the typical examples as shown in Table 4.1 which shows the distribution of the coding modes for various sequences including “Mother and Daughter”, “Salesman”, “Football”, “Canlendar”, “Table Tennis”, “Foreman” and “Carphone”. These sequences have been selected to emphasize different amount of motion activities. It is clear that over 90% and 34% of the MBs are coded without motion compensation for sequences containing low and high amount of motion activities respectively. By using a direct addition of the quantized DCT coefficients, the computational complexity and re-encoding errors of sequences containing more non-MC MBs can be reduced significantly.

Table 4.1. Percentage of the non-MC MB for various sequences.

Mother and daughter	Salesman	Football	Calendar	Table Tennis	Foreman	Carphone
90.23	81.79	45.51	43.19	41.72	36.93	34.12

4.2.2 Modified Quantizer-dequantizer Pair for Non-MC MBs

The purpose of the direct addition technique is to show the potential benefits of using RPS in the already MPEG encoded bitstream. From (4.13), when the effect of making a linear approximation of inverse quantization is negligible, performing a direct addition of quantized DCT coefficients will not give any

significant loss in reconstructing a video frame with a new reference. However, since in general there is no guarantee that the effect is negligible all the time, there are non-zero probabilities that the approximation may cause some loss in the reconstructed frame. In this section, we will discuss the effect of the linear approximation of inverse quantization when the aforementioned technique is applied to RPS for the already MPEG encoded bitstream.

From the derivations of the previous section, it follows that the newly quantized DCT coefficients, $Q_L\left(DCT\left(\widehat{e}^{n}_{(k,l)}\right)\right)$, should be applied to the decoder. The dequantization process specified in the MPEG-4 standard is performed as shown in the following equation:

$$Q_L^{-1}(x) = \begin{cases} 0 & if\ x = 0 \\ \left(x + \dfrac{sign(x)}{2}\right) \times L & otherwise \end{cases} \tag{4.14}$$

where

$$L = \frac{2 \times quantizer\_scale \times Q_m}{16}$$

and,

$$sign(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases}$$

L is the quantization step size which depends on the quantizer scale and the value of the weighting matrix (Q_m). The higher the value, the coarser the quantization. Thus, a different quantization step size can be used for each transform coefficient, if appropriate. The term $\frac{sign(x)}{2}$ is required to deliver the centroid representation of the dequantizer by introducing an addition or

subtraction of half of the quantization step size, which depends on the polarity of the quantized DCT coefficient.

If the linear approximation of inverse quantization is used, we obtain the output of the dequantizer by substituting $E^{n}_{(k,l)} + E^{n-1}_{(k,l)}$ into (4.14), and it yields

$$Q_L^{-1}\left(E^{n}_{(k,l)} + E^{n-1}_{(k,l)}\right) = \left(E^{n}_{(k,l)} + E^{n-1}_{(k,l)}\right) \times L + sign\left(E^{n}_{(k,l)} + E^{n-1}_{(k,l)}\right) \times \frac{L}{2} \quad (4.15)$$

On the other hand, the desired output of the dequantizer without linear approximation is $Q_L^{-1}\left(E^{n}_{(k,l)}\right) + Q_L^{-1}\left(E^{n-1}_{(k,l)}\right)$ which can be actually written as

$$Q_L^{-1}\left(E^{n}_{(k,l)}\right) + Q_L^{-1}\left(E^{n-1}_{(k,l)}\right) = \left(E^{n}_{(k,l)} + E^{n-1}_{(k,l)}\right) \times L + \left[sign\left(E^{n}_{(k,l)}\right) + sign\left(E^{n-1}_{(k,l)}\right)\right] \times \frac{L}{2} \quad (4.16)$$

Comparing (4.15) and (4.16), they are not equal since $sign(E^{n}_{(k,l)} + E^{n-1}_{(k,l)}) \neq sign(E^{n}_{(k,l)}) + sign(E^{n-1}_{(k,l)})$ when $E^{n}_{(k,l)}$ and $E^{n-1}_{(k,l)}$ are in opposite sign with different magnitude. In this case, $sign(E^{n}_{(k,l)}) + sign(E^{n-1}_{(k,l)})$ is equal to zero whereas $sign(E^{n}_{(k,l)} + E^{n-1}_{(k,l)})$ is either equal to 1 or –1. Besides, the possible outputs of $sign(E^{n}_{(k,l)} + E^{n-1}_{(k,l)})$ are {-1, 0, +1} while the possible outcomes of $sign(E^{n}_{(k,l)}) + sign(E^{n-1}_{(k,l)})$ are {-2, -1, 0, +1, +2}.

This implies that the reconstructed quality of using the RPS scheme in the already MPEG encoded bitstream with the linear approximation of inverse quantization deviates from the desired output. The effect of approximation errors is depicted in Table 4.2, which shows the PSNR performances of non-MC MBs after the corrupted frame when the direct addition (DA) technique and the conventional re-encoding technique are applied to RPS. This table indicates that

the RPS scheme using DA outperforms the conventional one. However, the approximation errors of inverse quantization in the proposed DA lead to a drop in the picture quality as compared to the desired output. Details on the simulation environment including computational savings and coding parameters used in the simulation are given in Chapter 6.

Table 4.2. PSNR performances after the corrupted frame for the direct addition technique and the conventional RPS using the re-encoding technique (only non-MC MBs).

Video Sequence	Desired output	Conventional RPS	RPS scheme using DA
Mother and Daughter	45.41	45.18	45.37
Salesman	43.76	43.21	43.41
Football	43.54	42.09	42.50
Calendar	44.39	42.02	43.20
Table Tennis	43.35	42.12	42.94
Foreman	43.98	42.37	43.66
Carphone	44.41	43.33	44.12

In order to avoid the approximation errors, we need to design a better way to quantize $Q_L^{-1}\left(E_{(k,l)}^{n}\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)$ and then transmit it to the decoder. If the optimum performance is desired, the two steps on quantization and dequantization have to be designed jointly. From (4.16), $Q_L^{-1}\left(E_{(k,l)}^{n}\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)$ is composed of two terms, $\left(E_{(k,l)}^{n}+E_{(k,l)}^{n-1}\right)\times L$ and $\left[sign\left(E_{(k,l)}^{n}\right)+sign\left(E_{(k,l)}^{n-1}\right)\right]\times\frac{L}{2}$. The former one is divisible by *L*. However, possible outputs of the latter one are $\{-L,-\frac{L}{2},0,+\frac{L}{2},+L\}$, some of which are not divisible by *L*. In certain cases, such as when $sign\left(E_{(k,l)}^{n}\right)+sign\left(E_{(k,l)}^{n-1}\right)$ is equal to –1 or 1, the requantization with a

quantization step size of *L* can lead to addition errors. One possible way to avoid the requantization errors is to use a step size of *L/2* instead, since both of the terms in (4.16) are divisible by *L/2*. Besides, in MPEG-4, the dead zone is adopted in the quantization process. The dead zone commonly refers to the central region of the quantizer, whereby the coefficients are quantized to zero and it is intended primarily to make more non-significant coefficients to become zero resulting in an increase of the coding efficiency. Figure 4.3(a) shows the decision levels and their corresponding representation levels of the quantizer with a quantization step size of *L/2* and a dead zone. Since the dead zone exists, the representation levels of this quantizer become $\{...,-\frac{5L}{4},-\frac{3L}{4},0,+\frac{3L}{4},+\frac{5L}{4},...\}$ which is not the multiple of *L/2*. That is, it does not match the desired representation levels of $Q_L^{-1}\left(E_{(k,l)}^n\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)$. In Figure 4.3(b), it shows the decision levels and their corresponding representation levels of the quantizer without a dead zone, again, the quantization step size is equal to *L/2*. In this case, it exactly matches the possible outcome of $Q_L^{-1}\left(E_{(k,l)}^n\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)$.

Taking all of the above considerations, the new quantized DCT coefficients with the new reference is obtained by

$$\begin{aligned}\tilde{Q}_{L/2}\left(DCT\left(\hat{e}_{(k.l)}^n\right)\right) &= \tilde{Q}_{L/2}\left(Q_L^{-1}\left(E_{(k,l)}^n\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)\right)\\ &= 2\left(E_{(k,l)}^n+E_{(k,l)}^{n-1}\right)+\left(sign(E_{(k,l)}^n)+sign(E_{(k,l)}^{n-1}\right)\end{aligned} \tag{4.17}$$

where $\tilde{Q}_{L/2}(\ )$ is the quantization operator without a dead zone and its quantization step size is equal to *L/2*. Similar to the direct addition technique of quantized DCT coefficients, (4.17) signifies that the newly quantized DCT coefficients $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}_{(k.l)}^n\right)\right)$ can be generated in the quantized DCT domain

since both $E^{n}_{(k,l)}$ and $E^{n-1}_{(k,l)}$ can be extracted from the already encoded MPEG video bitstream in the server by performing entropy decoding. The server doubles the sum of $E^{n}_{(k,l)}$ and $E^{n-1}_{(k,l)}$ and then adds it to $\left(sign(E^{n}_{(k,l)}) + sign(E^{n-1}_{(k,l)})\right)$ which is then entropy encoded. Again, requantization is not necessary to be carried out in the formation of $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}^{n}_{(k.l)}\right)\right)$.

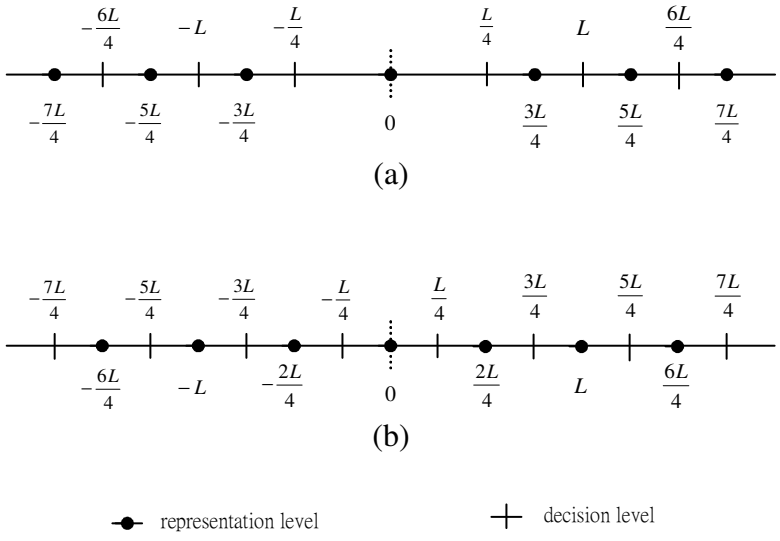


Figure 4.3. Decision levels (vertical line) and representation levels (dot) of the quantizer (a) with a dead zone and (b) without a dead zone. Both of them have a quantization step size of *L/2*.

Since $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}^{n}_{(k.l)}\right)\right)$ is generated from the quantizer without a dead zone, the term $\frac{sign(x)}{2}\times L$ is not necessary in the dequantization process and it can be modified as shown in (4.18) during decoding non-MC MBs with the new reference.

$$\tilde{Q}^{-1}_{L/2}(x) = \begin{cases} 0 & if\ \mathrm{x} = 0 \\ \dfrac{x \times L}{2} & otherwise \end{cases} \tag{4.18}$$

By using the modified quantizer-dequantizer pair without a dead zone, the receiver can perfectly reconstruct the desired output of MBs with a new reference frame. For the proposed technique, only a small change is needed for the receiver to equip with the dequantization process as shown in (4.18). It is exactly the MPEG dequantization equation for decoding intra-blocks with half of the quantization factor.

Note that, for MC MBs, the above two techniques cannot be employed since $MB_{(k,l)}^{n-1}$ is not on a MB boundary. In other words, $E_{(k,l)}^{n-1}$ is not available from the bitstream. To reconstruct MC MBs of frame *n* directly, the server will examine the motion vectors in the MPEG bitstream and all the related MBs from the previously nearest I-frame to frame *n-1* should be decoded. Then, the server performs re-encoding for those MC MBs with frame *n-2* as the reference.

The derivation in (4.17) assumes that the round-trip delay (RTD) corresponds to the duration of encoding time for one frame. For a duration longer than one frame, the concept can be further extended if the motion vectors of MBs that have same spatial location in the skipped frames are all equal to zero. From (4.17), it can be easily shown that

$$\tilde{Q}_{L/2}\left(DCT\left(\hat{e}_{(k.l)}^{n,RTD\_dist}\right)\right) = \sum_{i=0}^{RTD\_dist}\left[2E_{(k,l)}^{n-i} + sign(E_{(k,l)}^{n-i})\right] \tag{4.19}$$

where $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}_{(k.l)}^{n,RTD\_dist}\right)\right)$ is the newly quantized DCT coefficients of $MB_{(k,l)}^{n}$ by using frame *n-RTD_dist* as the new reference frame. Note that the index *RTD_dist* indicates the number of frames corresponding to the duration of encoding time at one particular RTD. Figure 4.4 shows an example when the

RTD is equivalent to the encoding time of two frames, that is, *RTD_dist* is equal to 2. Assume that frame *n* is the current frame and the server has been informed by the client that frame *n-2* is corrupted. Since the RTD is equivalent to the duration of encoding time of two frames, the server then needs to generate the quantized DCT coefficients of frame *n* with the new reference, frame *n-3*. Figure 4.4 also shows the situation in MB level in which the motion vectors of $MB^{n}_{(0,1)}$ and $MB^{n-1}_{(0,1)}$ are both equal to zero. The newly quantized DCT coefficients, $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}^{n,2}_{(k,l)}\right)\right)$, can be computed according to (4.19) which is again operated in the quantized DCT domain, thus eliminating requantization errors and the unnecessary operations for the re-encoding process.

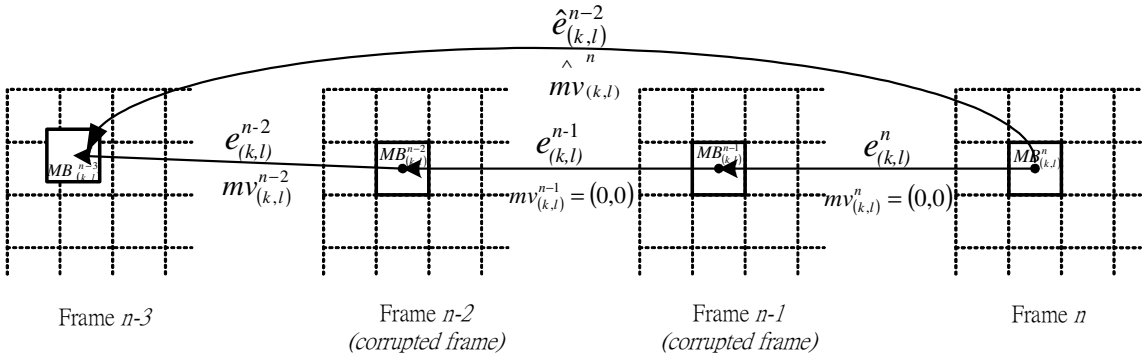


Figure 4.4. The proposed MB-based techniques when the round-trip delay (RTD) corresponds to the duration of encoding time for two frames.

During the transmission of $\tilde{Q}_{L/2}\left(DCT\left(\hat{e}^{n}_{(k,l)}\right)\right)$, MBs that use the modified dequantizer without a dead zone have to be notified by the decoder. We note that this notification has to be transmitted as a side information in the MPEG-4 video bitstream. However it is not specified in MPEG-4, and must therefore be additionally signaled by using user data; see [6] for the definition of user data.

4.3 Chapter Summary

In this chapter, we have addressed various issues in implementing the RPS scheme in an already MPEG encoded bitstream. We showed in the last chapter that the straightforward approach might result in much higher complexity of the server and introduce re-encoding errors; it is not desirable. A new idea for the compressed-domain techniques applied to RPS, which can be adopted in an already MPEG encoded bitstream, has been established in this chapter. The proposed techniques are 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 – MBs without motion compensation (non-MC MBs) and motion-compensated MBs (MC MBs). Then it selects the necessary MBs adaptively, processes them in the compressed domain, and sends the processed MBs to the receiver. For non-MC MBs, we have proposed two techniques. The first one is a direct addition of quantized DCT coefficients to deactivate most of the complex modules of adopting the RPS scheme in the already MPEG encoding bitstream. To avoid the non-linear problem of the direct addition technique, a modified quantizer-dequantizer pair for non-MC MBs has then been proposed to maintain the reconstructed quality. Since the proposed techniques are manipulated on the quantized DCT domain, it is not necessary to perform decoding and re-encoding of the video streams in the server. Thus it will not complicate the implementation of the server. Furthermore, since the re-encoding process can be prevented, the visual quality of non-MC MBs with a new reference will be nearly the same as that of the original sequence in the server. Simulation results

in chapter 6 will show that our proposed techniques can minimize (i) server complexity significantly and (ii) the quality degradation.

Chapter 5

The Proposed DCT-domain Operators for Motion-compensated Macroblocks

5.1 Introduction

The key to high performances in the techniques proposed in Chapter 4 lies in the center-biased motion vector distribution of real-world video sequences. Therefore, both techniques manipulating non-MC MBs could be quite helpful in the video sequences containing gentle, smooth and slow motion. When the amount of motion activity in a video sequence increases, the performances of the proposed techniques are dropped since it contains more MC MBs in which the direct addition and modified quantization-dequantization techniques cannot be employed. To further enhance our proposed scheme, in this chapter, we investigate various DCT-domain operators for handling MC MBs when RPS is applied in an already MPEG encoded bitstream.

5.2 Motion Vector Re-composition for MC MBs

In MC MBs, the techniques previously described for non-MC MBs cannot be employed directly. The underlying reason is depicted in Figure 5.1 in which there is no single motion vector existed in the video bitstream describing the motion of the whole 16x16 block, $MC^{n-1}\left(mv^{n}_{(k,l)}\right)$, in frame *n-1*. In fact, $MC^{n-1}\left(mv^{n}_{(k,l)}\right)$ in frame *n-1* overlaps with four MBs containing independent motion vectors. Hence, both of the motion vector and the corresponding prediction error of the current

MB, $MB^{n}_{(k,l)}$, require to be re-computed with the reference pointing to the last uncorrupted frame *n-2*. One simple way of getting the motion vector of $MB^{n}_{(k,l)}$ with the new reference is to perform full-scale full search motion estimation. This approach requires the pixel intensities of both $MB^{n}_{(k,l)}$ and the corresponding search area in the new reference frame (frame *n-2*). The server needs to decode all the related MBs to $MB^{n}_{(k,l)}$ and the corresponding search area in frame *n-2* starting from the previously nearest I-frame. Re-encoding between $MB^{n}_{(k,l)}$ and its best-matched MB in frame *n-2* is then required.

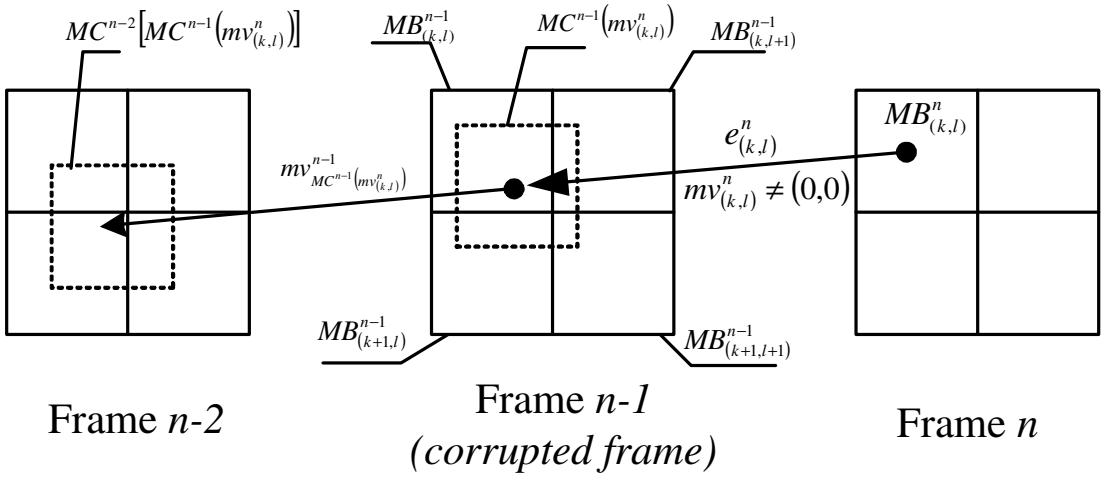


Figure 5.1. Motion vector re-composition in a MC MB.

Although the optimized motion vector can be achieved by this full-scale full search motion estimation, it is not desirable due to its high computational complexity. It has been a common practice to reuse incoming motion vectors in various video transcoders [56-57]. In our development, we borrow the idea from motion re-estimation of video transcoders to speed up our RPS scheme for pre-encoded video bitstreams.

In Figure 5.1, we assume that $MC^{n-1}\left(mv^{n}_{(k.l)}\right)$ is the best-matched MB to $MB^{n}_{(k,l)}$, and $MC^{n-2}\left[MC^{n-1}\left(mv^{n}_{(k.l)}\right)\right]$ is the best-matched MB to $MC^{n-1}\left(mv^{n}_{(k.l)}\right)$. Since frame *n-1* is damaged, for $MB^{n}_{(k,l)}$, it is necessary to find a motion vector pointing to a MB in frame *n-2*. One obvious way to get such a motion vector without carrying out full-scale full search motion estimation is to add $mv^{n}_{(k.l)}$ and $mv^{n-1}_{MC^{n-1}\left(mv^{n}_{(k.l)}\right)}$ together, as shown in Figure 5.1. However, this computation is not practical since $MC^{n-1}\left(mv^{n}_{(k.l)}\right)$ is not on a MB boundary. In other words, $mv^{n-1}_{MC^{n-1}\left(mv^{n}_{(k.l)}\right)}$ is not available from the pre-encoded video bitstream. In order to make an approximation of $mv^{n-1}_{MC^{n-1}\left(mv^{n}_{(k.l)}\right)}$, it is possible to use the bilinear interpolation from the motion vectors of the four overlapping MBs with $MC^{n-1}\left(mv^{n}_{(k.l)}\right)$ in frame *n-1*. However, the bilinear interpolation of motion vectors can lead to the inaccuracy of the resultant motion vector when the motion vectors of the four overlapping MBs are too divergent and too large to be described by a single motion vector, as illustrated in Figure 5.2.

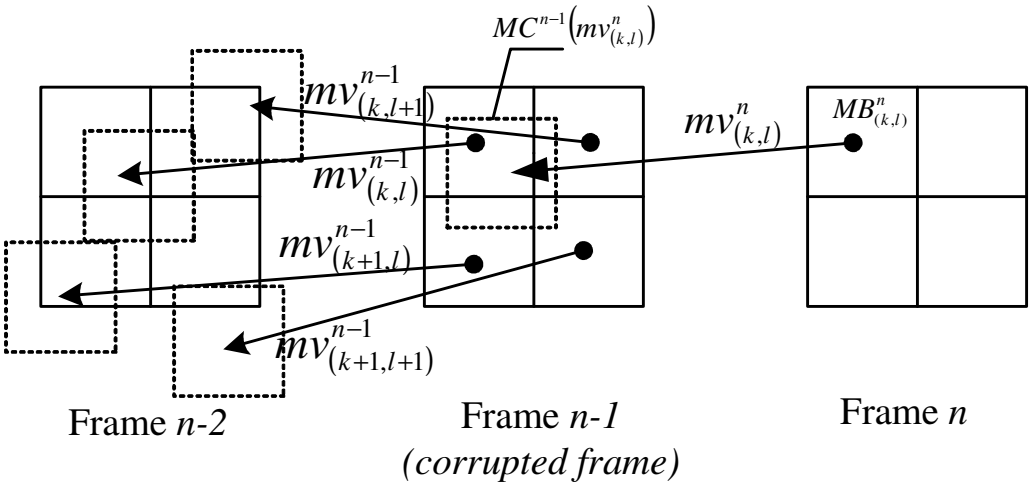


Figure 5.2. Motion vectors of the four overlapping MBs with $MB^{n}_{(k,l)}$.

Instead of using the bilinear interpolation of the available motion vectors, forward dominant vector selection (FDVS), which has much better performance than the bilinear approach, is adopted in our proposed scheme. As shown in Figure 5.3, the FDVS method is to select one dominant motion vector from the four overlapping MBs in frame *n-1*. A dominant motion vector is defined as the motion vector carried by a dominant MB. The dominant MB is the MB that has the largest overlapping segment with $MC^{n-1}\left(mv^{n}_{(k,l)}\right)$ pointed by $mv^{n}_{(k,l)}$. For the example in Figure 5.3, $MB^{n-1}_{(k,l)}$ is chosen as the dominant MB since it has the largest overlapping area with $MC^{n-1}\left(mv^{n}_{(k,l)}\right)$, while its motion vector $mv^{n-1}_{(k,l)}$ is selected as the dominant motion vector. Therefore, the resultant motion vector of $MB^{n}_{(k,l)}$, $\hat{mv}^{n}_{(k,l)}$, pointing to the MB in frame *n-2* is the sum of the dominant motion vector $mv^{n-1}_{(k,l)}$ and $mv^{n}_{(k,l)}$, which can be written as

$$\hat{mv}^{n}_{(k,l)} = mv^{n-1}_{(k,l)} + mv^{n}_{(k,l)} \tag{5.1}$$

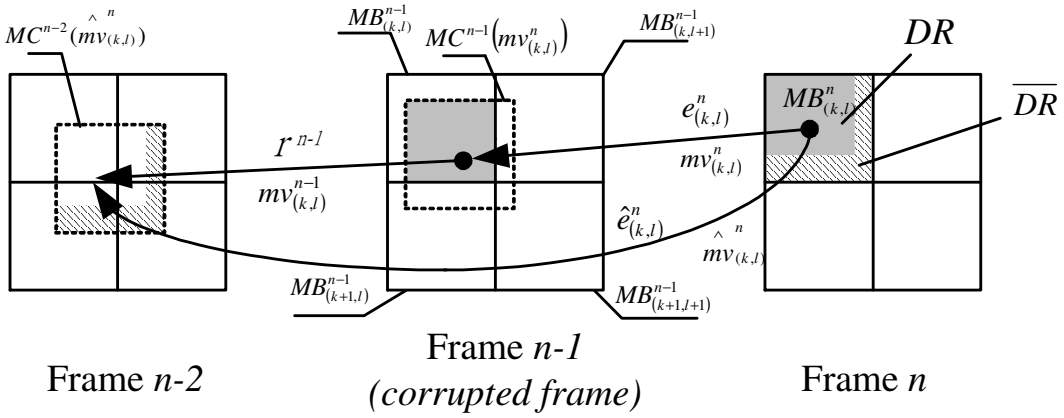


Figure 5.3. FDVS in a MC MB.

5.3 New Prediction Errors in MC MBs

After re-composing $\hat{mv}^{n}_{(k,l)}$, the next step is to compute the new prediction error, $\hat{e}^{n}_{(k,l)}$. With simple reordering, we can rewrite Equation (4.2) as

$$\hat{e}^{n}_{(k,l)} = MB^{n}_{(k,l)} - MC^{n-2}(\hat{mv}^{n}_{(k,l)}) \tag{5.2}$$

In Figure 5.3, pixels in $MB^{n}_{(k,l)}$ can be described as

$$MB^{n}_{(k,l)} = MC^{n-1}\left(mv^{n}_{(k,l)}\right) + e^{n}_{(k,l)} \tag{5.3}$$

Putting (5.3) into (5.2), we obtain

$$\hat{e}^{n}_{(k,l)} = e^{n}_{(k,l)} + r^{n-1} \tag{5.4}$$

where r^{n-1} represents the prediction error between $MC^{n-1}(mv^{n}_{(k,l)})$ and $MC^{n-2}(\hat{mv}^{n}_{(k,l)})$, and it can be written as

$$r^{n-1} = MC^{n-1}(mv^{n}_{(k,l)}) - MC^{n-2}(\hat{mv}^{n}_{(k,l)}) \tag{5.5}$$

In the MPEG-4 standard, DCT is applied to an *8×8* block and each MB is composed of four *8×8* blocks. Therefore, we can re-write (5.4) at block level and it is given by

$$\hat{e}^{n}_{(k,l),i} = e^{n}_{(k,l),i} + r^{n-1}_{i} \tag{5.6}$$

where *i= 1,2,3* and *4,* and it represents the coding order (in raster scan) of blocks within a MB. For the sake of convenience, we use the same convention for other symbols for the rest of this chapter; i.e. y_i represents one of four *8×8* blocks in the MB named *y* and the index *i* indicates the coding order of blocks within a MB, as

depicted in Figure 5.4. Therefore, r_1^{n-1}, r_2^{n-1}, r_3^{n-1} and r_4^{n-1} in (5.6) are the prediction errors of the four *8×8* blocks in r^{n-1}.

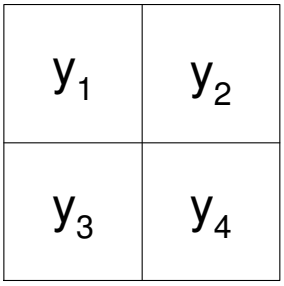


Figure 5.4. Block order.

Using the fact that

$$DCT(A+B) = DCT(A) + DCT(B) \tag{5.7}$$

and applying DCT to (5.6), we have

$$DCT(\hat{e}^n_{(k,l),i}) = DCT(e^n_{(k,l),i}) + DCT(r_i^{n-1}) \tag{5.8}$$

This equation indicates that the newly transformed prediction error of each 8×8 block $DCT\left(\hat{e}^n_{(k,l),i}\right)$ may be calculated in the DCT domain by adding $DCT\left(e^n_{(k,l),i}\right)$ and $DCT\left(r_i^{n-1}\right)$ if they can be extracted from the original encoded bitstream. If this is the case, the server does not need to decode all the related MBs to $MB^n_{(k,l)}$ from the previously nearest I-frame and it can avoid the unnecessary decoding process. In the actual situation shown in Figure 5.3, $DCT(r_i^{n-1})$ is not on a MB boundary. It means that $DCT(r_i^{n-1})$ is not directly available in the already MPEG encoded bitstream since $MC^{n-1}(mv^n_{(k,l)})$ is generally formed by using parts of four segments which come from its four neighboring MBs - $MB^{n-1}_{(k,l)}$, $MB^{n-1}_{(k,l+1)}$, $MB^{n-1}_{(k+1,l)}$ and $MB^{n-1}_{(k+1,l+1)}$.

Our focus here is to investigate whether it is possible to reuse the DCT coefficients of these four neighboring MBs to come up with $DCT(r_i^{n-1})$. Considering that FDVS is employed, the dominant motion vector $mv_{(k,l)}^{n-1}$ of $MB_{(k,l)}^{n-1}$ is used to re-compose the new motion vector $\hat{mv}_{(k,l)}^{n}$, as the example shown in Figure 5.3. But other MBs may have different motion vectors. If one of these motion vectors in the original MPEG bitstream is different from the dominant motion vector, it is impossible to re-use the DCT coefficients in that MB. With the help of FDVS, we can compute $DCT(\hat{e}_{(k,l),i}^{n})$ as much as possible in the DCT form in order to keep the benefits of the DCT-domain manipulation. To do so, we need to consider two different regions of each MC MB separately – the dominant region (DR) and the non-dominant region ($\overline{\mathrm{DR}}$). Figure 5.3 gives a clear account of our idea for defining the DR and $\overline{\mathrm{DR}}$, which are represented by the area filled with shaded color and diagonal lines respectively. The DR includes pixels in $MB_{(k,l)}^{n}$ that points to the dominant MB in the previously corrupted frame while the remaining pixels constitute the $\overline{\mathrm{DR}}$. Therefore, $\hat{e}_{(k,l)}^{n}$ can be divided into two terms

$$\hat{e}_{(k,l)}^{n} = \hat{e}_{(k,l)}^{n,\mathrm{DR}} + \hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}} \tag{5.9}$$

where $\hat{e}_{(k,l)}^{n,\mathrm{DR}}$ and $\hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}}$ are the prediction errors in the DR and $\overline{\mathrm{DR}}$ respectively.

By taking into account of the linearity of DCT in (5.7) and applying DCT to (5.9), it can be written as

$$DCT(\hat{e}^{n}_{(k,l)}) = DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l)}) + DCT(\hat{e}^{n,\overline{\mathrm{DR}}}_{(k,l)}) \tag{5.10}$$

In the following, we will discuss how to contrive two different DCT-domain operators for computing $DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l)})$ in the DCT domain. By doing so, we can further expedite the proposed RPS scheme mentioned in Chapter 4.

5.4 DCT-domain Operators for DRs

To compute $DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l)})$ in the DCT domain, the new DCT-domain operators should be designed in block level to compute $DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l),i})$, where *i=1, 2, 3 and 4*. The derivation of (5.8) can be done on $DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l),i})$ again. Therefore

$$DCT(\hat{e}^{n,\mathrm{DR}}_{(k,l),i}) = DCT(e^{n,\mathrm{DR}}_{(k,l),i}) + DCT(r_i^{n-1,\mathrm{DR}}) \tag{5.11}$$

where $e^{n,\mathrm{DR}}_{(k,l),i}$ is the prediction error in the DR of $MB^{n}_{(k,l)}$ and $r_i^{n-1,\mathrm{DR}}$ is the prediction error overlapping with the dominant MB in frame *n-1*. Note that (5.11) is no longer valid for $\overline{\mathrm{DR}}$. It is because $MC^{n-1}(mv^{n}_{(k,l)})$ in frame *n-1* is not on a MB boundary, and hence, area outside the DR may have different motion vectors as shown in Figure 5.2. Our objectives here are to acquire $DCT(r_i^{n-1,\mathrm{DR}})$ and $DCT(e^{n,\mathrm{DR}}_{(k,l),i})$ in the DCT domain from frame *n-1* and frame *n* respectively. Although both of them cannot be retrieved directly from the already encoded bitstream, they can be obtained by reusing the existing prediction errors stored in the bitstream through some DCT-domain operators.

5.4.1 Shifting Operator

Figure 5.5 illustrates an example that shows the detailed block structure of $MC^{n-1}(mv^{n}_{(k,l)})$ in frame *n-1*. r_1^{n-1}, r_2^{n-1}, r_3^{n-1} and r_4^{n-1} are the prediction errors of the corresponding four *8×8* blocks in $MC^{n-1}(mv^{n}_{(k,l)})$ and their regions overlapping with the dominant MB in frame *n-1* are denoted by $r_1^{n-1,DR}$, $r_2^{n-1,DR}$, $r_3^{n-1,DR}$ and $r_4^{n-1,DR}$, (the shaded region in Figure 5.5). Here, $e^{n-1}_{(k,l),1}$, $e^{n-1}_{(k,l),2}$, $e^{n-1}_{(k,l),3}$ and $e^{n-1}_{(k,l),4}$ are the prediction errors in $MB^{n-1}_{(k,l)}$. Their DCT coefficients ($DCT\left(e^{n-1}_{(k,l),1}\right)$, $DCT\left(e^{n-1}_{(k,l),2}\right)$, $DCT\left(e^{n-1}_{(k,l),3}\right)$, and $DCT\left(e^{n-1}_{(k,l),4}\right)$) are available in the original encoded bitstream after performing inverse quantization. These are also the four overlapping blocks with $r_1^{n-1,DR}$, $r_2^{n-1,DR}$, $r_3^{n-1,DR}$ and $r_4^{n-1,DR}$ from which they are contributed to derive $DCT\left(r_i^{n-1,DR}\right)$ as discussed below.

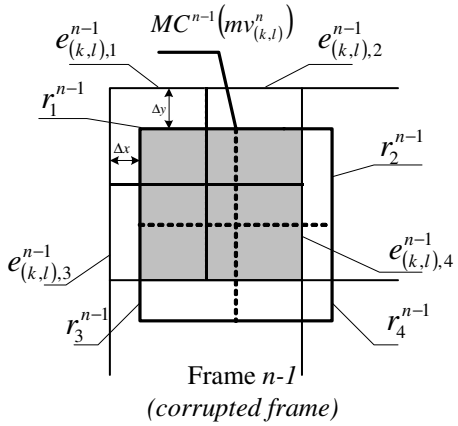


Figure 5.5. Block level in frame *n-1*.

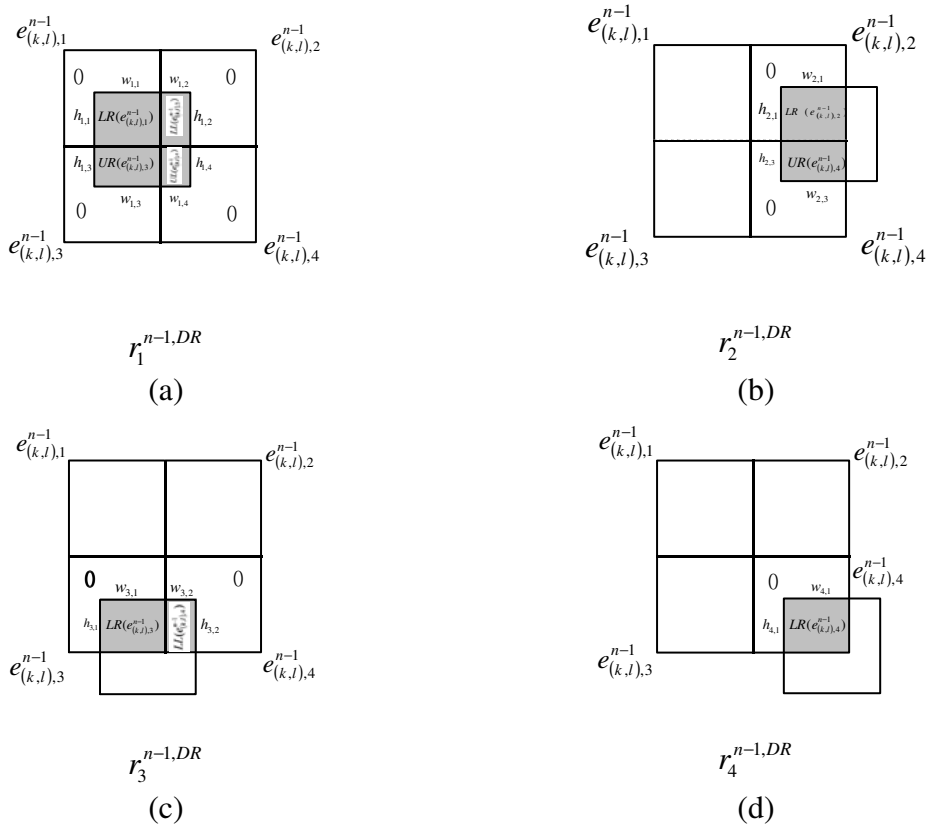


Figure 5.6. Contributions from the four overlapping blocks of (a) $r_1^{n-1,DR}$, (b) $r_2^{n-1,DR}$, (c) $r_3^{n-1,DR}$, and (d) $r_4^{n-1,DR}$.

In Figure 5.5, we assume that the motion vector $mv^n_{(k,l)}$ of $MB^n_{(k,l)}$ is (Δx, Δy). Then the shaded region in $e^{n-1}_{(k,l),1}$, $e^{n-1}_{(k,l),2}$, $e^{n-1}_{(k,l),3}$ and $e^{n-1}_{(k,l),4}$ are moved by (Δx, Δy). It is observed that

$$r_1^{n-1,DR} = r_1^{n-1} \tag{5.12}$$

Besides, $r_1^{n-1,DR}$ contains contributions from all $e^{n-1}_{(k,l),1}$, $e^{n-1}_{(k,l),2}$, $e^{n-1}_{(k,l),3}$ and $e^{n-1}_{(k,l),4}$. Specifically, it is composed of the lower-right corner (LR($e^{n-1}_{(k,l),1}$)) of $e^{n-1}_{(k,l),1}$, the lower-left corner (LL($e^{n-1}_{(k,l),2}$)) of $e^{n-1}_{(k,l),2}$, the upper-right corner (UR($e^{n-1}_{(k,l),3}$)) of $e^{n-1}_{(k,l),3}$, and the upper-left corner (UL($e^{n-1}_{(k,l),4}$)) of $e^{n-1}_{(k,l),4}$, as illustrated in Figure 5.6(a). In

this figure, we also supplement these contributions with zeros to image block with 8×8 pixels. Then, $r_1^{n-1,DR}$ can be computed as

$$r_1^{n-1,DR} = \mathrm{LR}(e_{(k,l),1}^{n-1}) + \mathrm{LL}(e_{(k,l),2}^{n-1}) + \mathrm{UR}(e_{(k,l),3}^{n-1}) + \mathrm{UL}(e_{(k,l),4}^{n-1}) \tag{5.13}$$

By taking the linearity of DCT, $r_1^{n-1,DR}$ in the DCT domain can be formulated as

$$\begin{aligned} DCT(r_1^{n-1,DR}) = {} & DCT(\mathrm{LR}(e_{(k,l),1}^{n-1})) + DCT(\mathrm{LL}(e_{(k,l),2}^{n-1})) \\ & + DCT(\mathrm{UR}(e_{(k,l),3}^{n-1})) + DCT(\mathrm{UL}(e_{(k,l),4}^{n-1})) \end{aligned} \tag{5.14}$$

Therefore, if we can explore the relationship between the shifted DCT sub-blocks in (5.14) ($DCT(\mathrm{LR}(e_{(k,l),1}^{n-1})$, $DCT(\mathrm{LL}(e_{(k,l),2}^{n-1})$, $DCT(\mathrm{UR}(e_{(k,l),3}^{n-1})$, and $DCT(\mathrm{UL}(e_{(k,l),4}^{n-1}))$ and the original DCT blocks ($DCT(e_{(k,l),1}^{n-1})$, $DCT(e_{(k,l),2}^{n-1})$, $DCT(e_{(k,l),3}^{n-1})$, and $DCT(e_{(k,l),4}^{n-1})$), then $DCT(r_1^{n-1,DR})$ can be calculated directly from the original encoded bitstream. To do so, we can rewrite (5.13) in the spatial-domain representation through matrix multiplications

$$r_1^{n-1,DR} = \sum_{i=1}^{4} s_i^{ver} e_{(k,l),i}^{n-1} s_i^{hor} \tag{5.15}$$

where s_i^{ver} and s_i^{hor} are matrices like $\begin{pmatrix} 0 & 0 \\ I_n & 0 \end{pmatrix}$ or $\begin{pmatrix} 0 & I_n \\ 0 & 0 \end{pmatrix}$ and I_n is identity matrix of size *n*. In (5.15), s_i^{ver} and s_i^{hor} are the pre-multiplication and post-multiplication matrices, and shift the sub-block of interest vertically and horizontally respectively. The sub-block of interest generally can be classified into 4 possible locations. They are upper left, upper right, lower left and lower right. In the equation of $r_1^{n-1,DR}$, the selection of matrices is summarized in the Table 5.1,

where the subscripts $h_{i,j}$ and $w_{i,j}$ are the number of rows and columns extracted and they are defined in Figure 5.6.

Table 5.1. Matrices s_i^{ver} and s_i^{hor} for $r_1^{n-1,DR}$.

Block	Position	s_i^{ver}	s_i^{hor}
$e_{(k,l),1}^{n-1}$	Lower right	$\begin{pmatrix} 0 & I_{h_{1,1}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{1,1}} & 0 \end{pmatrix}$
$e_{(k,l),2}^{n-1}$	Lower left	$\begin{pmatrix} 0 & I_{h_{1,2}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & I_{w_{1,2}} \\ 0 & 0 \end{pmatrix}$
$e_{(k,l),3}^{n-1}$	Upper right	$\begin{pmatrix} 0 & 0 \\ I_{h_{1,3}} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{1,3}} & 0 \end{pmatrix}$
$e_{(k,l),4}^{n-1}$	Upper left	$\begin{pmatrix} 0 & 0 \\ I_{h_{1,4}} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & I_{w_{1,4}} \\ 0 & 0 \end{pmatrix}$

Figure 5.7 shows an example illustrating how to extract $\mathrm{LR}(e_{(k,l),1}^{n-1})$. By multiplying $e_{(k,l),1}^{n-1}$ with s_1^{ver}, the last $h_{1,1}$ rows of $e_{(k,l),1}^{n-1}$ are extracted and translated to the top. After that, the resulted matrix is multiplied with s_1^{hor}. This post-multiplication matrix operation further extracts the last $w_{1,1}$ columns and translate this result to the left.

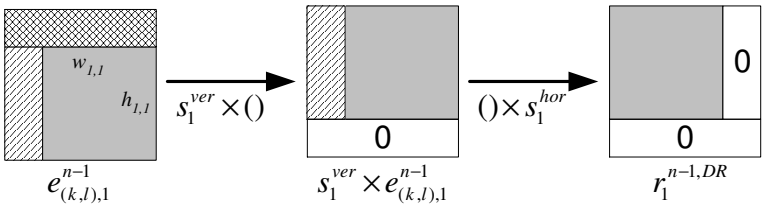


Figure 5.7. Example showing matrix multiplication to extract a sub-block and translate it to the appropriate location.

We are thus interested in obtaining $DCT(r_1^{n-1,DR})$ in the DCT domain. This can further reduce the computational burden of the server by avoiding the conversion

processes back and forth between the DCT domain and the spatial domain. It can be shown that all unitary orthogonal transforms such as the DCT are distributive to matrix multiplication, i.e.,

$$DCT(AB) = DCT(A)DCT(B) \tag{5.16}$$

By applying DCT to (5.15) and using the property of (5.16), (5.15) can be rewritten as

$$\begin{aligned} DCT(r_1^{n-1,DR}) &= DCT\left(\sum_{i=0}^{3} s_i^{ver} e_{(k,l),i}^{n-1} s_i^{hor}\right) \\ &= \sum_{i=0}^{3} S_i^{ver} E_{(k,l),i}^{n-1} S_i^{hor} \end{aligned} \tag{5.17}$$

where $S_i^{ver} = DCT\left(s_i^{ver}\right)$ and $S_i^{hor} = DCT\left(s_i^{hor}\right)$ are the shifting matrices in the DCT domain, and $E_{(k,l),i}^{n-1}$ denotes $DCT(e_{(k,l),i}^{n-1})$.

In the practical implementation, S_i^{ver} and S_i^{hor} can be pre-computed and stored in memory. Given a block size equal to *N*, only *2N-2* matrices need to be stored. $DCT(r_1^{n-1,DR})$ can then be extracted easily from the bitstream after the inverse VLC and de-quantization processes, which can reduce the required computations for MC MBs in the video server when applying RPS to the already encoded bitstream.

In Figure 5.6(b), $r_2^{n-1,DR}$ only contains contributions from $e_{(k,l),2}^{n-1}$ and $e_{(k,l),4}^{n-1}$ - the lower-right corner (LR($e_{(k,l),2}^{n-1}$)) of $e_{(k,l),2}^{n-1}$ and the upper-right corner (UR($e_{(k,l),4}^{n-1}$)) of $e_{(k,l),4}^{n-1}$. Therefore, $DCT(r_2^{n-1,DR})$ can be written as

$$DCT(r_2^{n-1,DR}) = DCT(\mathrm{LR}(e_{(k,l),2}^{n-1})) + DCT(\mathrm{UR}(e_{(k,l),4}^{n-1})) \tag{5.18}$$

By using the similar formulations from (5.15)- (5.17), (5.18) becomes

$$DCT(r_2^{n-1,DR}) = S_2^{ver} E_{(k,l),2}^{n-1} S_2^{hor} + S_4^{ver} E_{(k,l),4}^{n-1} S_4^{hor} \tag{5.19}$$

Table 5.2. Matrices s_i^{ver} and s_i^{hor} for $r_2^{n-1,DR}$

Block	Position	s_i^{ver}	s_i^{hor}
$e_{(k,l),2}^{n-1}$	Lower right	$\begin{pmatrix} 0 & I_{h_{2,1}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{2,1}} & 0 \end{pmatrix}$
$e_{(k,l),4}^{n-1}$	Upper right	$\begin{pmatrix} 0 & 0 \\ I_{h_{2,3}} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{2,3}} & 0 \end{pmatrix}$

Similarly, $r_3^{n-1,DR}$ is derived from contributions of $e_{(k,l),3}^{n}$ and $e_{(k,l),4}^{n}$ while $r_4^{n-1,DR}$ is contributed from $e_{(k,l),3}^{n}$ only. $r_3^{n-1,DR}$ and $r_4^{n-1,DR}$ can then be computed in the same way by using the suitable shifting multiplication matrices S_i^{ver} and S_i^{hor}, as defined in Table 5.3 and Table 5.4, respectively. They are represented as:

$$DCT(r_3^{n-1,DR}) = S_3^{ver} E_{(k,l),3}^{n-1} S_3^{hor} + S_4^{ver} E_{(k,l),4}^{n-1} S_4^{hor} \tag{5.20}$$

and

$$DCT(r_4^{n-1,DR}) = S_4^{ver} E_{(k,l),4}^{n-1} S_4^{hor} \tag{5.21}$$

Table 5.3. Matrices s_i^{ver} and s_i^{hor} for $r_3^{n-1,DR}$

Block	Position	s_i^{ver}	s_i^{hor}
$e_{(k,l),3}^{n-1}$	Lower right	$\begin{pmatrix} 0 & I_{h_{3,1}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{3,1}} & 0 \end{pmatrix}$
$e_{(k,l),4}^{n-1}$	Lower left	$\begin{pmatrix} 0 & I_{h_{3,2}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & I_{w_{3,2}} \\ 0 & 0 \end{pmatrix}$

Table 5.4. Matrices s_i^{ver} and s_i^{hor} for $r_4^{n-1,DR}$

Block	Position	s_i^{ver}	s_i^{hor}
$e_{(k,l),4}^{n-1}$	Lower right	$\begin{pmatrix} 0 & I_{h_{4,1}} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{w_{4,1}} & 0 \end{pmatrix}$

5.4.2 Chopping Operator

The DCT-domain coefficients for the second term in (5.11), i.e., $DCT(r_i^{n-1,\mathrm{DR}})$, has already been computed by using shifting operator. In this section, a similar idea can be applied to compute $DCT(e_{(k,l),i}^{n,\mathrm{DR}})$ in the DCT domain. Figure 5.8 shows the detailed block structure of $MB_{(k,l)}^n$ in frame *n*. $MB_{(k,l)}^n$ is composed of the prediction errors, $e_{(k,l),1}^n$, $e_{(k,l),2}^n$, $e_{(k,l),3}^n$ and $e_{(k,l),4}^n$. In the original encoded bitstream, their DCT coefficients ($DCT\left(e_{(k,l),1}^n\right)$, $DCT\left(e_{(k,l),2}^n\right)$, $DCT\left(e_{(k,l),3}^n\right)$, and $DCT\left(e_{(k,l),4}^n\right)$) can be easily extracted by performing inverse VLC and inverse quantization. These coefficients are useful for calculating $e_{(k,l),1}^{n,\mathrm{DR}}$, $e_{(k,l),2}^{n,\mathrm{DR}}$, $e_{(k,l),3}^{n,\mathrm{DR}}$ and $e_{(k,l),4}^{n,\mathrm{DR}}$, which are represented by the shaded region in Figure 5.8. In this Figure, it is found that $e_{(k,l),1}^{n,\mathrm{DR}}$ is equal to $e_{(k,l),1}^n$. Besides, $e_{(k,l),2}^{n,\mathrm{DR}}$, $e_{(k,l),3}^{n,\mathrm{DR}}$ and $e_{(k,l),4}^{n,\mathrm{DR}}$ are overlapping with $e_{(k,l),2}^n$, $e_{(k,l),3}^n$ and $e_{(k,l),4}^n$, respectively, without any shifting.

Therefore, instead of using the shifting operator mentioned in the previous section, a new chopping operator is necessary for extracting $e^{n,DR}_{(k,l),i}$ from $e^{n}_{(k,l),i}$.

Given the motion vector $mv^{n}_{(k,l)}$ of $MB^{n}_{(k,l)}$, (Δx , Δy), we can determine the necessary chopping region of each $e^{n,DR}_{(k,l),i}$. For example, in Figure 5.8, $w_2 = w_4 = 8 - \Delta x$, $h_3 = h_4 = 8 - \Delta y$, and $h_2 = w_3 = 8$.

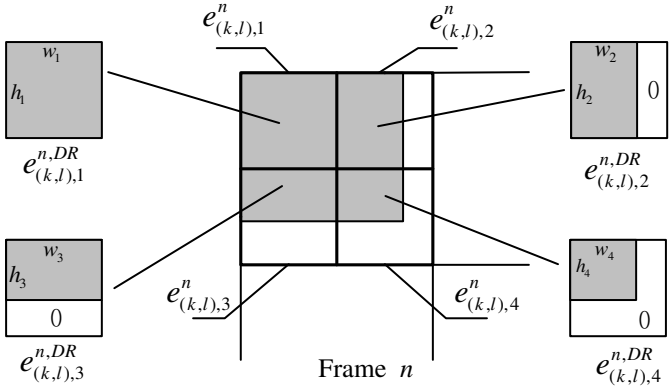


Figure 5.8. Block level in frame *n*.

The derivation of the shifting operator in (5.15) can be also done in that of the chopping operator. Given $e^{n}_{(k,l),i}$ and the values of h_i and w_i, we can describe $e^{n,DR}_{(k,l),i}$ in the spatial domain through matrix multiplications again.

$$e^{n,DR}_{i} = c^{row}_{h_i} e^{n}_{(k,l),i} c^{col}_{w_i} \tag{5.22}$$

where $c^{row}_{h_i}$ and $c^{col}_{w_i}$ are the row and column chopping matrices which are used to removes h_i rows and w_i columns in $e^{n}_{(k,l),i}$ and they are denoted by

$$c_{h_i}^{row} = \begin{pmatrix} I_{h_i} & 0 \\ 0 & 0 \end{pmatrix} \tag{5.23}$$

and,

$$c_{w_i}^{col} = \begin{pmatrix} 0 & 0 \\ 0 & I_{w_i} \end{pmatrix} \tag{5.24}$$

respectively. Note that I_n is identity matrix of size *n*.

In the example shown in Figure 5.8, the selected chopping matrices used for obtaining $e_{(k,l),1}^{n,DR}$, $e_{(k,l),2}^{n,DR}$, $e_{(k,l),3}^{n,DR}$ and $e_{(k,l),4}^{n,DR}$ are tabulated in Table 5.5.

Table 5.5. Chopping matrices $c_{h_i}^{row}$ and $c_{w_i}^{col}$ for $e_{(k,l),i}^{n,DR}$

Block	Target	c^{ver}	c^{hor}
$e_{(k,l),1}^{n}$	$e_{(k,l),1}^{n,DR}$	I_8	I_8
$e_{(k,l),2}^{n}$	$e_{(k,l),2}^{n,DR}$	I_8	$\begin{pmatrix} 0 & 0 \\ 0 & I_{w_2} \end{pmatrix}$
$e_{(k,l),3}^{n}$	$e_{(k,l),3}^{n,DR}$	$\begin{pmatrix} I_{h_3} & 0 \\ 0 & 0 \end{pmatrix}$	I_8
$e_{(k,l),4}^{n}$	$e_{(k,l),4}^{n,DR}$	$\begin{pmatrix} I_{h_4} & 0 \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 0 & I_{w_4} \end{pmatrix}$

By applying DCT to (5.22) and making use of distributive property in (5.16), we obtain

$$DCT(e_i^{n,DR}) = C_{h_i}^{row} E_{(k,l),i}^{n} C_{w_i}^{col} \tag{5.25}$$

where $C_{h_i}^{row} = DCT\left(c_{h_i}^{row}\right)$ and $C_{w_i}^{col} = DCT\left(c_{w_i}^{col}\right)$ are the chopping matrices in the DCT domain and they can be pre-stored in memory. This means that the chopping operator yields further speed up in the computation of MC MBs in the server.

5.6 Region Tracking for $\overline{\mathrm{DR}}$s

After obtaining the required prediction error of the DR, we now examine how the DCT coefficients of prediction errors in the $\overline{\mathrm{DR}}$, $DCT(\hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}})$, which is another necessary component of $DCT(\hat{e}_{(k,l)}^{n})$ in (5.10), can be calculated. As discussed in Section 5.3, $DCT(\hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}})$ is impossible to re-use DCT-domain coefficients from the existing bitstreams. In this case, the conventional re-encoding technique operated in the spatial domain is needed to compute $DCT(\hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}})$. In other words, $\hat{e}_{(k,l)}^{n,\overline{\mathrm{DR}}}$ is obtained by subtracting the pixel intensities of the reconstructed MB in frame *n* with the corresponding best-matched MB in the new reference, frame *n-2*. That is,

$$\hat{e}_{(k,l)}^{n,\overline{DR}} = MB_{(k,l)}^{n,\overline{DR}} - MB_{(k,l)}^{n-2,\overline{DR}} \qquad (5.26)$$

For the conventional re-encoding technique, all MBs from the previously nearest I-frame to frame *n-1* should be decoding. Then re-encoding is performed for those MC MBs with frame *n-2* as the reference. With the help of various DCT-domain techniques, we suggest using a region tracking technique to trace only the indispensable MBs during the decoding and re-encoding processes. This

technique traces from the current frame to the previously nearest I-frame and only the necessary MBs related to $\overline{\mathrm{DR}}$ needs to be decoded.

We will now explain the significance of using the region tracking technique. For comparison, an example for re-encoding MC MBs without the region tracking technique is depicted in Figure 5.9(a). This figure shows a situation in which $MB^n_{(0,1)}$ and $MB^n_{(1,1)}$ are both MC MBs in the current frame (frame *n*), and their corresponding motion vectors are denoted by the dotted arrows. Frame *n-1* is the corrupted frame and frame *n-2* is the new reference. To re-encode these MC MBs of frame *n* directly, the server will examine the motion vectors in the video bitstream and all the related MBs from frame *n-3* to frame *n* should be decoded. Three MBs (the shaded MBs) in frame *n-1* are required to act as references for performing motion compensation of $MB^n_{(0,1)}$ and $MB^n_{(1,1)}$. These MBs in frame *n-1* further requires their corresponding MBs in frame *n-2*. This process continues until the previously nearest I-frame. In this example, the server needs to decode 19 MBs for $MB^n_{(0,1)}$ and $MB^n_{(1,1)}$ from frame *n-3* to frame *n*. This situation gets worse when the current transmitted frame is far away from the previously nearest I-frame. However, by applying the region tracking technique, we find that only $\overline{\mathrm{DR}}$s of MC MBs needs to be re-encoded in the pixel domain. Therefore, some decoded MBs in Figure 5.9(a) do not actually contribute to $MB^n_{(0,1)}$ and $MB^n_{(1,1)}$. The idea of the region tracking technique is to identify the MBs having actual contribution to $\overline{\mathrm{DR}}$s of MC MBs in the previous frame. Let us use Figure 5.9(b) to give a clearer account of our idea. In Figure 5.9(b), both $MB^n_{(0,1)}$ and $MB^n_{(1,1)}$

are divided into two regions – the DR and the $\overline{\text{DR}}$. The $\overline{\text{DR}}$ is denoted by the block filled with diagonal lines (▨), which is also the actual region to be re-encoded in the pixel domain. Since the number of pixels in each $\overline{\text{DR}}$ is smaller than that in the whole MB, the number of MBs to be decoded in the previous frames can be reduced. In this example, only 2, 2, and 4 MBs are necessary to be required in frame *n-1, n-2* and *n-3* respectively. This means that the necessary MBs used to re-encode the current MC MBs can be saved considerably. If the length of GOP is longer, the savings of the proposed technique could be even larger.

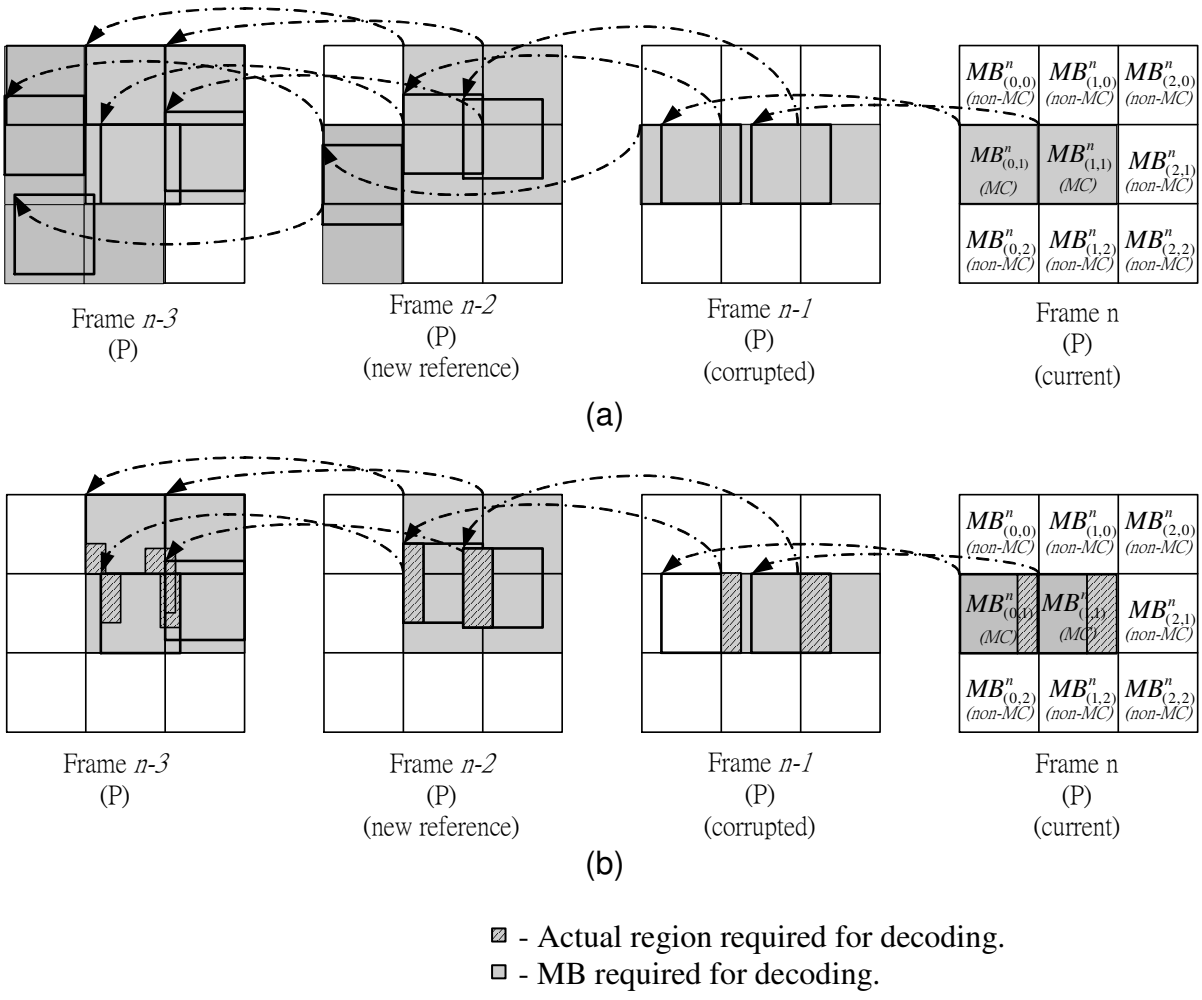


Figure 5.9. The situation of MC MBs adopting our proposed DCT-domain techniques (a) with region tracking and (b) without region tracking.

5.6 Chapter Summary

In this chapter, we have designed two DCT-domain operators for shifting and chopping the DCT coefficients of an already MPEG encoded bitstream in order to compute the DCT coefficients of the target block in MC MBs. Through the matrix manipulation, the shifting operator can obtain some related DCT coefficients from the coefficients of its four overlapping DCT blocks in the original encoded bitstream while the chopping operator can extract a part of DCT coefficients from the existing block. Both operators are carried out on the DCT domain. With the help of FDVS, these operators can be operated as much as possible in the DCT form in order to keep the benefits of the DCT-domain manipulation.

Although the DCT-domain operators can reduce the required computations for MC MBs in the server, some regions still require the conventional re-encoding technique, which is operated on the spatial domain. To minimize the use of the re-encoding technique, a region tracking technique is employed to trace only the essential MBs during the re-encoding process. It can ensure that only the necessary MBs related to regions, that cannot be manipulated by the DCT-domain operators, need to be decoded. Simulation results in the Chapter 6 will show that the techniques proposed here can further minimize server complexity significantly as well as preserving the video quality. As a concluding remark, all the techniques in this chapter can strength the practicality of implementing RPS in the already MPEG encoded bitstream.

All of our proposed techniques in Chapters 4 and 5 require similar memory usage as the conventional RPS. Our proposed algorithm requires one picture

memory for holding the reference frame and one for the coefficients of the current frame. Some extra memory is needed during region tracking for efficient MB decoding. Moreover, our proposed algorithm can be performed in one pass and it is frame-by-frame basis. Each video frame in the bitstream is processed once in order to avoid the inefficient scattered memory access time, which seriously affects the performance on almost all of the typical cache-limited processors. Hence, our algorithm should be able to implement efficiently in both of the general purpose computer and the hardware-oriented system.

Chapter 6

Experimental Results and Discussions

6.1 Introduction

In this chapter, a large amount of experimental works has been conducted to evaluate the performances of the proposed techniques including the direct addition of quantized DCT coefficients (DA), the modified quantizer-dequantizer pair ($\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$) for non-MC MBs, and the DCT-domain shifting and chopping operators of DCT coefficients for MC MBs when applying RPS in already MPEG encoded bitstreams. Comparisons including the computational requirements of the video server/proxy and the quality of reconstructed frames are given on the performances of different proposed techniques and the conventional re-encoding technique [55] when applied to the RPS scheme. The conventional re-encoding technique refers to the spatial-domain approach for decoding all related MBs from the previously nearest I-frame to the current transmitted frame, as mentioned in Chapter 3. This current frame is then re-encoded with the new reference which has been correctly decoded in the decoder.

For our proposed techniques, three different versions have been realized. Let us call them RPS-DA, RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ and MCMB+ RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$. RPS-DA uses the direct addition technique of quantized DCT coefficients for non-MC MBs while RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ employs the modified quantizer-dequantizer pair. For

MCMB+RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$, it further extends RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ to use FDVS for composing the new motion vectors, and shifting and chopping operators for creating the corresponding prediction errors of MC MBs in the DCT domain.

6.2 Simulation Conditions

The MPEG-4 encoder with main profile [21] was employed to encode a large variety of real video sequences with different spatial resolutions, motion characteristics and bitrates. All the test sequences have a length of *140* frames. "Mother and Daughter", "Salesman", "Calendar" and "Foreman" are in CIF (*352×288* pixels) format while "Football" and "Table Tennis" are in SIF (*352×240* pixels) format. "Carphone" is a typical videophone sequence in QCIF (*176×144 pixels*) format. The first frame of each sequence is shown in Figure 6.1. All the sequences were pre-encoded at three different bitrates. For all test sequences, the frame-rate of the video stream was *30* frames/s and the GOP length was 15 with an I-P structure.

6.3 Computational Complexity of the Video Server

In this section, the computational requirements in the server of our proposed techniques will be evaluated. In order to measure the computational requirements of the server associated with various RPS techniques, we adopted a cost that can directly reflect the computational burden of the server when one particular RPS technique is adopted.

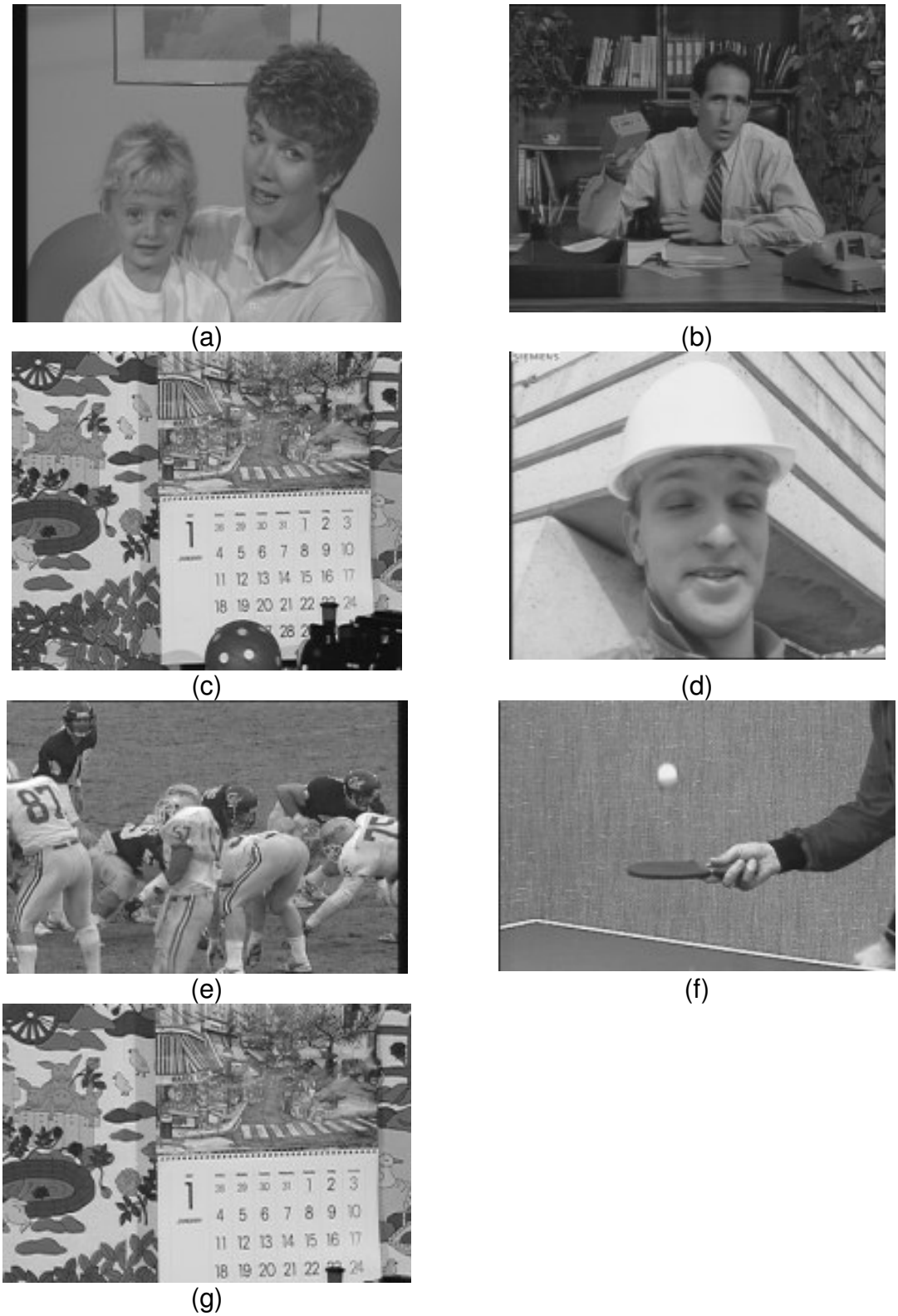


Figure 6.1 The first frames of various sequences. (a) Mother and Daughter, (b) Salesman, (c) Calender, (d) Foreman, (e) Football, (f) Table Tennis, and (g) Carphone.

For the RPS scheme using the conventional re-encoding technique, the server requires to decode all MBs in P-frames from the previously nearest I-frame to the current transmitted frame, and re-encode all the MBs in the current frame with a new reference. The major steps in decoding a single MB are two-dimensional inverse DCT (2D-IDCT), dequantization and motion compensation operations while the re-encoding process involves two-dimensional DCT (2D-DCT), quantization, motion estimation and compensation operations. The main difference in computational complexity between the decoding and re-encoding processes is the motion estimation operation. Actually, it depends on the motion estimation algorithm to be used for the re-encoding process. In general, measuring the difference between the costs of these two processes would not be an easy task. In spite of these, it is possible to assume their complexities are the same for the sake of simplicity. In the following comparisons, we thus approximate the cost to the required number of MBs to be decoded and re-encoded in the server.

To re-encode all MBs in the current transmitted frame of the proposed RPS-DA and RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, the server will examine the motion vectors in the already MPEG encoded bitstream and only the related MBs from the previously nearest I-frame of the current frame should be decoded. We use the example in Figure 5.9(a) again. In this example, the server needs to decode 19 MBs for $MB_{(0,1)}^{n}$ and $MB_{(1,1)}^{n}$ from frame *n* to frame *n-3* instead of 36. In this situation, the number of necessary MBs to be decoded in the server can be reduced as compared with the pixel-domain approach. For non-MC MBs in the current frame, our proposed MB-based techniques can either use DA or $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ to do the re-encoding

process. From (4.13) and (4.19), only simple arithmetic operations are required instead of the complicated decoding and re-encoding processes. Note that the computational requirement of RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ is quite similar to that of RPS-DA, and only two more addition and multiplication operations are required. Practically, the multiplication by 2 in (4.19) can be implemented by a shift operator. Therefore, these arithmetic operations are negligible as compared with the 2D-DCT or 2D-IDCT operation.

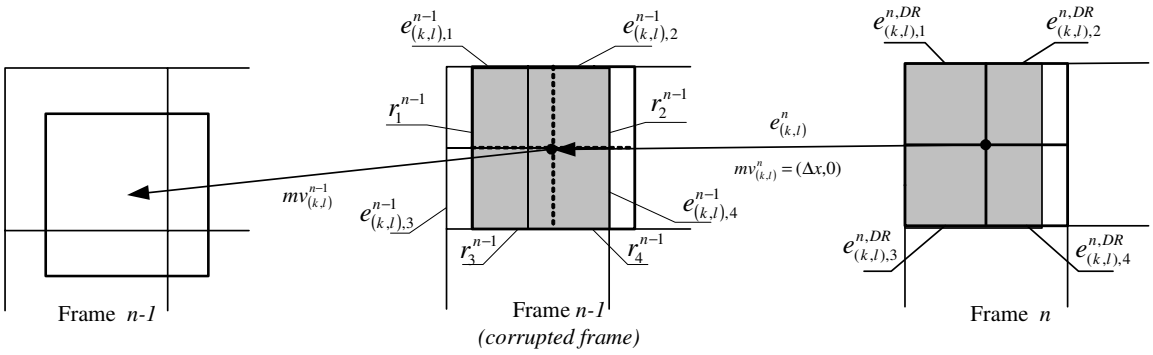
For using MCMB+RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ in the RPS scheme, the DCT-domain shifting and chopping operators are proposed to use in handling MC MBs. These operators involve some matrix manipulations and they construct the DCT-domain values of each 8×8 target block separately. The operators need the computation of the DCT coefficients of its contributing block. In general, a target block is predicted from up to four contributing blocks. The computation of DCT coefficients of contributing blocks actually requires pre- and post-multiplication of an 8×8 data block with two 8×8 matrices, where the matrices can be preprocessed and stored in the memory of the video server. Given an 8×8 data block D and two 8×8 matrices M_{pre} and M_{post}, the computation of $M_{pre}DM_{post}$ includes two matrix multiplications. Let assume, for the sake of argument, that the type $M_{pre}DM_{post}$ is called a TM operation. In the following, the TM operation is taken as the unit to measure and compare the computational complexity of the proposed DCT-domain operators. Since DCT and IDCT operations cover the majority computations of the whole server/proxy process, using TM operation as the basic unit of computational complexity comparison can realistically reflect the actual efficiency of different algorithms. For using the shifting operator in the

example shown in Figure 5.5, equations (5.17), (5.19) to (5.21) show that the computation of $r_1^{n-1,DR}$, $r_2^{n-1,DR}$, $r_3^{n-1,DR}$ and $r_4^{n-1,DR}$ requires 4 TM operations, 2 TM operations, 2 TM operations, and 1 TM operation, respectively. Therefore, 9 TM operations are needed to obtain $r^{n-1,DR}$. Besides, the chopping operators require 3 TM operations to compute all $r_2^{n,DR}$, $r_3^{n,DR}$ and $r_4^{n,DR}$, as discussed in Section 5.4.2. Putting all of these together, 12 TM operations are required to compute the new prediction errors $\hat{e}_{(k,l)}^n$ in this example when MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ is adopted in handling one MC MB. On the other hand, for using the conventional re-encoding technique in RPS, the major computation is from 2D-IDCT. Since IDCT is a separable transform, its 2D version can be expressed in a matrix form x=C^TXC, where X and *x* are the 8×8 matrices containing DCT coefficients and pixel values, respectively. *C* is the 8×8 transform kernel and C^T is the transpose of *C*. Therefore, 4 TM operations are required for each 16×16 MB. Since 2D-IDCT occupies the most time consuming part in decoding one MB and the DCT-domain operators are the major step to obtain the new prediction errors for MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, the computational requirement of the proposed DCT-domain operators for manipulating one MB is nearly equal to three (12/4) times as that of decoding one MB by using the conventional re-encoding technique. Actually, the required number of TM operations of the proposed MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ depends on the motion vector $mv_{(k,l)}^n$ of $MB_{(k,l)}^n$. Table 6.1 summarizes the actual number of TM operations for $mv_{(k,l)}^n$ with different combinations of horizontal (Δx) and vertical (Δy) displacements. If there is only horizontal or vertical displacement, the DCT coefficients of a target block requires the construction of DCT-domain values of up to two contributing blocks only. Its

detailed computations for the 8×8 target blocks are shown in Figure 6.2(a) (Figure 6.2(b)) for zero vertical displacement (zero horizontal displacement). In these two cases, it can be found that the computational requirement of the proposed DCT-domain operators for manipulating one MB is reduced to two times as that of decoding one MB by using the conventional re-encoding technique, as shown in Table 6.1.

Table 6.1. The required numbers of TM operations for $mv^{n}_{(k,l)}$ with different combinations of horizontal (Δx) and vertical (Δy) displacements when the shifting and chopping operators are used. The maximum displacement of Δx and Δy are confined to ±8 pixels.

	Shifting operator (TM operations)				Chopping operator (TM operations)				Total TM operations (equivalent to no. of MBs to be decoded)
	$r_1^{n-1,DR}$	$r_2^{n-1,DR}$	$r_3^{n-1,DR}$	$r_4^{n-1,DR}$	$e_{(k,l),1}^{n,DR}$	$e_{(k,l),2}^{n,DR}$	$e_{(k,l),3}^{n,DR}$	$e_{(k,l),4}^{n,DR}$	
Δx≠0 and Δy≠0	4	2	2	1	0	1	1	1	12 (3)
0<|Δx|<8 and Δy=0	2	1	2	1	0	1	0	1	8 (2)
Δx=0 and 0<|Δy|<8	2	2	1	1	0	0	1	1	8 (2)



Frame *n-1*

$$r_1^{n-1} = e_{(k,l),1}^{n-1}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} + e_{(k,l),2}^{n-1}\begin{pmatrix} 0 & 0 \\ 0 & I_{8-\Delta x} \end{pmatrix} \quad \text{(2 TM operations)}$$

$$r_2^{n-1} = e_{(k,l),2}^{n-1}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} \quad \text{(1 TM operation)}$$

$$r_3^{n-1} = e_{(k,l),3}^{n-1}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} + e_{(k,l),4}^{n-1}\begin{pmatrix} 0 & 0 \\ 0 & I_{8-\Delta x} \end{pmatrix} \quad \text{(2 TM operations)}$$

$$r_4^{n-1} = e_{(k,l),4}^{n-1}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} \quad \text{(1 TM operation)}$$

Frame *n*

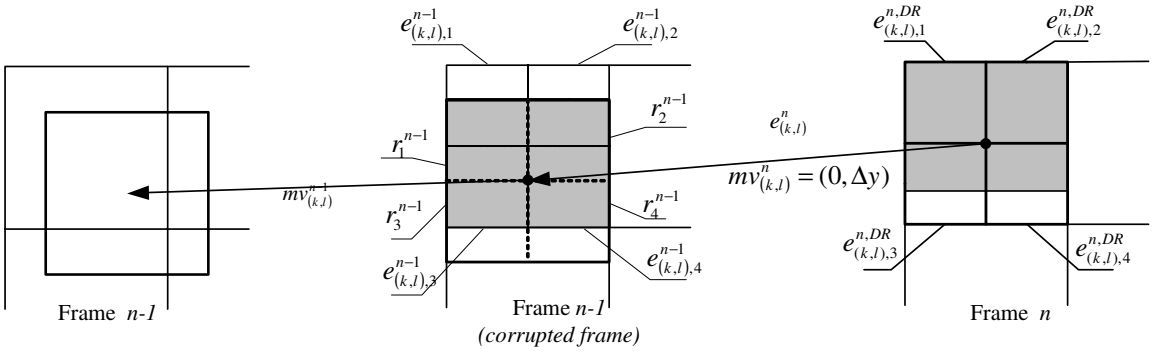
$$e_{(k,l),1}^{n,DR} = e_{(k,l),1}^{n} \quad \text{(0 TM operation)}$$

$$e_{(k,l),2}^{n,DR} = e_{(k,l),2}^{n}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} \quad \text{(1 TM operation)}$$

$$e_{(k,l),3}^{n,DR} = e_{(k,l),3}^{n} \quad \text{(0 TM operation)}$$

$$e_{(k,l),4}^{n,DR} = e_{(k,l),4}^{n}\begin{pmatrix} I_{\Delta x} & 0 \\ 0 & 0 \end{pmatrix} \quad \text{(1 TM operation)}$$

(a)



Frame *n-1*

$$r_1^{n-1} = \begin{pmatrix} 0 & I_{\Delta y} \\ 0 & 0 \end{pmatrix} e_{(k,l),1}^{n-1} + \begin{pmatrix} 0 & 0 \\ I_{8-\Delta y} & 0 \end{pmatrix} e_{(k,l),3}^{n-1} \quad \text{(2 TM operations)}$$

$$r_2^{n-1} = \begin{pmatrix} 0 & I_{\Delta y} \\ 0 & 0 \end{pmatrix} e_{(k,l),2}^{n-1} + \begin{pmatrix} 0 & 0 \\ I_{8-\Delta y} & 0 \end{pmatrix} e_{(k,l),4}^{n-1} \quad \text{(2 TM operations)}$$

$$r_3^{n-1} = \begin{pmatrix} 0 & I_{\Delta y} \\ 0 & 0 \end{pmatrix} e_{(k,l),3}^{n-1} \quad \text{(1 TM operation)}$$

$$r_4^{n-1} = \begin{pmatrix} 0 & I_{\Delta y} \\ 0 & 0 \end{pmatrix} e_{(k,l),4}^{n-1} \quad \text{(1 TM operation)}$$

Frame *n*

$$e_{(k,l),1}^{n,DR} = e_{(k,l),1}^{n} \quad \text{(0 TM operation)}$$

$$e_{(k,l),2}^{n,DR} = e_{(k,l),2}^{n} \quad \text{(0 TM operation)}$$

$$e_{(k,l),3}^{n,DR} = \begin{pmatrix} I_{\Delta y} & 0 \\ 0 & 0 \end{pmatrix} e_{(k,l),3}^{n} \quad \text{(1 TM operation)}$$

$$e_{(k,l),4}^{n,DR} = \begin{pmatrix} I_{\Delta y} & 0 \\ 0 & 0 \end{pmatrix} e_{(k,l),4}^{n} \quad \text{(1 TM operation)}$$

(b)

Figure 6.2. Number of TM operations for DCT-domain shifting and chopping operators with (a) zero vertical displacement, and (b) zero horizontal displacement.

By taking all these considerations, the detailed comparisons of the computational savings with different RTD of the proposed techniques are tabulated in Table 6.2. All data in Table 6.2 are the average savings when an error is occurred in all possible frames. We show that all RPS-DA, RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, and MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ outperform the RPS scheme using the conventional re-encoding technique in all sequences. For RPS-DA and RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, Table 6.2 shows that the results are more significant for sequences "Mother and Daughter" and "Salesman" in which 50% to 80% savings in server burden is achieved. It is due to the reason that these sequences contain more non-MC MBs, in which the technique of DA or $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ can be performed more efficiently. For sequences containing high motion activities such as "Football", "Calendar", "Table Tennis", "Foreman" and "Carphone", the RPS-DA and RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ still have good savings in terms of about 20-40%. To further reduce the number of MBs to be decoded, the DCT-domain operators proposed in Chapter 5 can work with RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ for MC MBs, MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$. Table 6.2 also shows that, by applying the DCT-domain shifting and chopping techniques to MC MBs, MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ produces further savings, about 10%-35%, as compared with that of RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$. This can be explained by the benefits of the DCT-domain operators which can manipulate the prediction errors as much as possible in the compressed form. Besides, the region tracking technique can further minimize the number of MBs to be decoded, as shown in Figure 5.9. In Table 6.2, it is noted that the savings of MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ drops as

RTD_dist increases. This is the drawback of our proposed techniques. The reason behind is that the area of the dominant region in MC MB reduces quickly for a longer RTD. Eventually, more MBs are required to undergo the pixel-domain process. However, significant amount of savings over the conventional approach can still be found when RTD is equal to 3, as shown in Table 6.2. Note that the performances of proposed techniques must be better than that of the conventional one even the RTD is exceptionally large. It is because the worst case of the proposed algorithms is to decode and re-encode all frames within the GOP in the situation that error occurs in the I-frame and RTD is equal to the length of GOP at the same time. However, in this rare situation, our proposed techniques can still keep the same computational complexity as the conventional approach. Therefore, the proposed techniques are very promising solution.

Table 6.2. Savings of using the proposed MB-based techniques as compared with the conventional re-encoding technique in RPS.

Sequences	Input bitrate	$RPS-DA$ or $RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ (%)			$MCMB+RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ (%)		
		RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3
Mother and daughter	537 kbps	78.5	77.5	76.9	91.2	91.5	86.2
	249 kbps	77.7	76.8	76.2	90.9	90.9	85.9
	112 kbps	74.6	73.4	72.6	88.2	85.6	83.7
Salesman	1.14 Mbps	47.1	48.3	49.6	71.2	68.6	67.7
	686 kbps	50.0	51.4	52.6	73.6	70.9	70.5
	256 kbps	57.7	58.4	59.1	80.0	76.8	76.4
Football	2.25 Mbps	42.4	42.4	42.6	62.8	59.3	56.9
	1.54 Mbps	43.0	42.9	43.0	63.2	59.7	57.3
	898 kbps	43.0	42.9	43.2	63.3	59.8	57.5
Calendar	1.64 Mbps	23.8	23.4	23.2	37.3	34.4	32.5
	995 kbps	23.3	22.9	22.7	37.3	34.0	32.1
	828 kbps	23.1	22.6	22.4	37.5	33.7	31.7
Table tennis	1.44 Mbps	31.2	30.5	30.1	43.1	40.2	38.3
	934 kbps	31.4	30.6	30.2	43.3	40.2	38.2
	504 kbps	31.0	30.3	29.9	44.0	40.6	38.6
Foreman	1.12 Mbps	18.4	14.7	10.8	35.1	27.0	20.4
	712 kbps	17.6	13.8	10.1	34.6	26.1	19.3
	354 kbps	17.0	13.6	10.2	33.9	25.2	19.1
Carphone (qcif)	457 kbps	29.7	26.1	24.1	64.7	51.2	41.5
	278 kbps	29.5	25.9	23.8	64.6	52.3	47.2
	102 kbps	29.0	25.2	23.2	64.9	52.2	43.1

6.4 Quality Performance

We use the Peak Signal-to-Noise Ratio (PSNR) as the measure for the quality performance. The PSNR, defined in terms of the reconstructed frame and the original frame, is used to compare the quality performance of the proposed techniques with that of the conventional re-encoding technique. PSNR is defined as

$$PSNR\,(dB) = 10\log_{10}\frac{255^2}{MSE} \tag{6.1}$$

where MSE is the mean squared error between the reconstructed frame relative to the original frame .

The detailed comparisons of the average PSNR among different techniques for the succeeding frame right after the corrupted frame are tabulated in Table 6.3. The PSNR performances of the four techniques, the conventional re-encoding technique, RPS-DA, RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, and MCMB+RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, are compared. This table shows only the results of non-MC MBs in which either RPS-DA, RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ or MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ is used. Since both RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ and MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ use the $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ technique for non-MC MBs, they produce the same PSNR results. Owing to this reason, we put their PSNR results in the same column in Table 6.3. We show that the PSNR performance of the proposed RPS-DA is better than that of the re-encoding technique when the round-trip delay (RTD) corresponds to the duration of encoding time for one frame. When RTD is greater than 1, the average PSNR performance of RPS-DA drops, or it is even worse than the re-encoding

technique in some sequences such as “Calendar”, “Football” and “Table Tennis”. This is because the problem of the non-linear property of inverse quantization in RPS-DA becomes more significant for a longer RTD. This problem can be revolved by using RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$/MCMB+ RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, as shown in Table 6.3. For different values of RTD, the average PSNR of RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$/MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ is nearly the same as the desired output, which is the average PSNR of the already MPEG encoding bitstream for all non-MC MBs. Note that values of the average PSNR values have slightly distinct for various values of RTD in the desired output. The reason behind is that, for *RTD > 1*, the numbers of non-MC MBs in various values of RTD are different since a MB is defined as non-MC only if its related MBs with the same spatial location of consecutive frames have zero motion vectors, as depicted in Figure 4.4. Theoretically, RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ for non-MC MBs will not introduce any quality degradation. However, Table 6.3 shows a negligibly small PSNR degradation in RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$/ MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$. The main reason for this negligibly small degradation in PSNR is due to the assumption of linearity in the derivation of equation (4.12). The assumption is given as

$$DCT^{-1}\left(Q_L^{-1}\left(E_{(k,l)}^{n}\right)\right)+DCT^{-1}\left(Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)\right)=DCT^{-1}\left(Q_L^{-1}\left(E_{(k,l)}^{n}\right)+Q_L^{-1}\left(E_{(k,l)}^{n-1}\right)\right) \qquad (6.2)$$

As mentioned in Chapter 4, this equation builds the fundamentals to derive both RPS-DA and RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$/ MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, but is only valid for the IDCT’s implementation without precision inaccuracy. In practice, the MPEG-4 codec [58] conforming to the precision specified in [59] uses a finite wordlength to implement IDCT which will incur rounding errors for each IDCT operation. For the desired output, two IDCTs (left-hand side of (6.2)) with precision inaccuracy

are performed. On the other hand, only one (right-hand side of (6.2)) is carried out for RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$. Therefore, in comparison with the desired output, some arithmetic inaccuracy is introduced which has little effect on the reconstruction quality of non-MC MBs for *RTD=1*. For longer RTD, more cascaded IDCT operations are needed for the desired output, and this causes a further drop in PSNR of the RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$, but this drop is still insignificant as shown in Table 6.3.

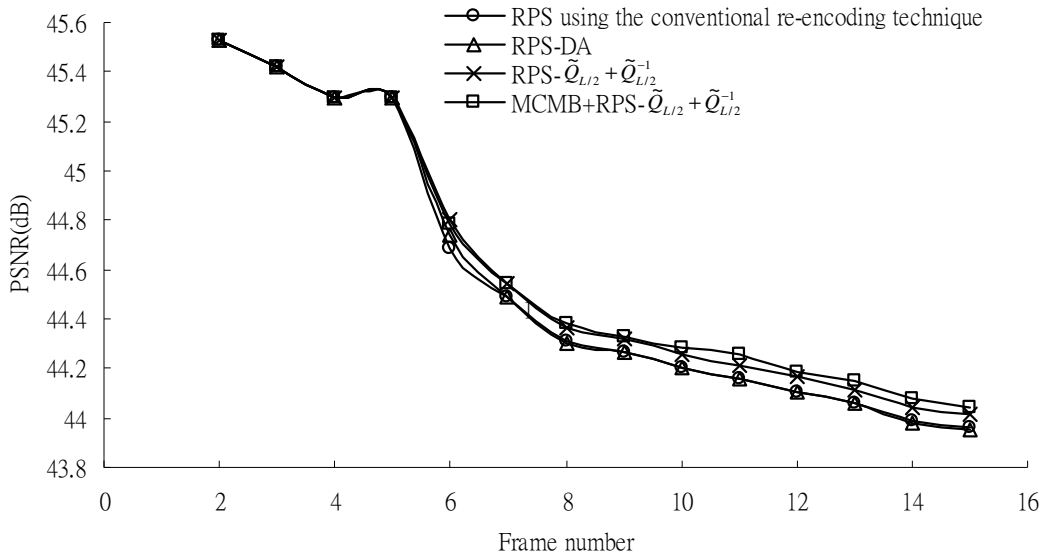
From Table 6.3, the PSNR improvement of RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ / MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ is much more remarkable at high bitrate. This is due to the fact that, for low bitrate, a considerable percentage of the DCT blocks have a significant amount of zero elements. In this case, re-encoding errors will not be introduced by using the re-encoding technique.

Table 6.4 shows the PSNR performances of all MBs for the succeeding frame after the corrupted frame of using the re-encoding technique, RPS-DA, RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ and MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$. The result is consistent with Table 6.3, which is expected since a better PSNR of the non-MC MBs is contributed to the current transmitted frame, which is encoded with a new reference.

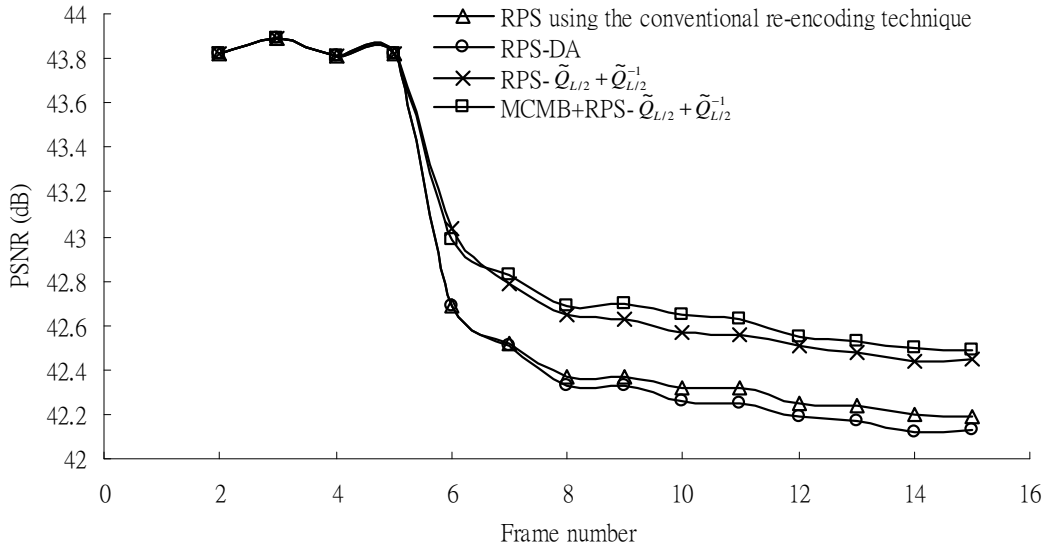
Furthermore, the quality degradation will not only be confined to the frame at the re-encoding point but can propagate and be accumulated in the subsequent P frames. Such drift will last until the next I-frame. In Figure 6.3, we have realized the effect of drift by using the re-encoding technique and our proposed MB-

based techniques for different video sequences. It can be seen from the Figure 6.3 that the proposed RPS-DA, RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ and MCMB+RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ have remarkable PSNR improvements over the re-encoding technique for different sequences. This further demonstrates the effect of the proposed techniques when applied to RPS in the already encoded MPEG bitstream. That is, our proposed techniques can provide a better subjective quality during video playback.

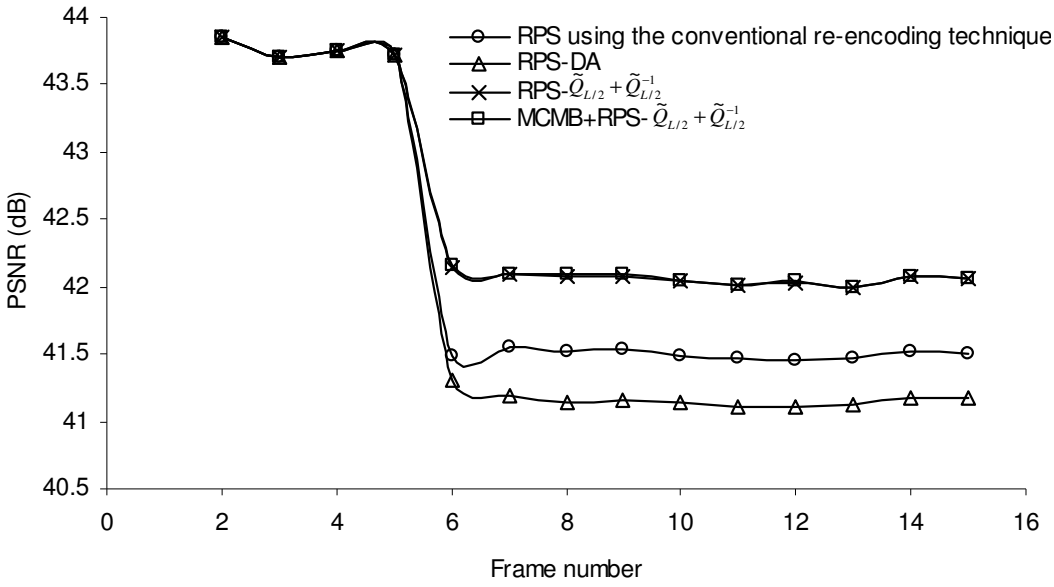
For MCMB+RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$, its PSNR performance is almost the same as that of RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$. This result reflects that the performance of FDVS, which performs motion approximation by using the existing motion information, gives virtually the same performance as the full-scale full search motion estimation technique. Nevertheless, the computational complexity of FDVS is substantially lower than that of the full-scale full search motion estimation technique. This observation can be further confirmed by the results shown in Tables 6.5 – 6.7. These tables show that the PSNR comparisons of the reconstructed frame after the corrupted frame. They include the results of the re-encoding and three proposed techniques along with the desired output for different values of RTD. All results are shown in details involving the PSNR of non-MC MBs, MC MBs and the overall average. Again, it can be observed that the video quality of MC MBs in MCMB+RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ gives very similar result as that of RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$, which uses full-scale full search motion estimation, for different bitrates and RTD values.


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR(dB)
Frame number

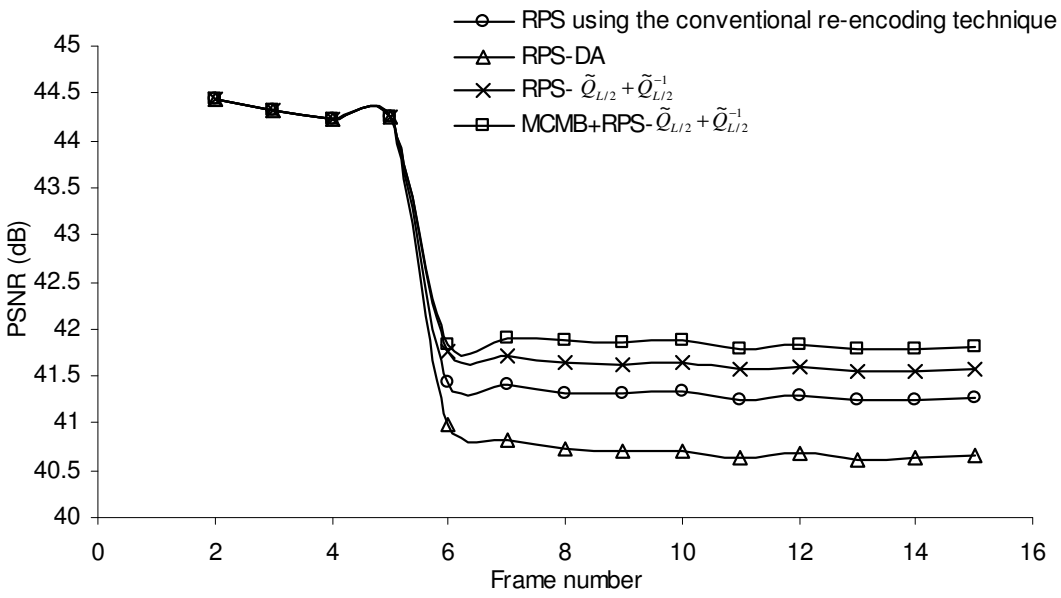
(a)


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR (dB)
Frame number

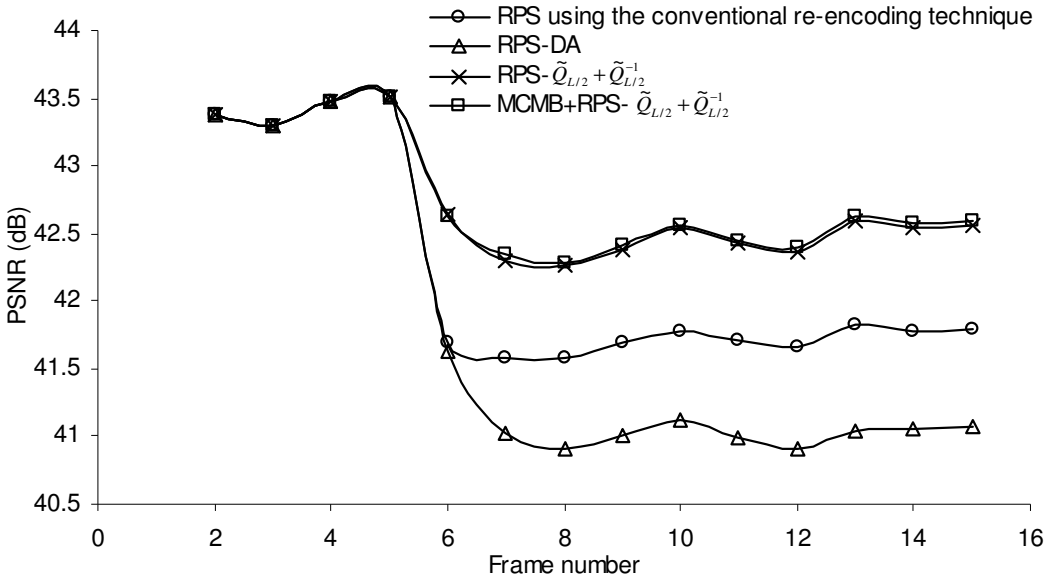
(b)


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR (dB)
Frame number
44
43.5
43
42.5
42
41.5
41
40.5
0
2
4
6
8
10
12
14
16

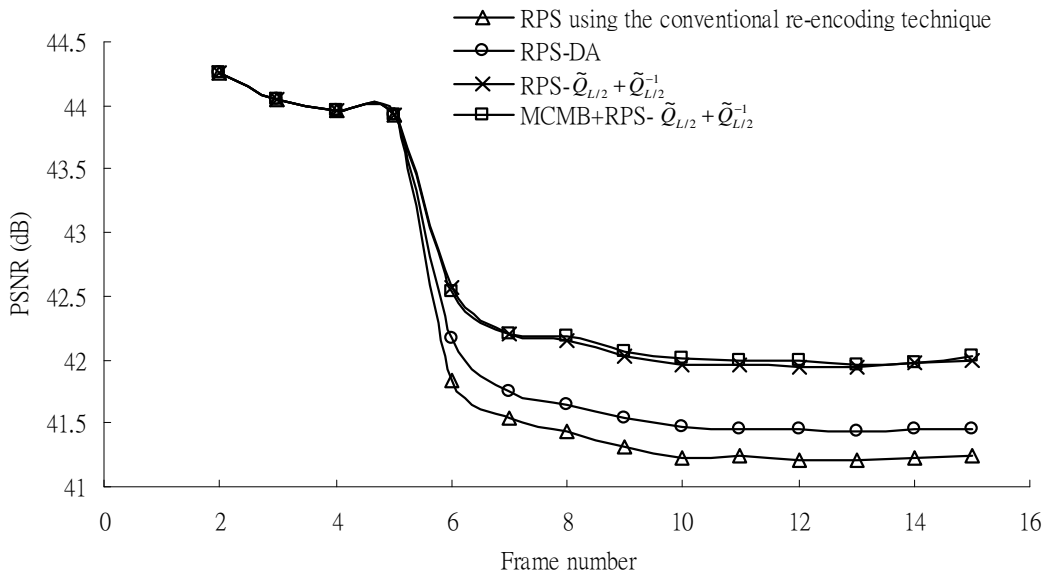
(c)


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR (dB)
Frame number
45
44.5
44
43.5
43
42.5
42
41.5
41
40.5
40
0
2
4
6
8
10
12
14
16

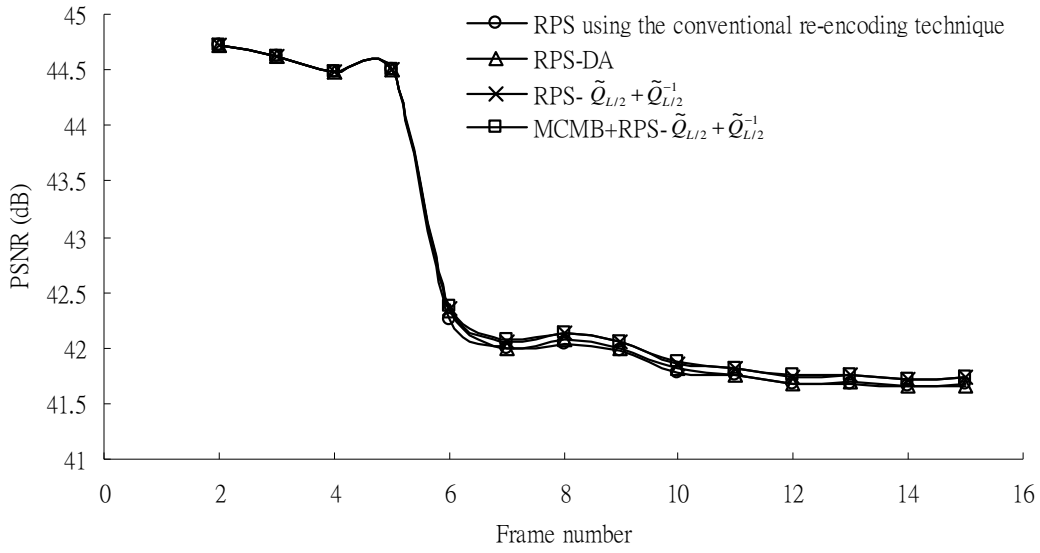
(d)


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR (dB)
Frame number
44
43.5
43
42.5
42
41.5
41
40.5
0
2
4
6
8
10
12
14
16

(e)


RPS using the conventional re-encoding technique
RPS-DA
RPS-$\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
MCMB+RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$
PSNR (dB)
Frame number
44.5
44
43.5
43
42.5
42
41.5
41
0
2
4
6
8
10
12
14
16

(f)



(g)

Figure 6.3. The PSNR performances of the conventional technique and the proposed MB-based techniques in the case when frame 2 is corrupted and the round-trip delay (RTD) corresponds to the duration of encoding time for three frames. (a) the “Mother and daughter” sequence encoded at 537 kbits/s, (b) the “Salesman” sequence encoded at 1.14 Mbits/s, (c) the “Football” sequence encoded at 2.25 Mbits/s, (d) the “Calendar” sequence encoded at 1.64 Mbits/s, (e) the “Table tennis” sequence encoded at 1.44 Mbits/s, (f) the “Foreman” sequence encoded at 1.12 Mbits/s, and (g) the “Carphone” sequence encoded at 457 kbits/s.

Table 6.3. Average PSNR performances for non-MC MBs of the succeeding frame after the corrupted frame by using different techniques in RPS.

Video Sequence	Bitrate (bits/s)	Desired Output			RPS using the re-encoding technique			RPS-DA (PSNR gain over conventional re-encoding technique)			$RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ / $MBMC+RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$ (PSNR gain over conventional re-encoding technique)		
		RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3
Mother and daughter	537k	45.41	45.41	45.43	45.18	45.20	45.25	45.37 (0.19)	45.34 (0.14)	45.32 (0.07)	45.41 (0.23)	45.41 (0.21)	45.41 (0.16)
	249k	43.56	43.56	43.59	43.35	43.38	43.49	43.54 (0.19)	43.53 (0.15)	43.52 (0.03)	43.56 (0.21)	43.56 (0.18)	43.58 (0.09)
	112k	40.97	41.00	41.00	40.87	40.95	40.97	40.87 (0.00)	40.99 (0.04)	40.98 (0.01)	40.97 (0.10)	41.00 (0.05)	41.00 (0.03)
Salesman	1.14M	43.76	43.76	43.75	43.21	43.27	43.07	43.41 (0.20)	43.49 (0.22)	43.07 (0.00)	43.74 (0.53)	43.65 (0.38)	43.59 (0.52)
	686k	40.77	40.77	40.75	40.49	40.48	40.42	40.64 (0.15)	40.65 (0.17)	40.45 (0.03)	40.77 (0.28)	40.77 (0.29)	40.72 (0.30)
	256k	37.50	37.51	37.50	37.39	37.39	37.33	37.47 (0.08)	37.46 (0.07)	37.41 (0.08)	37.50 (0.11)	37.50 (0.11)	37.48 (0.15)
Football	2.25M	43.54	43.52	43.47	42.09	42.02	41.74	42.50 (0.41)	41.98 (-0.04)	41.39 (-0.35)	43.43 (1.34)	43.31 (1.29)	43.15 (1.41)
	1.54M	39.90	39.89	39.84	39.04	38.92	38.81	39.39 (0.35)	39.05 (0.13)	38.68 (-0.13)	39.86 (0.82)	39.82 (0.90)	39.74 (0.93)
	898k	35.64	35.67	35.67	35.15	35.07	34.98	35.42 (0.27)	35.28 (0.21)	35.09 (0.11)	35.64 (0.49)	35.65 (0.58)	35.64 (0.66)
Calendar	1.64M	44.39	44.41	44.45	42.02	42.08	41.92	43.20 (1.18)	41.53 (-0.55)	39.84 (-2.08)	44.30 (2.28)	44.22 (2.14)	44.15 (2.22)
	995k	40.99	41.04	40.95	38.99	38.83	38.72	40.00 (1.01)	38.51 (-0.32)	37.03 (-1.69)	40.95 (1.96)	40.96 (2.13)	40.95 (2.23)
	828k	36.55	36.55	36.49	34.89	34.55	34.29	35.89 (1.00)	34.69 (0.14)	33.43 (-0.86)	36.54 (1.65)	36.51 (1.96)	36.44 (2.15)
Table tennis	1.44M	43.35	43.34	43.27	42.12	41.89	41.79	42.94 (0.82)	41.84 (-0.05)	41.69 (-0.10)	43.23 (1.11)	43.12 (1.23)	43.01 (1.22)
	934k	39.66	39.65	39.63	39.04	38.77	38.73	39.51 (0.47)	38.87 (0.10)	38.84 (0.11)	39.62 (0.58)	39.58 (0.81)	39.50 (0.77)
	504k	35.13	35.11	35.09	34.82	34.65	34.65	35.08 (0.26)	34.86 (0.21)	34.84 (0.19)	35.13 (0.31)	35.09 (0.44)	35.06 (0.41)
Foreman	1.12M	43.98	43.99	43.94	42.37	42.30	42.14	43.66 (1.29)	43.24 (0.94)	42.85 (0.71)	43.91 (1.54)	43.86 (1.56)	43.77 (1.63)
	712k	40.97	41.01	40.98	39.87	39.82	39.71	40.80 (0.93)	40.61 (0.79)	40.33 (0.62)	40.97 (1.10)	41.00 (1.18)	40.95 (1.24)
	354k	37.55	37.67	37.73	37.05	37.03	36.99	37.47 (0.42)	37.49 (0.46)	37.41 (0.42)	37.55 (0.50)	37.67 (0.64)	37.72 (0.73)
Carphone (qcif)	457k	44.41	44.73	45.75	43.33	43.13	44.90	44.12 (0.79)	43.85 (0.72)	45.54 (0.64)	44.38 (1.05)	44.63 (1.50)	45.65 (0.75)
	278k	41.58	42.10	43.61	40.70	40.77	42.84	41.39 (0.69)	41.50 (0.73)	43.49 (0.65)	41.58 (0.88)	42.01 (1.24)	43.54 (0.70)
	102k	38.12	38.48	40.29	37.47	37.65	39.69	38.04 (0.57)	38.13 (0.48)	40.23 (0.54)	38.12 (0.65)	38.47 (0.82)	40.24 (0.55)

Table 6.4. Average PSNR performances for all MBs of the succeeding frames after the corrupted frame by using different techniques in RPS.

Video Sequence	Bitrate (bits/s)	RPS using the re-encoding technique			RPS-DA			$RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$			$MCMB+RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$		
		RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3	RTD=1	RTD=2	RTD=3
Mother and daughter	537k	44.65	44.61	44.68	44.80	44.72	44.73	44.84	44.78	44.80	44.88	44.79	44.79
	249k	42.74	42.76	42.85	42.89	42.87	42.87	42.92	42.90	42.92	42.96	42.91	42.90
	112k	40.27	40.27	40.30	40.30	40.29	40.30	40.35	40.31	40.32	40.35	40.31	40.31
Salesman	1.14M	42.84	42.79	42.68	42.98	42.93	42.68	43.21	43.04	43.03	43.28	43.06	42.98
	686k	40.27	40.19	40.21	40.38	40.31	40.23	40.50	40.40	40.43	40.51	40.39	40.38
	256k	37.20	37.11	37.13	37.27	37.16	37.18	37.30	37.20	37.24	37.31	37.22	37.24
Football	2.25M	41.67	41.62	41.48	41.89	41.59	41.30	42.36	42.25	42.14	42.37	42.30	42.17
	1.54M	38.46	38.37	38.31	38.65	38.44	38.25	38.89	38.81	38.73	39.96	38.92	38.78
	898k	34.40	34.30	34.24	34.54	34.4	34.29	34.64	34.56	34.52	34.96	34.65	34.55
Calendar	1.64M	41.49	41.46	41.42	41.72	41.35	40.97	41.89	41.79	41.75	42.12	41.86	41.84
	995k	38.20	38.10	38.10	38.38	38.04	37.76	38.53	38.42	38.40	38.61	38.56	38.53
	828k	34.00	33.89	33.86	34.18	33.91	33.70	34.28	34.17	34.15	34.21	34.14	34.15
Table tennis	1.44M	42.01	41.78	41.69	42.68	41.75	41.62	42.91	42.77	42.63	42.98	42.81	42.67
	934k	38.98	38.72	38.66	39.37	38.81	38.75	39.46	39.38	39.26	39.48	39.42	39.23
	504k	34.80	34.62	34.60	35.02	34.79	34.75	35.06	34.99	34.93	35.15	34.99	34.76
Foreman	1.12M	41.93	41.84	41.83	42.63	42.27	42.17	42.75	42.54	42.56	42.92	42.58	42.53
	712k	39.53	39.40	39.40	40.05	39.76	39.68	40.15	39.92	39.94	40.22	39.95	39.97
	354k	36.66	36.49	36.50	36.89	36.69	36.67	36.94	36.76	36.79	36.94	36.76	36.77
Carphone (qcif)	457k	43.17	42.44	42.26	43.88	42.79	42.34	44.12	43.13	42.35	44.15	43.12	42.38
	278k	40.52	39.93	39.80	41.13	40.31	39.89	41.30	40.57	39.91	41.35	40.58	39.90
	102k	37.28	36.92	36.55	37.79	37.17	36.63	37.86	37.35	36.63	37.88	37.38	36.62

Table 6.5. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD=1).

Video Sequence	Bitrate (b/s)	Desired Output			RPS using the re-encoding technique			RPS-DA			RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$			$MCMB+RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$		
		non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All
Mother and daughter	537 k	45.41	44.07	45.23	45.18	41.99	44.65	45.37	41.99	44.80	45.41	41.99	44.84	45.41	42.02	44.85
	249 k	43.56	41.28	43.23	43.35	39.72	42.74	43.54	39.72	42.89	43.56	39.72	42.92	43.56	39.81	42.94
	112 k	40.97	38.11	40.52	40.87	37.28	40.27	40.86	37.28	40.30	40.97	37.28	40.35	40.97	37.29	40.35
Salesman	1.14 M	43.76	43.69	43.74	43.21	41.78	42.84	43.41	41.78	42.98	43.74	41.78	43.21	43.74	41.80	43.22
	686 k	40.77	40.75	40.77	40.49	39.49	40.27	40.64	39.49	40.38	40.77	39.49	40.50	40.77	39.51	40.50
	256 k	37.50	37.25	37.47	37.39	36.25	37.20	37.47	36.25	37.27	37.50	36.25	37.30	37.50	36.25	37.30
Football	2.25 M	43.54	44.06	43.73	42.09	41.08	41.67	42.50	41.08	41.89	43.43	41.08	42.36	43.43	41.18	42.40
	1.54 M	39.90	40.30	40.05	39.04	37.68	38.46	39.39	37.68	38.65	39.86	37.68	38.89	39.86	37.69	38.90
	898 k	35.64	35.76	35.67	35.15	33.45	34.40	35.42	33.45	34.54	35.64	33.45	34.64	35.64	33.46	34.64
Calendar	1.64 M	44.39	44.11	44.18	42.02	41.34	41.49	43.20	41.34	41.72	44.30	41.34	41.89	44.30	41.36	41.91
	995 k	40.99	40.50	40.61	38.99	37.97	38.20	40.00	37.97	38.38	40.95	37.97	38.53	40.95	38.05	38.59
	828 k	36.55	35.95	36.09	34.89	33.74	34.00	35.89	33.74	34.18	36.54	33.74	34.28	36.54	33.71	34.26
Table tennis	1.44 M	43.35	43.84	43.42	42.12	41.37	42.01	42.94	41.37	42.68	43.23	41.37	42.91	43.23	41.39	42.92
	934 k	39.66	40.72	39.80	39.04	38.60	38.98	39.51	38.60	39.37	39.62	38.60	39.46	39.62	38.60	39.46
	504 k	35.13	36.57	35.31	34.82	34.69	34.80	35.08	34.69	35.02	35.13	34.69	35.06	35.13	34.68	35.06
Foreman	1.12 M	43.98	43.70	43.87	42.37	41.28	41.93	43.66	41.28	42.63	43.91	41.28	42.75	43.91	41.41	42.81
	712 k	40.97	40.85	40.92	39.87	39.01	39.53	40.80	39.01	40.05	40.97	39.01	40.15	40.97	39.13	40.20
	354 k	37.55	37.46	37.51	37.05	36.12	36.66	37.47	36.12	36.89	37.55	36.12	36.94	37.55	36.15	36.95
Carphone (qcif)	457 k	44.41	43.83	44.37	43.33	41.28	43.17	44.12	41.28	43.88	44.38	41.28	44.12	44.38	41.30	44.13
	278 k	41.58	40.74	41.52	40.70	38.66	40.52	41.39	38.66	41.13	41.58	38.66	41.30	41.58	38.66	41.30
	102 k	38.12	36.18	38.00	37.47	34.68	37.28	38.04	34.68	37.79	38.12	34.68	37.86	38.12	34.66	37.85

Table 6.6. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD =2).

Video Sequence	Bitrate (b/s)	Desired Output			Conventional RPS using re-encoding			RPS-DA			RPS- $\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$			$MCMB+RPS-\tilde{Q}_{L/2}+\tilde{Q}_{L/2}^{-1}$		
		non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All
Mother and daughter	537 k	45.41	44.07	45.23	45.20	41.78	44.61	45.34	41.78	44.72	45.41	41.78	44.78	45.41	41.80	44.79
	249 k	43.56	41.28	43.23	43.38	39.70	42.76	43.53	39.70	42.87	43.56	39.70	42.90	43.56	39.78	42.92
	112 k	41.00	38.16	40.52	40.95	37.22	40.27	40.99	37.22	40.29	41.00	37.22	40.31	41.00	37.18	40.29
Salesman	1.14 M	43.76	43.69	43.74	43.27	41.58	42.79	43.49	41.58	42.93	43.65	41.58	43.04	43.65	41.60	43.06
	686 k	40.77	40.75	40.77	40.48	39.28	40.19	40.65	39.28	40.31	40.77	39.28	40.40	40.77	39.29	40.40
	256 k	37.51	37.23	37.47	37.39	35.89	37.11	37.46	35.89	37.16	37.50	35.89	37.20	37.50	35.89	37.20
Football	2.25 M	43.52	44.05	43.73	42.02	41.11	41.62	41.98	41.11	41.59	43.31	41.11	42.25	43.31	41.12	42.25
	1.54 M	39.89	40.30	40.05	38.92	37.69	38.37	39.05	37.69	38.44	39.82	37.69	38.81	39.82	37.71	38.82
	898 k	35.67	35.72	35.67	35.07	33.47	34.30	35.28	33.47	34.4	35.65	33.47	34.56	35.65	33.46	34.56
Calendar	1.64 M	44.41	44.12	44.18	42.08	41.30	41.46	41.53	41.30	41.35	44.22	41.30	41.79	44.22	41.30	41.79
	995 k	41.04	40.51	40.61	38.83	37.93	38.10	38.51	37.93	38.04	40.96	37.93	38.42	40.96	37.94	38.42
	828 k	36.55	35.98	36.09	34.55	33.74	33.89	34.69	33.74	33.91	36.51	33.74	34.17	36.51	33.78	34.18
Table tennis	1.44 M	43.34	43.83	43.42	41.89	41.29	41.78	41.84	41.29	41.75	43.12	41.29	42.77	43.12	41.30	42.77
	934 k	39.65	40.68	39.80	38.77	38.49	38.72	38.87	38.49	38.81	39.58	38.49	39.38	39.58	38.50	39.39
	504 k	35.11	36.51	35.31	34.65	34.48	34.62	34.86	34.48	34.79	35.09	34.48	34.99	35.09	34.48	34.99
Foreman	1.12 M	43.99	43.74	43.87	42.30	41.35	41.84	43.24	41.35	42.27	43.86	41.35	42.54	43.86	41.40	42.57
	712 k	41.01	40.83	40.92	39.82	39.00	39.40	40.61	39.00	39.76	41.00	39.00	39.92	41.00	39.00	39.92
	354 k	37.67	37.36	37.51	37.03	36.02	36.49	37.49	36.02	36.69	37.67	36.02	36.76	37.67	36.01	36.76
Carphone (qcif)	457 k	44.73	43.90	44.37	43.13	41.62	42.44	43.85	41.62	42.79	44.63	41.62	43.13	44.63	41.61	43.12
	278 k	42.10	40.62	41.52	40.77	38.73	39.93	41.50	38.73	40.31	42.01	38.73	40.57	42.01	38.73	40.57
	102 k	38.48	37.24	38.00	37.65	35.83	36.92	38.13	35.83	37.17	38.47	35.83	37.35	38.47	35.81	37.34

Table 6.7. PSNR performances of the proposed technique and the conventional re-encoding technique after the corrupted frame (for RTD =3).

Video Sequence	Bitrate (b/s)	Desired Output			Conventional RPS using re-encoding			RPS-DA			RPS- $\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$			$MCMB + RPS - \tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$		
		non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All	non-MC	MC	All
Mother and daughter	537 k	45.43	44.08	45.23	45.25	42.06	44.68	45.32	42.06	44.73	45.41	42.06	44.80	45.41	41.97	44.75
	249 k	43.59	41.33	43.23	43.49	39.92	42.85	43.52	39.92	42.87	43.58	39.92	42.92	43.58	39.94	42.93
	112 k	41.00	38.16	40.52	40.97	37.28	40.30	40.98	37.28	40.30	41.00	37.28	40.32	41.00	37.30	40.33
Salesman	1.14 M	43.75	43.71	43.74	43.07	41.70	42.68	43.07	41.70	42.68	43.59	41.70	43.03	43.59	41.64	43.00
	686 k	40.75	40.84	40.77	40.42	39.56	40.21	40.45	39.56	40.23	40.72	39.56	40.43	40.72	39.55	40.43
	256 k	37.50	37.31	37.46	37.33	36.23	37.13	37.41	36.23	37.18	37.48	36.23	37.24	37.48	36.20	37.22
Football	2.25 M	43.47	44.06	43.73	41.74	41.20	41.48	41.39	41.20	41.30	43.15	41.20	42.14	43.15	41.10	42.10
	1.54 M	39.84	40.32	40.05	38.81	37.78	38.31	38.68	37.78	38.25	39.74	37.78	38.73	39.74	37.70	38.69
	898 k	35.67	35.72	35.69	34.98	33.51	34.24	35.09	33.51	34.29	35.64	33.51	34.52	35.64	33.50	34.52
Calendar	1.64 M	44.45	44.12	44.18	41.92	41.31	41.42	39.84	41.31	40.97	44.15	41.31	41.75	44.15	41.33	41.76
	995 k	40.95	37.96	38.40	38.72	37.96	38.10	37.03	37.96	37.76	40.95	37.96	38.40	40.95	37.96	38.40
	828 k	36.49	36.00	36.09	34.29	33.77	33.86	33.43	33.77	33.70	36.44	33.77	34.15	36.44	33.78	34.15
Table tennis	1.44 M	43.27	43.74	43.33	41.79	41.31	41.69	41.69	41.31	41.62	43.01	41.31	42.63	43.01	41.31	42.63
	934 k	39.63	40.61	39.80	38.73	38.38	38.66	38.84	38.38	38.75	39.50	38.38	39.26	39.50	38.39	39.26
	504 k	35.09	36.39	35.31	34.65	34.41	34.60	34.84	34.41	34.75	35.06	34.41	34.93	35.06	34.40	34.93
Foreman	1.12 M	43.94	43.80	43.87	42.14	41.51	41.83	42.85	41.51	42.17	43.77	41.51	42.56	43.77	41.45	42.54
	712 k	40.98	40.87	40.92	39.71	39.10	39.40	40.33	39.10	39.68	40.95	39.10	39.94	40.95	39.12	39.95
	354 k	37.73	37.33	37.51	36.99	36.10	36.50	37.41	36.10	36.67	37.72	36.10	36.79	37.72	36.10	36.79
Carphone (qcif)	457 k	45.75	44.01	44.37	44.90	41.67	42.26	45.54	41.67	42.34	45.65	41.67	42.35	45.65	41.67	42.35
	278 k	43.61	40.83	41.52	42.84	38.90	39.80	43.49	38.90	39.89	43.54	38.90	39.91	43.54	38.92	39.91
	102 k	40.29	37.19	38.00	39.69	35.55	36.55	40.23	35.55	36.63	40.24	35.55	36.63	40.24	35.50	36.61

Chapter 7

Conclusions and Future Directions

7.1 Conclusions of the present work

In this thesis, we give results for an investigation on compressed-domain techniques for adopting RPS in transmitting an already MPEG encoded bitstream over lossy networks. These compressed-domain techniques are really conducive to the implementation of RPS in the pre-encoded bitstream with the minimum requirement on server complexity and the reduced quality degradation of the reconstructed video.

Although RPS is one of the well-known error-resilient schemes for robust video transmission over noisy channels, it has not been successfully implemented in the video transmission system with pre-encoded video bitstreams. However, the forthcoming video services and applications are also expected to use pre-encoded bitstreams for storage and transmission. A straightforward implementation in this scenario has been presented in Chapter 3. Results of our study also indicated that this pixel-domain implementation is still primitive, and this introduces high processing complexity of the server and quality degradation of the reconstructed frames. A further need for using RPS in already MPEG encoded bitstreams may arise.

Therefore, in this research, the investigation on the use of compressed-domain processing to modify the already encoded bitstream for the purpose of alteration its reference frame prior to transmission over a noisy channel has been examined. In Chapters 4 and 5, various compressed-domain solutions have been proposed to different types of MBs. They give a new direction for the implementation of RPS in the already MPEG encoded bitstream. In previous studies, the investigation of the straightforward implementation has always been at the frame level. In fact, different MBs in one particular frame have different properties in the compressed domain. A MB-level scheme has thus been proposed in this thesis for adopting RPS in the pre-encoded bitstream. The proposed video streaming server divides MBs into two types. They are non-MC MBs and MC MBs. After the classification of MBs with our proposed server, they are processed by using the novel compressed-domain techniques and then transmitted to the receiver.

In Chapter 4, we have proposed two techniques for handling non-MC MBs. The first one is direct addition of quantized DCT coefficients. This technique, which depends on the linear properties of DCT and IDCT, is able to reuse the existing quantized DCT coefficients to generate the new prediction errors in the compressed domain. The derivation in Chapter 4 has proven that complete decoding and encoding are not required. As a result, substantial computational savings can be achieved in the video server. However, due to the non-linear property of the practical quantization and inverse quantization processes adopted in the MPEG standard, the direct addition technique cannot guarantee a good reconstructed quality for a non MC MB re-encoded with a new reference. That is, it needs further development in order to maintain the reconstructed

quality when the RPS scheme is applied to the pre-encoded bitstream. To solve the non-linear problem of the direct addition technique, a modified quantizer-dequantizer pair for non-MC MBs has been designed to avoid the quality degradation of the reconstructed MBs. This improved technique performs all computations in the quantized DCT domain. It has been found that the proposed quantizer-dequantizer pair really achieves a noticeable and worthwhile improvement over the conventional straightforward implementation in both server complexity and quality reconstruction. Since the re-encoding process can be prevented, the visual quality of non-MC MBs with a new reference will be nearly the same as that of the original sequence in the server.

The content in Chapter 5 is another contribution of the thesis. First, it has been shown that the compressed-domain techniques adopted in non-MC MBs cannot be employed directly in MC MBs. Second, it has described several DCT-domain operators such as shifting and chopping for the purpose of further enhancing our proposed RPS scheme. We have derived various DCT-domain operators for computing the new DCT coefficients of MC MBs. These operators have also been suggested working in conjunction with forward domain vector selection (FDVS), which can get the new motion vectors of MC MBs without doing full-scale full search motion estimation. Besides, the FDVS method can ensure to compute the new DCT coefficients as much as possible in the DCT form in order to keep the benefits of our proposed DCT-domain shifting and chopping operators. As a result, further savings in computation can be achieved by using these DCT-domain operators.

To examine the performances of our proposed compressed-domain techniques in the RPS scheme for transmitting already MPEG encoded bitstreams through error prone networks, we built up a simulation platform based on the MPEG-4 VM [22]. Various software simulations have been performed to evaluate the overall efficiency of our proposed techniques including direct addition of quantized DCT coefficients (RPS-DA), the modified quantizer-dequantizer pair (RPS-$\tilde{Q}_{L/2} - \tilde{Q}_{L/2}^{-1}$) and the DCT-domain shifting and chopping operators of DCT coefficients for MC MBs (MCMB+ RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$). As compared with the conventional re-encoding approach, detailed comparisons and discussions have been done in Chapter 6. To have the wide spectrum of our evaluations, an MPEG-4 VM [22] encoder was employed to pre-encode various video sequences with different spatial resolutions, motion characteristics and bitrates. Experimental works have shown that all of our proposed techniques outperform the conventional re-encoding approach in term of both server complexity and reconstructed video quality. Specifically, due to the centre-biased property of motion in natural video, the performances of RPS-DA and RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ have shown that these techniques are capable of giving about 80% computational savings of the server complexity over the conventional re-encoding approach. Besides, RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ can avoid the quality degradation of the reconstructed video. To further reduce the server complexity, the DCT-domain shifting and chopping operators can work with RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ for MCMBs, MCMB+RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$. Results of the realization have shown that MCMB+RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$ produces further savings, about 10%-35%, as compared with that of RPS-$\tilde{Q}_{L/2} + \tilde{Q}_{L/2}^{-1}$.

In conclusion, we have explored the freedom of performing RPS in the compressed domain. Specifically, some novel compressed-domain operators have been derived for re-encoding MBs with a new reference. Actually, these operators can be applied to general orthogonal transforms. Compared to the spatial-domain approach which converts compressed video back to the spatial domain, the proposed techniques can increase the efficiency of the server, with a factor depending on the characteristics of the input video. We sincerely believe that the results obtained in this work are significant to an efficient realization of the modern video streaming system over noisy channels.

7.2 Future Directions

In this thesis, we have designed a practical video streaming system where video contents are compressed and stored for future delivery. Transmitting the pre-encoded video over lossy networks is an active research topic. For adopting RPS in our system, the challenge is how to use the information from the pre-encoded video bitstream to reduce the complexity of the video server when a new reference frame is required for the current transmitted frame. With the successful techniques described above 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 The Proposed RPS Scheme with Varying Quantization Factors.

One constraint of our proposed RPS scheme is that we assume the quantization factor is kept constant throughout the pre-encoded video bitstream. In this bitstream, the number of bits in each coded frame is likely to vary considerably. This can cause significant problems for transmission since some practical networks cannot cope with unconstrained variation in bit rate. Therefore, in some existing video applications, rate control techniques are necessary for controlling this rate variation to enable effective transmission and storage. A common rate control algorithm is involved in adjusting the quantization factor for each video frame or slice to fit a target bit rate. Hence, transmitting the pre-encoded video with varying quantization factors is another issue for nowadays video applications.

In one of our proposed techniques, in order to keep the desired output quality, a tailor-made quantization-dequantization pair has been designed so that it can represent all the possible coefficients of the newly transmitted frame. The derivation in Chapter 4 is only valid for a fixed quantization factor. Thus, the problem of determining an optimal quantization-dequantization pair for varying quantization factors is very challenging. One possible suggestion is to find a common quantization domain that can fit all the possible dequantized values of various quantization factors used in the pre-encoded video bitstream. To illustrate the idea more efficiently, let us use the following simple quantization and dequantization pair as an example.

$$Q_s(x) = \left\lceil \frac{x}{L} \right\rceil \tag{7.1}$$

$$Q_s^{-1}(x) = x.L \tag{7.2}$$

Suppose two frames, frame *n* and *n-1,* in the compressed bitstream encoded with quantization factors 6 and 9 respectively. The idea is to select a new quantization factor so that all the possible coefficients of the new frame can be represented. It is found that the highest common factor (HCF) of the existing quantization factors can be chosen. Therefore, quantization factor 3 is likely to be selected in this case. This can be illustrated by analyzing their representation levels of dequantization in Figures 7.1 to 7.3.

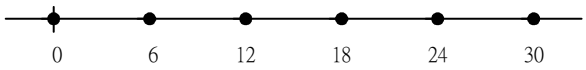


Figure 7.1. Representation level of quantization Q_s^{-1} with quantization factor = 6.

Figure 7.2. Representation level of quantization Q_s^{-1} with quantization factor = 9.

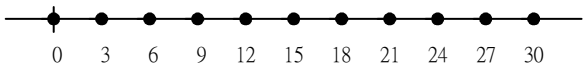


Figure 7.3. Representation level of quantization Q_s^{-1} with quantization factor = 3.

Since all the possible representation levels from both of the dequantizers and all the combination of their sums are able to represent by the dequantization equation when the quantization factor is equal to 3, RPS can then be carried out perfectly without degradation in non-MC MBs.

7.2.2 RPS in already H.264 encoded bitstreams

Recently, the Motion Picture Experts Group and the Video Coding Experts Group (MPEG and VCEG) have jointly established an advanced video coding standard - H.264/AVC. The H.264/AVC standard achieves much higher coding efficiency than the previous video coding standards such as MPEG4 and H.263. The new standard supports a collection of new encoding techniques including integer transform, new arrangement of quantization and dequantization pair, multiple reference frames for motion prediction, variable block size, etc. The integer transform is based on DCT but has an advance property. It is a transformation such that all operations are carried out using only integer arithmetic, without loss of decoding accuracy. This property can ensure no rounding drift error in both forward and backward transform, which happens in the typical DCT used in the previous MPEG and H.26x standards. For motion compensation using multiple reference frames, an H.264 encoder may use more than one previously encoded pictures as a reference for motion-compensated prediction.

These new features supported in H.264 provide both good and bad effects on performing RPS in the already H.264 encoded bitstream. For instance, the adoption of integer transform, quantization and dequantization can facilitate the employment of RPS in H.264 encoded video. It is because the forward and backward integer transform operations are drift free, so that $I(x_1)+I(x_2)$ is exactly equal to $I(x_1+x_2)$, where $I(\ )$ indicates the integer transform operator. In additional, the dequantization process used in H.264 standard is linear. Consequently, unlike the drift problem as shown in (4.13), the equation $Q_{H.264}(I(e_1+e_2))=Q_{H.264}(I(e_1))+Q_{H.264}(I(e_2))$ becomes valid and hence the RPS

operation never induces error, where $Q_{H.264}(\ )$ is the quantization operation in the H.264 standard. On the other hand, H.264 allows using motion estimation and compensation with multiple reference frames severely complicates the RPS process. This process will generate a complicated frame relationship in the video bitstream. In this case, reference alteration of one particular P-frame becomes more challenging. This is crucial issue that should be investigated before they can be put into practical for adopting RPS in the pre-encoded bitstream compressed by the newest H.264 standard.

Reference

[1] ISO/IEC 11172-2:1993 Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 2: Video.

[2] " Information technology—Generic coding of moving pictures and associated audio information—Part 2: Video," ISO/IEC DIS 13818-2, 1994.

[3] Video codec for audiovisual services at p *x* 64kbits, "International Telecommunications Union, Geneva, Switzerland, Itu-T Recommendation H.261," 1993.

[4] Video Coding for Low Bitrate Communication, ITU-T Recommendation H.263, May 1998.

[5] Iain E. G. Richardson, "H.264 and MPEG-4 Video Compression Video Coding for Next-generation Multimedia", Wiley, 2003.

[6] E. Steinbach, N. Farber, and B. Girod, "Standard Compatible Extension of H.263 for Robust Video Transmission in Mobile Environments," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 7, No. 6, pp.872 -881, December 1997, U.S.A.

[7] B. Girod and N. Farber. "Feedback-based error control for mobile video transmission," In Proc. IEEE, Vol. 87, pp.1707 – 1723, Oct. 1999.

[8] S. Fukunaga, T. Nakai, and H. Inoue, "Error resilient video coding by dynamic replacing of reference pictures," in Proc. IEEE Global Telecommunications Conf., Vol. 3, pp. 1503–1508, Nov. 1996.

[9] Y. Tomita, T. Kimura, and T. Ichikawa, "Error resilient modified inter-frame coding system for limited reference picture memories," in Proc. Int. Picture Coding Symposium, pp. 743–748, Sept. 1997.

[10] ISO/IEC 14496-2, “Information technology – coding of audio-visual objects: visual,” 1998.

[11] Barry G. Haskell, Atul Puri and Arun N. Netravali, “Digital Video: an Introduction to MPEG-2”, International Thomson Publishing, 1997.

[12] Yao Wang and Qin-Fan Zhu, “Error Control and Concealment for Video Communication: A Review,” Proceeding of IEEE, Vol. 86, No. 5, pp. 974-997, May 1998.

[13] K. N. Ngan and R. Stelle, “Enhancement of PCM and DPCM images corrupted by transmission errors,” IEEE Trans. Commun., vol. COM-30, pp. 257-265, Jan. 1982.

[14] K. M. Rose and A. Heiman, “ Enhancement of one-dimensional variable-length DPCM images corrupted by transmission errors,” IEEE Trans. Commun., vol. 37, pp. 373-379, Apr. 1989.

[15] Lin. S., D. J. Costello, Error control coding: fundamentals and applications. Englewood Cliffs, NJ: Prentice Hall, 1983.

[16] Q. F. Zhu, “Device and method of signal loss recovery for real-time and/or interactive communications,” U.S. Patent 5 550 847, Aug. 1996.

[17] M. Ghanbari, “Cell-loss concealment in ATM video codes,” IEEE Trans. Circuits System Video Technology, vol. 3, pp. 238–247, June 1993.

[18] R. Aravind, M. R. Civanlar, and A. R. Reibman, “Packet loss resilience of MPEG-2 scalable video coding algorithms,” IEEE Transaction Circuits System Video Technology., vol. 6, pp. 426–435, Oct.1996.

[19] A. Narula and J. S. Lim, “Error concealment techniques for an all-digital high-definition television system,” in Proc. SPIE Conf. Visual Communication Image Processing, Cambridge, MA, 1993, pp. 304–315. A–Annex C, 1998.

[20] P. Haskell and D. Messerschmitt, “Resynchronization of motion compensated video affected by ATM cell loss,” in Proc. ICASSP’92, San Francisco, CA, pp. III545–548, Mar. 1992.

[21] W. M. Lam, A. R. Reibman, B. Liu, “Recovery of lost or erroneously received motion vectors,” Acoustics, Speech, and Signal Processing, ICASSP-93, IEEE International Conference on, vol 5, pp.417 - 420, 27-30 April 1993.

[22] Q.-F. Zhu, Y. Wang, and L. Shaw, “Coding and cell loss recovery for DCT-based packet video,” IEEE Trans. Circuits System. Video Technology., vol. 3, pp. 248–258, June 1993.

[23] W. Zhu and Y. Wang, “A comparison of smoothness measures for error concealment in transform coding,” in Proc. SPIE Conference Visual Communication and Image Processing, Taipei, Taiwan, 1995, vol. II, pp. 1205–1214.

[24] W. Kwok and H Sum, “Multi-directional interpolation for spatial error concealment,” IEEE Transaction Consumer Electronic, vol. 39, pp. 455-460, August 1993.

[25] T. H. Tsai, Y. X. Lee, “The hybrid video error concealment algorithm with low complexity approach,” Information, Proceedings of the 2003 Joint Conference of the Fourth International Conference, vol 1, pp. 268 – 271, Dec. 2003.

[26] T. Sikora, “The MPEG-4 Video Standard Verification Model,” IEEE Transactions on Circuits and Systems for Video Technology,” Vol. 7, No.1, pp.19-31, Feb. 1997.

[27] S.-M. Lei and M.-T Sun, “An entropy coding system for digital HDTV applications,” IEEE Trans. Circuits Syst. Video Technol., vol. 1, pp. 147–154, Mar. 1991.

[28] W.-M. Lam and A. R. Reibman, “Self-synchronizing variablelength codecs for image transmission,” in Proc. ICASSP’92, San Francisco, CA, Mar. 1992, pp. III477–480.

[29] A. H. Sadka, F. Eryurthlu, A. M. Kondoz, “Error-resilience improvement for block-transform video coders,” Vision, Image and Signal Processing, IEE Proceedings-vol 144, Issue 6, pp. :369 – 376, Dec. 1997

[30] S. S. Hemami and R. M. Gray, “Image reconstruction using vector quantized linear interpolation,” in Proc. ICASSP’94, Australia, May 1994, pp. V-629–632.

[31] Y. Yu, X. Zhu, “Error resilience of multireference motion compensated prediction in video coding,” Vision, Image and Signal Processing, IEE Proceedings, vol 149, Issue 6, pp. 355 – 364, Dec. 2002.

[32] C. S. Kim, R. C. Kim, S. U. Lee, “Robust transmission of video sequence using double-vector motion compensation,” Circuits and Systems for Video Technology, IEEE Transactions on, vol 11, Issue 9, pp. 1011 – 1021, Sept. 2001.

[33] D. L. Robie, R. M. Mersereau, “Video error correction using steganography,” Image Processing, 2001. Proceedings. 2001 International Conference on, vol 1, pp. 930 – 933, Oct. 2001.

[34] M. Kansari *et al*., “Low bit rate video transmission over fading channels for wireless microcellular systems,” IEEE Transaction Circuits System Video Technology, vol. 6, pp. 1–11, Feb. 1996.

[35] A. S. Tom, C. L. Yeh, and F. Chu, “Packet video for cell loss protection using deinterleaving and scrambling,” in Proc. ICASSP’91, Toronto, Canada, Apr. 1991, pp. 2857–2850.

[36] Y.Wang and D. Chung, “Non-hierarchical signal decomposition and maximally smooth reconstruction for wireless video transmission,”in Mobile

Multimedia Communications, D. Goodman and D. Raychaudhuri, Eds. New York: Plenum, to be published.

[37] D. Wang, N. Canagarajah, D. Bull, Circuits and Systems, ISCAS 2005. IEEE International Symposium, vol. 2, pp. 960 - 963 May 2005.

[38] V. A. Vaishampayan and J. Domaszewicz, “Design of entropy constrained multiple description scalar quantizers,” IEEE Trans. Inform. Theory, vol. 40, pp. 245–251, Jan. 1994.

[39] ITU-T. Recommendation H.245: Control Protocol for Multimedia Communication, 1998.

[40] B. S. Choi, S. Il Chae, and J. B. Ra, “A Feedback Channel Based Error Compensation Method for Mobile Video Communications Using H.263 Standard,” Image Processing, ICIP 99. Proceedings. International Conference on, vol 2, pp. 555 – 559, 1999.

[41] M. R. Hueda, “New Error Resilience Method for Video Transmission in DS-CDMA Systems,” Wireless Personal Multimedia Communications, 2002. The 5th International Symposium on, vol 3, pp. 1098 – 1102, Oct 2002

[42] W. Wada, “Selective recovery of video packet loss using error concealment,” IEEE J. Select. Areas Commun., vol. 7, pp. 807–814, June 1989.

[43] H. Yasuda, H. Kuroda, H. Kawanishi, F. Kanaya, and H. Hashimoto, “Transmitting 4 MHz TV signals by combinational difference coding,” IEEE Trans. Commun., vol. COM-25, pp. 508–516, May 1977.

[44] D. W. Redmill, D. R. Bull, “Resilient video compression using absolute value coding,” Image Processing and Its Applications, Seventh International Conference on (Conf. Publ. No. 465) vol 2, pp. 591 – 595, July 1999

[45] K. H. Yang, D. W. Kang, and A. F. Faryar, “Efficient Intra Refreshment and Synchronization Algorithms for Robust Transmission of Video over Wireless

Networks," Image Processing, Proceedings. 2001 International Conference on , vol 1 , pp.7-10 Oct. 2001.

[46] Fukunaga, S, Nakai, T., Inoue, H, "Error resilient video coding by dynamic replacing of reference pictures", GLOBECOM 96, vol. 3, pp. 1503-1508, Nov 1996.

[47] Yasuhiro Tmoita, Tsukasa Kimura, and Tadashi Ichikawa, "Error Resilient Modified inter-frame coding system for Limited Reference Picture Memories" Proc. Of PCS'97, pp. 743-748, Sep. 1997.

[48] H. Kimata, Y. Tomita, H. Yamaguchi, S. Ichinose, "Study on adaptive reference picture selection coding scheme for the NEWPRED-receiver-oriented mobile visual communication system", IEEE on Global Telecommunications Conference, vol. 3, pp. 1431-1436, Nov 1998.

[49] Y. J. Liang, M. Flierl and B. Girod, "Low-Latency Video Transmission over Lossy Packet Networks Using Rate-distortion Optimized Reference Picture Selection," Image Processing, Proceedings, 2002 International Conference on vol 2, pp. II-181 - II-184, Sept. 2002.

[50] S. Lin, S. Mao, Y. Wang and S. Panwar, "A Reference Picture Selection Scheme for video Transmission over Ad-Hoc Networks using Multiple Paths," Multimedia and Expo, ICME 2001. IEEE International Conference on, pp. 96 – 99, Aug. 2001

[51] Y. J. Liang, E. Setton, B. Girod, "Channel-adaptive video streaming using packet path diversity and rate-distortion optimized reference picture selection," Multimedia Signal Processing, 2002 IEEE Workshop on, pp. 420 – 423, Dec. 2002

[52] M. Ghanbari, "Postprocessing of late cells for packet video,"IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp. 669–678, Dec. 1996.

[53] I. Rhee, S. R. Joshi, “Error recovery for interactive video transmission over the Internet,” Selected Areas in Communications, IEEE Journal on, vol 18, Issue 6, pp. 1033 – 1049, June 2000.

[54] Gustavo de los Reyes, Amy R. Reibman, Shih-Fu Chuag, and Justin C. –I. Chuang, “Error-Resilient Transcoding for Video over Wireless Channels,” IEEE Journal on Selected Areas in Communications, vol. 18, no. 6, pp. 1063-1074, June 2000.

[55] Safak Dogan, Akin Cellatoglu, Mustafa Uyguroglu, Abdul H. Sadka, and Ahmet M. Kondoz, “Error-Resilient Video Transcoding for Robust Internetwork Communications Using GPRS,” IEEE Transactions on Circuits and Systems for Video Technology, vol.12, no. 6, pp. 453-464, June 2002.

[56] Jeongnam Youn, Ming-Ting Sun and Chia-Wen Lin, “Motion vector refinement for high-performance transcoding,” IEEE Trans. Multimedia, vol. 1, pp. 30-40, March 1999.

[57] Jeongnam Youn, Ming-Ting Sun and Chia-Wen Lin, “Motion estimation for high performance transcoding,” IEEE Trans. Consumer Electron., vol. 44, pp. 649-658, Aug. 1998.

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

[59] IEEE Standard Specifications for the Implementation of 8×8 Inverse Discrete Cosine Transform, IEEE Std. 1180–1190.