Real-time Interactive Video Streaming over Lossy Networks: High Performance Low Delay Error Resilient Algorithms

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy
by

Jimin Xiao

Department of Electrical Engineering and Electronics
School of Electrical Engineering and Electronics and Computer Science
University of Liverpool

Nov. 18, 2013

Abstract

According to Cisco's latest forecast, two-thirds of the world's mobile data traffic and 62 percent of the consumer Internet traffic will be video data by the end of 2016. However, the wireless networks and Internet are unreliable, where the video traffic may undergo packet loss and delay. Thus robust video streaming over unreliable networks, i.e., Internet, wireless networks, is of great importance in facing this challenge. Specifically, for the real-time interactive video streaming applications, such as video conference and video telephony, the allowed end-to-end delay is limited, which makes the robust video streaming an even more difficult task. In this thesis, we are going to investigate robust video streaming for real-time interactive applications, where the tolerated end-to-end delay is limited.

Intra macroblock refreshment is an effective tool to stop error propagations in the prediction loop of video decoder, whereas redundant coding is a commonly used method to prevent error from happening for video transmission over lossy networks. In this thesis two schemes that jointly use intra macroblock refreshment and redundant coding are proposed. In these schemes, in addition to intra coding, we proposed to add two redundant coding methods to enhance the transmission robustness of the coded bitstreams. The selection of error resilient coding tools, i.e., intra coding and/or redundant coding, and the parameters for redundant coding are determined using the end-to-end rate-distortion optimization.

Another category of methods to provide error resilient capacity is using forward error correction (FEC) codes. FEC is widely studied to protect streamed video over unreliable networks, with Reed-Solomon (RS) erasure codes as its commonly used implementation method. As a block-based error correcting code, on the one hand, enlarging the block size can enhance the performance of the RS codes; on the other hand, large block size leads to long delay which is not tolerable for real-time video applications. In this thesis two sub-GOP (Group of Pictures, formed by I-frame and all the

following P/B-frames) based FEC schemes are proposed to improve the performance of Reed-Solomon codes for real-time interactive video applications. The first one, named DSGF (Dynamic sub-GOP FEC Coding), is designed for the ideal case, where no transmission network delay is taken into consideration. The second one, named RVS-LE (Real-time Video Streaming scheme exploiting the Late- and Early-arrival packets), is more practical, where the video transmission network delay is considered, and the late- and early-arrival packets are fully exploited. Of the two approaches, the sub-GOP, which contains more than one video frame, is dynamically tuned and used as the RS coding block to get the optimal performance.

For the proposed DSGF approach, although the overall error resilient performance is higher than the conventional FEC schemes, that protect the streamed video frame by frame, its video quality fluctuates within the Sub-GOP. To mitigate this problem, in this thesis, another real-time video streaming scheme using randomized expanding Reed-Solomon code is proposed. In this scheme, the Reed-Solomon coding block includes not only the video packets of the current frame, but also all the video packets of previous frames in the current group of pictures (GOP). At the decoding side, the parity-check equations of the current frame are jointly solved with all the parity-check equations of the previous frames. Since video packets of the following frames are not encompassed in the RS coding block, no delay will be caused for waiting for the video or parity packets of the following frames both at encoding and decoding sides.

The main contribution of this thesis is investigating the trade-off between the video transmission delay caused by FEC encoding/decoding dependency, the FEC error-resilient performance, and the computational complexity. By leveraging the methods proposed in this thesis, proper error-resilient tools and system parameters could be selected based on the video sequence characteristics, the application requirements, and the available channel bandwidth and computational resources. For example, for the applications that can tolerate relatively long delay, sub-GOP based approach is a suitable solution. For the applications where the end-to-end delay is stringent and the computational resource is sufficient (e.g. CPU is fast), it could be a wise choice to use the randomized expanding Reed-Solomon code.

Contents

List of Tables

List of Figures

List of Abbreviations

ARQ   Automatic Repeat-reQuest

CABAC   Context-Adaptive Binary Arithmetic Coding

CDF   Cumulative Distribution Function

CIF   Common Intermediate Format

CPU   Central Processing Unit

FEC   Forward Error Correction

FMO   Flexible Macroblock Ordering

fps   frame per second

GF   Galois Field

GOP   Group of Pictures

HRMIR   Hybrid Redundant Macroblock and Intra-macroblock Refreshment

i.i.d.   Independent Identically Distributed

JRVIR   Joint Redundant motion Vector and Intra-macroblock Refreshment

LT   Luby Transform

MB   Macroblock

MDC   Multiple Description Coding

ME   Motion Estimation

MSE   Mean Squared Error

MTU   Maximum Transmission Unit

NAL   Network Abstraction Layer

PLR	Packet Loss Rate
PSNR	Peak Signal-to-Noise Ratio
RDO	Rate-Distortion Optimization
RE-RS	Randomized Expanding Reed-Solomon
RMV	Redundant Motion Vector
ROPE	Recursive Optimal Per-pixel Estimate
RPS	Reference Picture Selection
RS	Reed-Solomon Code
RS-MDC	Redundant Slice Multiple Description Coding
RTCP	Real Time Control Protocol
RTP	Real Time Protocol
RTT	Round-Trip Time
RVS-LE	Real-time Video Streaming scheme exploiting the Late- and Early-arrival packets
SVC	Scalable Video Coding
VCL	Video Coding Layer
ULP	Unequal Loss Protection

Acknowledgement

Foremost, I would like to express my sincere gratitude to my primary PhD supervisor professor Tammam TILLO for his continuous support of my PhD study and research, for his patience, guidance, encouragement, kindness, and immense knowledge. His guidance helped me in all the time of my research and writing of this thesis. I could not have imagined having a better supervisor for my PhD study. Besides my primary supervisor, I would like to thank my co-supervisors, professor Jeremy S Smith and Dr. Waleed Al-Nuaimy.

My sincere thanks also go to professor Weisi LIN, in Nanyang Technological University, Singapore, for offering me the research assistant position in his research group and leading me working on diverse exciting research fields.

I thank our research partners in Beijing Jiaotong University, including professor Yao ZHAO, Dr. Chunyu LIN, and Chao YAO. I also thank my fellow PhD students in the department, Chun ZHAO, Shi CHENG, Jie REN, for starting this journey together.

Last but not least, I would like to thank my family, my wife Qing, her understanding and love during the past few years. Her support and encouragement was in the end what made this dissertation possible.

Chapter 1

Introduction to Error Resilient Video Streaming

1.1 Motivation

A video communication system typically includes five stages, as shown in Fig.1.1. The video is firstly compressed by the video encoder so as to reduce the bit rate, and then the bit stream is segmented into fixed or variable length packets. If the deployed communication network is unreliable, transmitting the hybrid-coded video over such unreliable environments would make it suffer from error propagations and this leads to the well-known drifting phenomenon [3, 4]. Therefore, the video packets usually undergo a channel coding stage, where typically the Forward Error Correction (FEC) protection packets are used to protect them. At the receiver side, the received packets are FEC decoded to help recovering the lost source packets. After that, the source packets are video decoded, and displayed. For these still unrecovered source packets, error concealment is widely used to "guesss" the lost regions.

For the real-time interactive video streaming applications, i.e., video conference and video telephony, the tolerated end-to-end delay is limited, typically, the acceptable delay is between 150 ms and 400 ms according to ITU-T G.114 [5]. This stringent delay constraint poses challenges to the video communication systems, since the total delay
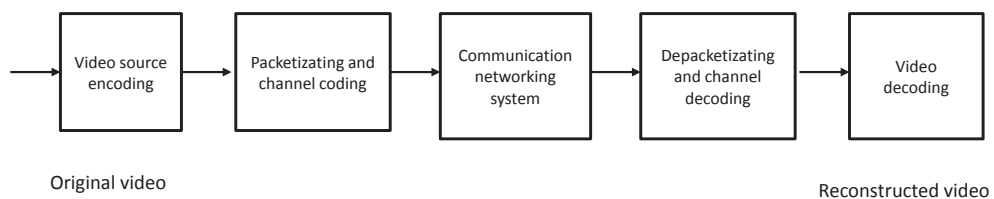
Figure 1.1: A typical video communication system for unreliable networks.

caused in each stage cannot be more than this limitation. In this thesis, we are going to design error resilient video communication systems with stringent end-to-end delay for real-time interactive video applications. To fulfill this objective, the research work will be carried out from both source coding and channel coding aspects. Firstly, we are going to investigate error resilient techniques that are implemented in the standard video compression stage. For example, adding more intra coding regions can efficiently stop error propagations; exploiting the redundant slice [6, 8] concept in the H.264/AVC [9, 10] video compression standard can prevent error happening due to network packet losses. It is worth pointing out that, these source coding error resilient techniques are independent of the channel coding stage. Secondly, we are going to study the Forward Error Correction techniques for real-time interactive video streaming, where the error correction performance and the caused delay will be jointly taken into consideration.

1.2 Overview of H.264/AVC Video Standard

H.264/MPEG-4 Part 10 or AVC (Advanced Video Coding) [9, 10] is a standard for video coding/compression, and it is currently one of the most widely used formats for compression, and distribution of high definition video. The final drafting work on the first version of the standard was completed in May 2003 [9]. As the previous video coding standards (e.g., H.263 [11] and MPEG-2 [12]), H.264/AVC is a block-oriented motion-compensation-based video coding standard. This standard was developed by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG), this partnership is known as the Joint Video Team (JVT). The ITU-T H.264 standard and the ISO/IEC MPEG-4 AVC standard (formally, ISO/IEC 14496-10 C MPEG-4 Part 10, Advanced Video Coding) are jointly maintained so that they have identical technical content.

The main goals of the H.264/AVC standardization effort have been enhancing compression performance and provision of a "network-friendly" video representation addressing "conversational" (video telephony and conference) and "nonconversational" (storage, broadcast) applications. In terms of compression performance, H.264/AVC provides gains of up to 50% over a wide range of bit rates and video resolutions compared to previous standards [13]. In terms of "network-friendly" video representation, the concept of network abstraction layer (NAL) is designed in order to provide network friendliness to enable simple and effective customization of the use of the video coding

2
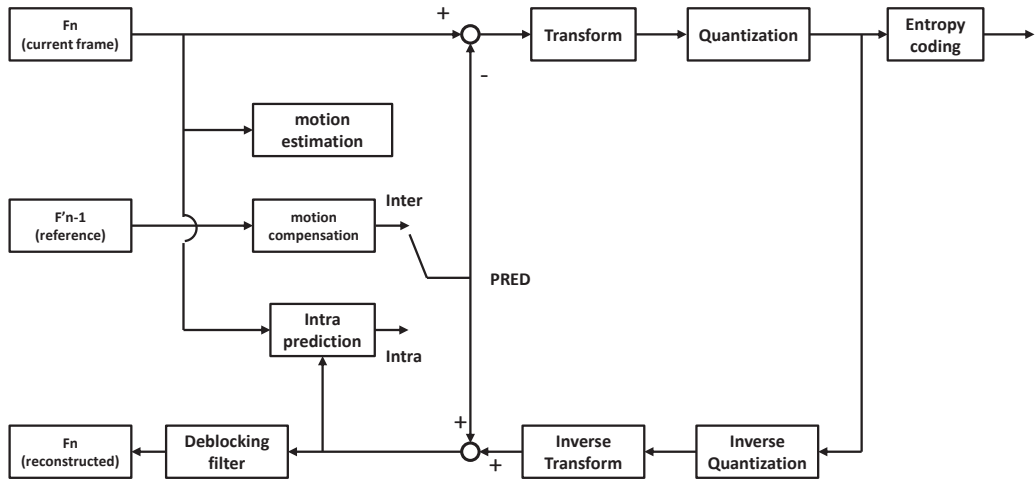
Figure 1.2: H.264/AVC encoder structure.
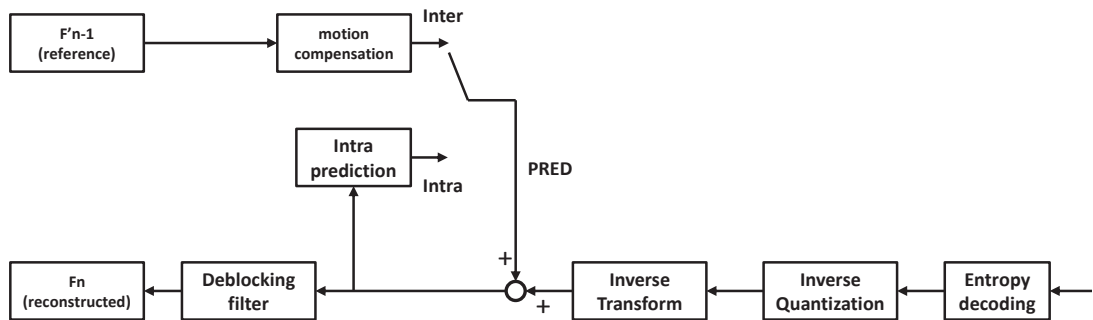
Figure 1.3: H.264/AVC decoder structure.

layer (VCL) for a broad variety of systems.

1.2.1 Encoder and Decoder Structure in H.264/AVC

Inherited from the earlier coding standards, H.264/AVC defines the syntax of an encoded video stream rather than explicitly defining a Codec; however the decoding methods of the bit stream is defined in the standard. Typically, the diagrams of the compliment video encoder and decoder are as depicted as in Figure.1.2 and Figure.1.3, respectively.

Similar to the previous video coding standards, the basic functional units of H.264/AVC include motion prediction, motion estimation, intra prediction, transform, quantization, and entropy coding. To enhance the overall compression performance, the details of these common units are modified in H.264/AVC. The main features of the design that enable high coding efficiency include the following enhancements: variable block-

size motion compensation with small block sizes; quarter-sample-accurate motion compensation; multiple reference picture motion compensation; generalized B frame concepts; in-the-loop deblocking filtering; advanced entropy coding, i.e., CABAC (context-adaptive binary arithmetic coding) and CAVL (context-adaptive variable-length coding).

The H.264/AVC encoder (Fig.1.2) includes two dataflow paths, a *forward* path and a *reconstruction* path. An input frame or field F_n is processed in units of a macroblock (16×16 pixels). Each macroblock is encoded using either Intra prediction mode or Inter prediction mode. For each block (4×4 pixels) in the macroblock, a prediction PRED is formed based on the reconstructed picture samples. In Intra mode, PRED is generated from samples in the current slice that have previously encoded, decoded and reconstructed; for the Inter mode, PRED is formed from the motion compensation region(s) of one or two reference picture(s) selected from the set of list 0 and/or list 1 reference pictures. Then the difference between the prediction signal PRED and the current block is regarded as *residual* signal, which will be transformed, quantized and finally entropy encoded to form the video bitstream.

Inversely, at the video decoder side, the compressed video bitstream will be entropy decoded, inverse transformed, and inverse quantized. Together with the reconstructed prediction signal, the decoded frame will be generated.

1.2.2 Inter Prediction

In the Inter prediction, a macroblock is predicted from one or more previously encoded video frames or fields with block-based motion compensation. Different from the earlier standards, H.264/AVC supports various block sizes (from 16×16 to 4×4) and quarter-pixel accuracy motion vector for the luminance component. In this thesis we focus on the Inter prediction tools available in the Baseline profile.

The luminance component of each 16×16 macroblock can be split in 4 ways for the motion compensation. As depicted in the Fig.1.4, it is either one 16×16 block, two 16×8 blocks, two 8×16 blocks or four 8×8 blocks. If there is 8×8 block chosen within one macroblock, this 8×8 block can be further partitioned in 4 ways (depicted in the Fig.1.5), which are one 8×8 block, two 8×4 blocks, two 4×8 blocks or four 4×4 blocks. Partitioning macroblocks into motion compensated sub-blocks of varying size is known as *tree structured motion compensation*. Each sub-marcoblock requires one
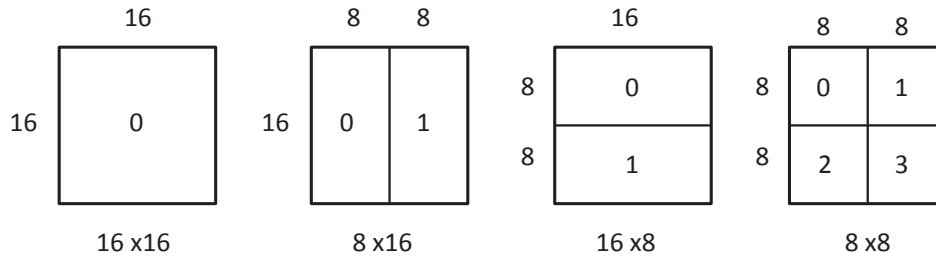
Figure 1.4: Macroblock partition methods: 16×16, 16×8, 8×16, 8×8.
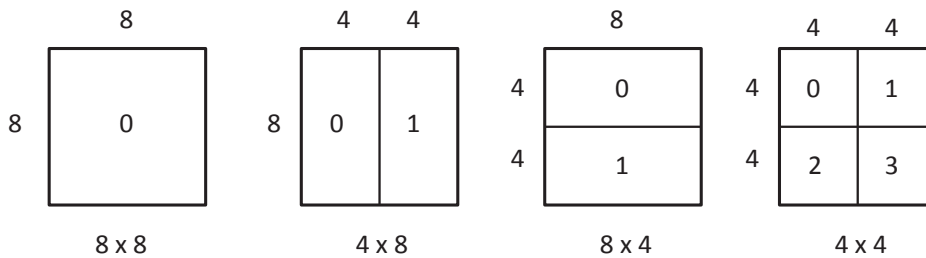
Figure 1.5: Sub-macroblock partition methods: 8×8, 8×4, 4×8, 4×4.

motion vector, so the finer the macroblock is partitioned, and more motion vectors will be needed, and therefore more bitrate will be dedicated for the motion information. Nevertheless, with the finer macroblock partitioning methods, the energy of the residual is lower, and less bitrate is required to represent the residual. So it is a trade-off process to select the proper macroblock partitioning methods. Typically, for the image region with complex texture/detail, finer partitioning methods are preferred, whereas for the homologous region, coarse partitioning methods are enough. The choice of partition size therefore has a significant impact on compression performance.
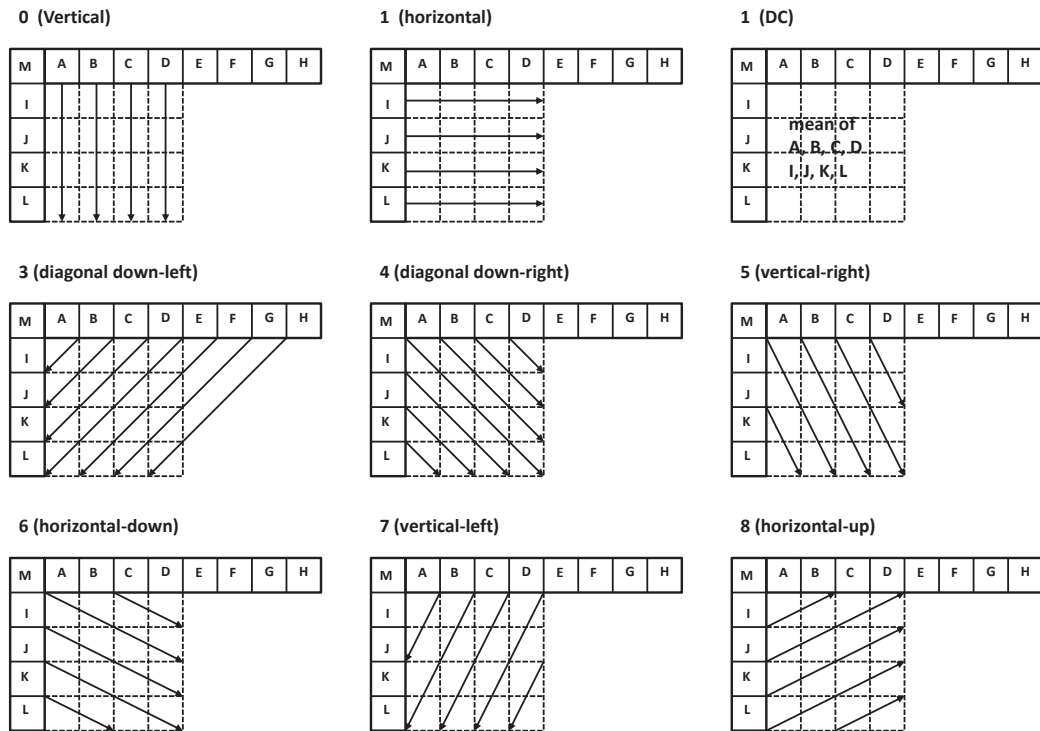
Figure 1.6: 9 intra prediction modes for 4×4 luminance block.

1.2.3 Intra Prediction

In Intra prediction, the prediction block PRED is generated based on the previously encoded and reconstructed neighboring blocks and is subtracted from the current block before encoding. For the luminance component, the predicted block PRED block is form for each 4×4 or 16×16 block. There are a total of 9 possible prediction modes for the intra 4×4 luminance block, which are shown in Fig.1.6. and 4 modes for a 16×16 luminance block, as described in Fig.1.7.
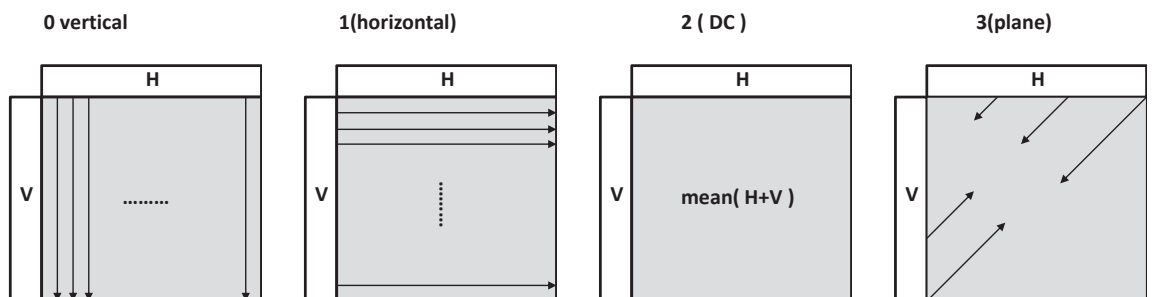
Figure 1.7: 4 intra prediction modes for 16×16 luminance block.

1.2.4 Error Resilient Tools in H.264/AVC

Beside the high efficiency compression performance, a number of design aspects to enable error resiliency are considered in the H.264/AVC standard, including the following highlighted features [10]:

1. *Parameter set structure*: The parameter set design provides robust and efficient representation of header information. As the loss of some key bits of information (such as sequence/picture header information) could have a severe negative impact on the decoding process when using prior standards, this key information was separated for handling in a more flexible and specialized manner in the H.264/AVC design.

2. *NAL unit syntax structure*: Each syntax structure in H.264/AVC is placed into a logical data packet named NAL unit. Instead of forcing a specific bitstream interface to the system as in prior video coding standards, the NAL unit syntax structure allows greater customization of the method of carrying the video content in a manner appropriate for each specific network.

3. *Flexible slicing method and FMO*: Unlike the prior standards, slice sizes in H.264/AVC are highly flexible. Each slice can contain a user-specified length, in terms of bytes or number of macroblocks. Moreover, a new functionality to partition the picture slice groups has been developed, with each slice becoming an independently-decodable subset of a slice group. When used effectively, flexible macroblock ordering (FMO) can significantly enhance robustness to data losses by managing the spatial relationship between the regions that are coded in each slice.

4. *Redundant slices/pictures*: In order to enhance robustness to the video slice loss, the H.264/AVC design contains a new feature to allow the encoder to send redundant version information for regions of pictures, enabling a (typically somewhat lower quality) representation of regions of pictures for which the primary representation has been lost during network transmission. However, the standard does only specify the encapsulation method, but it does not specify how to generate the redundant slices/pictures. In [6], a new approach is proposed to mathematically evaluate the quantization parameter (QP) of the redundant slices, and then the

primary and redundant slices are interleaved into two equally important multiple descriptions for transmission.

5. *Data Partitioning*: Since some coded information (e.g., motion vectors and other prediction information) is more important or more valuable than other information for the purposes of representing the video content, H.264/AVC allows the syntax of each slice to be separated into up to three different partitions for transmission, depending on the categorization of syntax elements.

   The data partition A contains the most important slice date, the header information such as macroblock (MB) types, quantization parameters, and motion vectors. With the loss of data in data partition A, data of the other two partitions becomes useless. Data partition B contains intra coded block patterns (CBPs) and transform coefficients of I-blocks. Because the intra frames and intra-MBs are used as references, the loss of this part will severely impair the video quality of successive frames due to error propagations. Data partition C contains Inter CBPs and coefficients of P-blocks. Compared to the data partition A and B, the data contained in data partition C is less important. However, it is the biggest partition of a coded slice as a large number of frames are coded as P-frames.

6. *SP/SI synchronization/switching pictures*: The H.264/AVC design includes new picture types, SP/SI pictures, that allow exact synchronization of the decoding process of some decoders with an ongoing video stream produced by other decoders without penalizing all decoders with the loss of efficiency resulting from sending an I picture. This can enable switching a decoder between representations of the video content that used different data rates, recovery from data losses or errors, as well as enabling trick modes such as fast-forward, fast-reverse, etc.

1.3 Real-time Video Streaming over Lossy Networks

In this section, the existing real-time error resilient techniques will be categorized and reviewed.

1.3.1 End-to-End Distortion Driven Intra Coding and Motion Prediction

One commonly used method to combat network packet losses is to use Intra coding. The intra-macroblock refreshment approach is standard compatible, and it is an useful tool to combat network packet losses. It can be employed to weaken the inter-picture dependency due to inter prediction, and eventually, cut-off the error propagations. The early intra-macroblock refreshment algorithms are based on randomly inserting Intra macroblocks [14] or periodically inserting Intra contiguous macroblocks [15]. However, in both [14] and [15] the Intra refresh frequency is determined in a heuristic way, and it is costly to code an entire picture by intra-coding. So the trade-off between code efficiency and error resiliency need to be balanced. Zhang *et al.* first treated this problem as the optimization of coding mode selection for each macroblock in [16], and proposed the well-known Recursive Optimal Per-pixel Estimate (ROPE) approach to determine intra-macroblock. In [16] the expected end-to-end distortion for each pixel is calculated in a recursive way, and in the mode selection step, the expected end-to-end distortion is used in the rate-distortion optimization process. In [17], another flexible intra macroblock update algorithm was investigated to optimize the expected rate-distortion performance. In this approach, the end-to-end distortion is calculated by emulating the real channel behaviors, and therefore, the computation complexity is tremendous. Among the methods to get the expected end-to-end distortion, [16] is a pixel-based approach, another block-based approach [18] generates and recursively updates a block-level distortion map for each frame. Recent advances in Recursive Optimal Per-pixel Estimate (ROPE) further expanded its capability to accommodate sub-pixel prediction [19] and burst packet loss [20].

The above mentioned methods exploit network lossy-aware end-to-end distortion to optimally select the Intra coding mode for each macroblock. To further enhance the error resiliency performance of the encoded video streaming, in [21, 22], end-to-end distortion is applied in the motion estimation and motion prediction stage, which is so-called loss-aware motion estimation and loss-aware motion prediction. With this extension, the error-resilience performance is improved further. In recent work [23], SSIM [24] is used to evaluate the end-to-end distortion instead of using conventional PSNR.
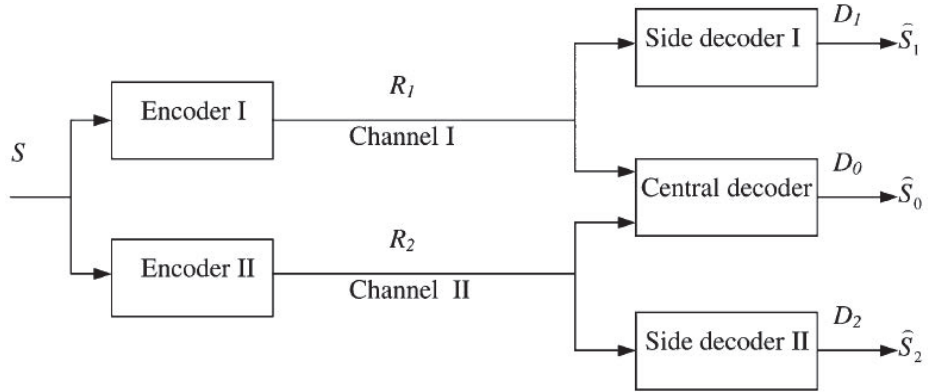
Figure 1.8: A typical MDC structure. Receiving one description could lead to low reconstructed quality; receiving both descriptions could lead to better quality.

1.3.2 Multiple Description Coding and Redundant Coding

Multiple Description Coding (MDC) is an effective tool to combat packet losses in unreliable and non-prioritized networks, particularly for real-time video communication applications where the acceptable end-to-end delay is limited and retransmission is unacceptable. In the paradigm of MDC, two or more equally importance *descriptions* of the video sequence are generated. Each description can be decoded independently with lower but acceptable quality. The decoding quality is improved when more descriptions are received. A typical structure of MDC is shown in Fig.1.8. If one description is received, the side decoder reconstructs the signal with distortion D_1 or D_2. If both descriptions are received, the central decoder reconstructs the signal with distortion D_0, and $D_0 < \min(D1, D2)$. It is not possible to simultaneously minimize both D_0 and $D_1 + D_2$ when the total rate is limited.

One of the most popular solutions to the MDC problem is Multiple Description based on Scalar Quantization (MDSQ) [25]. In this work, two descriptions are generated by quantizing the transform coefficients using two quantization levels. If both of the descriptions are received, the reconstructed quality is equivalent to using a finer quantizer, whereas if only one description is received, the reconstructed quality is equivalent to using a coarse quantizer. This work is later applied to video coding in [26]. However, the main drawback of MDSQ is that it yields descriptions that are not standard compliant. In [27], MDC scheme with correlating transform has been proposed, and it has been extended to video coding in [28]. However, the descriptions generated

with this method cannot be decoded with standard tools.

To generate descriptions that are compatible with video standards (e.g. H.264/AVC [10]), MDC is introduced in the pre/post-processing stages [29]. The basic idea is to split the video source into two sub-videos, which are encoded independently. At the decoder side, in case of two description reception, the decoded descriptions are post-processed to recover the full quality video. On the other hand, if a description gets lost, the received one can reconstruct the video at a coarse quality. In order to introduce a controlled redundancy among the descriptions, so as to achieve the desired tradeoff between the two quality levels, the data must be properly processed prior to splitting. Over-sampling of the original image has been proposed in [30], and generalized to video sequences in [31]. In these papers, zero padding in the 2-D discrete cosine transform (DCT) domain is addressed. Similarly, in [32] an algorithm is proposed for the generation of an arbitrary number of descriptions, based on zero padding in the DCT domain followed by multiple description generation by means of a polyphase downsampler. All the above MDC methods downsample the video sequences in spatial domain; however, it is often preferable to downsample temporally rather than to perform spatial downsampling [33, 34].

Another method to generate standard compatible Multiple Descriptions is to use the redundant slices/pictures concept, which has been defined in the Baseline and Extended Profiles of H.264/AVC [10] standard. In [35], the authors proposed one method to select the frames that need redundant pictures, whereas other frames do not require redundant version. The main flaw of these approaches is that the redundancy is introduced at slice/frame level, meaning that all MBs belonging to the same slice/frame will be regarded as equally important. This degrades the performance of these approaches, especially for non-stationarity video content. In fact, MBs have different characteristics, which is the reason for having different coding modes for different MBs in H.264/AVC. Consequently, it would be better to tune the inserted redundancy at MB level. Armed with this theory, in [36], the authors proposed to allocate the redundancy at macroblock level. The problem of this approach is that it needs to analyze one GOP of frames for redundancy allocation, which means one GOP of delay will be caused, making it not suitable for the real-time interactive video streaming applications. To tackle this problem, a new macroblock level redundancy allocation scheme is proposed in [37], where both the redundancy and the optimal coding mode, i.e., intra coding or inter

11

coding, are selected based on the end-to-end distortion using the ROPE approach [16]. It should be noted that, [37] is done by the thesis author, which is part of this thesis.

1.3.3 Feedback-based Real-time Video Streaming

Retransmission of lost information triggered by feedback from the receiver side is regarded as the most efficient error resiliency approaches for traditional data communication applications. The advantage of feedback-based retransmission is its inherent adaptiveness to various loss rates. Retransmissions are only triggered if the information is actually lost. The overhead encountered is therefore a direct function of loss rate and the sender does not need to receive or estimate information about the expected channel condition. However, for the real-time interactive video communication applications, the packet retransmission is limited because of the stringent one-way latency requirement, which is usually 150-400 ms.

Nevertheless, the feedback information can still be exploited to improve the error resiliency performance in real-time interactive applications. In [38], it assumes that received video data can be decoded faster than real-time, so the later arrival packets (with delay larger than allowed) are exploited to generate error-free reference frames. Later, in [39] an elegant retransmission-based approach for end-to-end error recovery called RESCU (Recovery from Error Spread using Continuous Updates) has been proposed. The main idea of RESCU is to change the frame dependencies in a video sequence such that a retransmission of lost information can be used for error recovery with the help of Accelerated Retroactive Decoding (ARD) [38], despite the low delay requirements of real-time video communication. In RESCU, ARD is used to generate error-free reference frames from retransmitted packets. Another proposal that uses feedback information to stop error propagation is NEWPRED (New Preidction) [40]. Here, feedback about lost packets or correctly received packets is used to restrict the prediction from those image areas that have been successfully decoded. The Reference Picture Selection (RPS) concept introduced in H.263 Annex U and adopted in H.264/AVC supports a standard-compatible implementation of NEWPRED. Based on RESCU [39] and NEWPRED [40], a proxy-based reference picture selection scheme for error resilient conversational video in mobile networks has been proposed in [41]. The problem of the scheme in [41] is that its motion prediction is inefficient, where in the motion prediction process, it does not predict from the most recent frame, but from

the frame $N(N > 1)$ frames before the current frame.

1.3.4 Forward Error Correction Based Unequal Loss Protection

Layered coding splits the information related to an image or video into different sub-streams or layers, which are hierarchically organized. A base layer guarantees the base-quality version of the content; the higher layer, built upon the base layer, allows to improve the base-quality. This hierarchical organization makes a higher layer useless in case that any lower layer is lost [42], making this solution suitable for prioritized networks. On the other hand, if layered data, e.g., JPEG2000 [43] and SVC [44], have to be transmitted on non prioritized networks, Unequal Loss Protection (ULP) tools [45–50] could be employed, where different FEC codes are assigned to different quality layers. Typically, the more protection is provided for the base layer, and less protection for the higher layer.

ULP methods have been proposed not only for hierarchical date, but also for the non-hierarchical date, such as H.264/AVC. Different importance of video frames exists in a group-of-pictures (GOP) level. Due to temporal dependency, the decoding of the current frame strongly depends on its preceding frames in the GOP. The earlier an error occurs in a GOP, the more frames will be corrupted. In [51], unequal amounts of protection are assigned to different frames, accounting for their position in the Group of Pictures (GOP). The distortion is analytically evaluated using a suitable model of the drift [52]. The unequal importance of video data can also be found at the resynchronization packet level if error resilience techniques are used. In [53, 54], the concept of data partitioning in H.264/AVC is exploited for the unequal loss protections systems. In [55], the allocation of the ULP is based on both the frame positions in the GOP and the data partitioning type. In [1], the error resilience features of H.264/AVC (FMO) are exploited together with Reed-Solomon codes to enhance the protection of the video stream. An iterative procedure is proposed for the classification of MBs into slice groups and the identification of the optimal channel rate allocation. In [56], starting from a pre-encoded H.264/AVC stream, i.e., non-progressive video data, hierarchies in the video coded units are identified, so as to enable unequal protection.

1.4 Overview of The Thesis

1.4.1 Contribution and Organization of This Thesis

In this thesis, we are targeting real-time interactive video communication systems for the unreliable networks, i.e, wireless network and Internet. Thus how do design low-delay error resilient techniques, while maintaining good packet loss recovery performance is our objective. The main contributions of the thesis are:

- **Joint redundant motion vector and intra macroblock refreshment**

  We propose a scheme for error-resilient transmission of videos which jointly uses intra-macroblock refreshment and redundant motion vector. The selection of using Intra refreshment or redundant motion vector is determined by the rate-distortion optimization procedure. The end-to-end distortion is used for the rate-distortion optimization, which can be easily calculated with the ROPE [16] method. Experimental results show that the proposed method outperforms both the Intra Refreshment approach and Redundant Motion Vector approach significantly, when the two approaches are deployed separately. This work was published in [57], and presented in Chapter 2.

- **Redundant video coding with end-to-end rate-distortion optimized at macroblock level**

  In Chapter 2, redundant motion vector and intra macroblock are jointly used. In this part, in addition to Intra coding, we propose to add two marcoblock coding modes to enhance the transmission robustness of the coded bitstream, which are inter coding with redundant macroblock and intra coding with redundant macroblock. The selection of coding modes and the parameters for coding the redundant version of the macroblock are determined by the rate-distortion optimization. The end-to-end distortion is employed in the optimization procedure, which considers the source coding and channel conditions. Extensive experimental results show that the proposed approach outperforms other error resilient approaches, for some video sequences, the average PSNR can be up to 4 dB higher than that of the optimal Intra refreshment approach. This work could be regarded as extension work of Chapter 2. This work was published in [37], and also presented in Chapter 3.

- **Dynamic Sub-GOP Forward Error Correction Code for Real-time Video Applications**

In the previous chapters, redundant coding is implemented by duplicating information with the same or lower fidelity. In the following chapters, the methods described are based on *error correction code*, where the inserted redundant information is used to recover the lost information. Reed-Solomon erasure codes are commonly studied as a method to protect the video streams when transmitted over unreliable networks. As a block-based error correcting code, on one hand, enlarging the block size can enhance the performance of the Reed-Solomon codes; on the other hand, large block size leads to long delay which is not tolerable for real-time video applications. In this thesis a novel Dynamic Sub-GOP FEC (DSGF) approach is proposed to improve the performance of Reed-Solomon codes for video applications. With the proposed approach, the sub-GOP, which contains more than one video frame, is dynamically tuned and used as the RS coding block, yet no delay is introduced. For a fixed number of extra introduced packets, for protection, the length of the sub-GOP and the redundancy devoted to each sub-GOP becomes a constrained optimization problem. To solve this problem, a fast greedy algorithm is proposed. Experimental results show that the proposed approach outperforms other real-time error resilient video coding technologies. This work was published in [58], and presented in Chapter 4.

- **A Real-time Error Resilient Video Streaming Scheme Exploiting the Late- and Early-arrival Packets**

In the Chapter 4, the sub-GOP concept is proposed and optimized for the ideal case, where no transmission network delay is taken into consideration. In this chapter, the sub-GOP method is designed for the practical applications. For real-time video streaming systems, the video packets arriving after the display deadline of their frames are considered as *late-arrival* packets, and typically they are discarded. This will affect the current frame and the following ones due to error propagations. For this reason, in this thesis, we propose an approach to exploit the *late-arrival* and out-of-order packets, which includes two mechanisms. The first mechanism will use these packets to update the reference frames to make them more consistent with the encoder side, and this will eventually reduce the

15

error propagations. The second mechanism will use these packets to increase the chance of successfully decoding the Reed-Solomon (RS) code. In the proposed approach, a sub-GOP based systematic RS code is used and optimized to exploit these packets, where the size of each sub-GOP and the parity packet number for each sub-GOP are optimally tuned, taking into consideration the maximum end-to-end delay, the network conditions, and other system parameters, so as to make the best use of the *late-arrival* packets and to exploit the out-of-order packets. Finally, the experimental results show the advantage of the proposed approach over other approaches. This work was published in [59], and presented in Chapter 5.

- **Real-Time Video Streaming Using Randomized Expanding Reed-Solomon Code**

  For the proposed DSGF approach in Chapter 4, although the overall error resilient performance is higher than the conventional FEC schemes, that protect the streamed video frame by frame, its video quality fluctuates within the Sub-GOP. To mitigate this problem, another interactive video streaming scheme using randomized expanding Reed-Solomon code is proposed. For the FEC schemes, typically enlarging the FEC coding block size can improve the error correction performance. For video streaming applications, this could be implemented by grouping more than one video frame into one FEC coding block. However, in this case, it leads to decoding delay, which is not tolerable for real-time video streaming applications. In this thesis, to solve this dilemma, a real-time video streaming scheme using randomized expanding Reed-Solomon code is proposed. In this scheme, the Reed-Solomon coding block includes not only the video packets of the current frame, but could also include all the video packets of previous frames in the current GOP. At the decoder side, the parity-check equations of the current frame are jointly solved with all the parity-check equations of the previous frames. Since video packets of the following frames are not encompassed in the coding block, no delay will be caused for waiting for the video or parity packets of the following frames both at encoding and decoding sides. Experimental results show that the proposed scheme outperforms other real-time error resilient video streaming approaches significantly, This work was published in [60], also is

presented in Chapter 6.

Chapter 2

Joint Redundant Motion Vector and Intra Macroblock Refreshment for Video Transmission

2.1 Introduction

Due to the unreliable underlying networks, the development of error-resilient video coding techniques is a crucial requirement for video communications over lossy networks. Among all the error-resilient video coding techniques, two categories of robust coding approaches are promising and suitable for the real-time interactive video communication systems. One category is based on intra-macroblock refreshment, and another one is redundant coding. The intra-macroblock refreshment approach is standard compatible, and it is an useful tool to combat network packet losses. It can be employed to weaken the inter-picture dependency due to inter prediction, and eventually, cut-off the error propagations. The early intra-macroblock refreshment algorithms are based on randomly inserting Intra macroblocks [14] or periodically inserting contiguous Intra macroblocks [15]. However, in both [14] and [15] the Intra refresh frequency is determined in a heuristic way, and it is costly to code an entire picture by intra-coding. So the trade-off between coding efficiency and error resiliency need to be balanced. Zhang *et al.* first treated this problem as the optimization of coding mode selection for each macroblock in [16], and proposed the well-known Recursive Optimal Per-pixel Estimate (ROPE) approach to determine intra-macroblock. In [16] the expected end-to-end distortion for each pixel is calculated in an recursive way, then in the mode selection step, the expected end-to-end distortion is used in the rate-distortion optimization process.

In [17], another flexible intra macroblock update algorithm was investigated to optimize the expected rate-distortion performance. In this approach, the end-to-end distortion is calculated by emulating the real channel behaviors; therefore, the computation complexity is tremendous. Among the methods to get the expected end-to-end distortion, [16] is pixel-based; another block-based approach [18] generates and recursively updates a block-level distortion map for each frame.

Redundant coding is another effective tool for robust video communications over lossy networks. In [35], an optimal algorithm is presented to determined whether one picture needs redundant version. In [6], redundant slices are optimally allocated based on the slice position in the GOP; and the primary and redundant slices are then interleaved to generate two equal important descriptions of the same data using the MDC paradigm. In [34], the two descriptions are generated by splitting the video pictures into two threads; and then redundant pictures are periodically inserted into the two threads. In both [35] and [34] redundant coding are optimized in frame level; namely all the macroblocks in one frame are encoded with the same redundant coding parameters. For [6], redundant information is allocated in slice level. In all the three approaches, redundant bitrate is allocated to both motion vectors and residual information. In [61] a new approach with only redundant motion vectors is proposed. As the redundant bitrate for motion vector is low, this approach improves the bandwidth utilization with limited primary picture quality degradation. In [62] a significant motion vector protection (SMVP) scheme for error-resilient transmission of videos is proposed. This scheme shows how to determine the significant motion vectors (SMVs) and how much rate should be dedicated to SMVs. The idea behind this scheme is to give more protection to significant motion vectors.

Intra-macroblock refreshment can stop errors in the previous frames, while redundant coding is a way of preventing and minimizing propagated errors in the future frames. Motivated by the two approaches, in this section, we propose an innovative approach that jointly uses intra-macroblock refreshment and redundant motion vector. For each macroblock, intra coding or redundant motion vector is chosen based on the rate-distortion optimization procedure. The loss-aware end-to-end expected distortion is used for this RD optimization (RDO), and the end-to-end distortion is calculated with the ROPE [16] method.

The rest of the section is organized as follows. In Section 2.2, the ROPE method is

19

presented as a preliminary, as it is the method we adopted to calculate the end-to-end distortion, In Section 2.3, the proposed Joint Redundant motion Vector and Intra-macroblock Refreshment (JRVIR) approach is introduced. In Section 2.4, extensive experimental results are given, which validate our approach. Finally, some conclusions are drawn in Section 2.5.

2.2 Preliminary and ROPE Approach

In an ideal error-free environment, the rate-distortion optimized intra/inter mode decision is an efficient tool to determine the macroblock mode based on the cost function defined in [63]. The cost function of all the macroblocks is defined as

$$J_{MB} = D_{MB} + \lambda_{mode} \cdot R_{MB} \tag{2.1}$$

where λ_{mode} is the Lagrange multiplier, D_{MB} and R_{MB} are the encoding distortion and the bitrate in different encoding modes, respectively. This optimization mode is tailored for error-free environments, and no packet loss is considered here.

However, when the compressed videos are transmitted over error-prone networks, traditional schemes cannot adaptively insert intra refresh macroblocks to efficiently stop the channel error propagations. The ROPE approach uses the end-to-end distortion in the RD optimization, which takes into account the channel packet losses. With the ROPE approach, intra macroblocks are optimally used to stop error propagations, and it is defined as follows:

Let f_n^i denotes the original value of pixel i in frame n, and let $\hat{f}_n^i$ and $\tilde{f}_n^i$ denote its encoder and decoder reconstruction, respectively. Because of possible packet loss in the channel, $\tilde{f}_n^i$ can be modeled at the encoder side as a random variable. In the ROPE approach, the D_{MB} is redefined as the overall expected decoder distortion in one macroblock.

$$D_{MB} = \sum_{i \in MB} d_n^i \tag{2.2}$$

$$d_n^i = E\{(f_n^i - \tilde{f}_n^i)^2\}$$

$$= (f_n^i)^2 - 2\,f_n^i\,E\{\tilde{f}_n^i\} + E\{(\tilde{f}_n^i)^2\} \tag{2.3}$$

The overall expected mean-squared-error (MSE) distortion of a pixel is d_n^i, and obviously, it is determined by the first and second moments of the decoder reconstruction.

ROPE provides a recursive algorithm to accurately calculate the two moments for each pixel in a frame.

Let us assume that packet loss events are independent for simplicity, and the average Packet Loss Rate (PLR) p is available at the encoder side. To make it more general, there is no limitation on the slice shape and size. So, the motion vectors from neighboring macroblocks are not always available in the error concealment stage. Therefore, the decoder may not be able to use motion vector from neighboring macroblocks for concealment. Therefore, we assume the decoder copies reconstructed pixels from the previous frame for concealment. The motion prediction at the encoder only employs the previous reconstructed frame. The recursive formulas of ROPE are as follows.

- Pixel in the Intra macroblock

$$E\{\tilde{f}_n^i\} = (1 - p)\hat{f}_n^i + pE\{\tilde{f}_{n-1}^i\} \tag{2.4}$$

$$E\{(\tilde{f}_n^i)^2\} = (1 - p)(\hat{f}_n^i)^2 + pE\{(\tilde{f}_{n-1}^i)^2\} \tag{2.5}$$

- Pixel in the Inter macroblock

$$
\begin{aligned}
E\{\tilde{f}_n^i\} =& (1 - p)(\hat{e}_n^i + E\{\tilde{f}_{n-1}^{i+mv}\}) \\
& + pE\{\tilde{f}_{n-1}^i\}
\end{aligned} \tag{2.6}
$$

$$
\begin{aligned}
E\{(\tilde{f}_n^i)^2\} =& (1 - p)((\hat{e}_n^i)^2 + 2\hat{e}_n^i E\{\tilde{f}_{n-1}^{i+mv}\} \\
& + E\{(\tilde{f}_{n-1}^{i+mv})^2\}) \\
& + pE\{(\tilde{f}_{n-1}^i)^2\}
\end{aligned} \tag{2.7}
$$

where inter coded pixel i is predicted from pixel $i + mv$ in the previous frame. The prediction residual e_n^i is quantized to $\hat{e}_n^i$.

It is important to notice that in order to make it simple, we apply ROPE in its simple setting, where the motion estimation is evaluated at pixel level accuracy, and we use constrained intra prediction, so there are no error propagations in the intra prediction. Recent advances in ROPE further expand its capability to accommodate sub-pixel prediction [19], bursty packet loss [20]. But they are not incorporated here so as to avoid diluting the focus. In the ROPE approach, the end-to-end distortion is only used in the mode selection stage. However, recently, in [21, 22] end-to-end distortion is applied in the motion estimation and motion prediction stage, which is so-called

loss-aware motion estimation and loss-aware motion prediction. With this extension, the error-resilience capability of ROPE is improved further. The loss-aware motion estimation and loss-aware motion prediction are not used in our approach, because we extend ROPE in a different direction. In fact, the gain can be accumulated if both the loss-aware motion estimation and loss-aware motion prediction are applied.

2.3   Proposed JRVIR Approach

As both the redundant motion vector and intra macroblock refreshment are powerful tools for error resilient video communications, in the proposed Joint Redundant motion Vector and Intra macroblock Refreshment (JRVIR) approach, they are jointly applied to further protect the video stream. With the JRVIR approach, all the macroblocks of one frame are divided into three types, namely intra macroblock, inter macroblock (including skip) without redundant motion vector and inter macroblock (including skip) with redundant motion vector. The redundant motion vectors are encapsulated in the redundant picture. Let us take macroblocks in Fig.2.1 as an example. Let us suppose the last macroblock in the first row is a macroblock with redundant motion vector, accordingly, it is stored in the redundant picture. On the contrary, for intra refresh macroblock and inter macroblock without redundant motion vector, there will be no redundant information to be sent in the redundant picture. Therefore, for inter macroblock with redundant motion vector, if the macroblock in the primary picture is lost due to packet losses, the redundant extra motion vector can be used to recover the macroblock. It is important to note that, in the proposed JRVIR approach, as not all the macroblocks need to have redundant motion vector, a new flag is applied in each macroblock to indicate whether there is redundant motion vector. For these macroblocks with redundant motion vector, there will be no transformed coefficients to be encapsulated in the redundant macroblcoks. Therefore, the proposed JRVIR would not be standard compatible, and some small modifications are required for both the encoder and decoder.

In general, Intra coding is more expensive, in terms of rate requirement, with respect to redundant motion vector. Therefore, for the macroblocks with smooth texture and/or macroblocks with slow and translational movements, providing redundant motion vector would lead to better resource utilization, i.e., bitrate, with respect to the Intra coding. Whether to encode one macroblock with intra mode, inter mode with
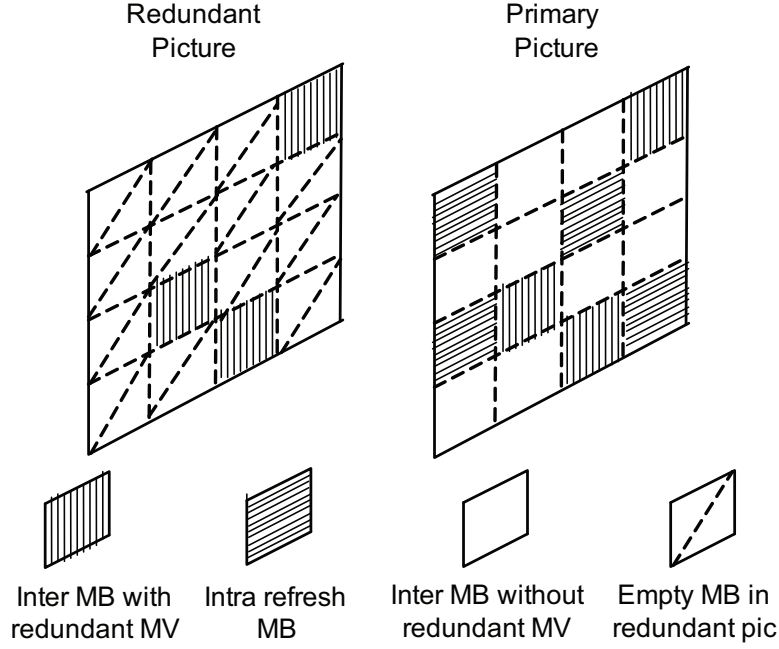
Figure 2.1: Three types of macroblocks in one frame; for the macroblocks with redundant motion vector, the redundant motion vectors are stored in the redundant picture.

redundant motion vector or without motion vector is determined by our JRVIR rate-distortion optimization process.

2.3.1 JRVIR Rate-distortion Optimization

As in other encoding approaches, in the JRVIR rate-distortion optimization process, the encoder selects the coding option O^* for the current encoding macroblock, so that the Lagrangian cost functional is minimized.

$$O^* = \underset{o \in \Gamma_{JRVIR}}{\arg\min} \left(D_{MB}(o) + \lambda_{mode} R_{MB}(o) \right) \tag{2.8}$$

where $D_{MB}(o)$ is the expected end-to-end distortion for mode o, $R_{MB}(o)$ is the rate for this mode and λ_{mode} is the Lagrangian multiplier. Γ_{JRVIR} is a set of encoding options which includes all encoding modes. For the original ROPE approach, the available encoding modes includes Intra mode, SKIP mode and Inter mode, so Γ_{ROPE} = { Intra, SKIP, Inter16 $\times$ 16, Inter16 $\times$ 8, Inter8 $\times$ 16, Inter8 $\times$ 8 }. However, in our JRVIR approach, there are five new modes, they are SKIP, Inter16 $\times$ 16, Inter16 $\times$ 8, Inter8 $\times$ 16 and Inter8 $\times$ 8, all with redundant motion vector. For simplicity, let us use Skip_dup, Inter_dup16 $\times$ 16, Inter_dup16 $\times$ 8, Inter_dup8 $\times$ 16, Inter_dup8 $\times$ 8 to denote the five new modes, with *dup* standing for duplicating motion vetor. Therefore, for the JRVIR

23

approach, the set of encoding options becomes $\Gamma_{JRVIR} = \{$ Intra, SKIP, Inter16×16, Inter16×8, Inter8×16, Inter8×8, Skip_dup, Inter_dup16×16, Inter_dup16×8, Inter_dup8×16, Inter_dup8×8 $\}$.

2.3.2 JRVIR End-to-end Distortion Evaluation

When calculating the expected end-to-end distortion, we can still use formulas (2.4)(2.5) for intra macroblock, and formulas (2.6)(2.7) for inter macroblock without redundant motion vector. For inter macroblock with redundant motion vector, first and second moments of the decoder reconstruction are as follows.

$$
\begin{aligned}
E\{\tilde{f}_n^i\} =&(1-p)(\hat{e}_n^i + E\{\tilde{f}_{n-1}^{i+mv}\}) \\
&p(1-p)(\tilde{f}_{n-1}^{i+mv}) \\
&+ p^2 E\{\tilde{f}_{n-1}^i\} \quad\quad\quad (2.9) \\
E\{(\tilde{f}_n^i)^2\} =&(1-p)((\hat{e}_n^i)^2 + 2\hat{e}_n^i E\{\tilde{f}_{n-1}^{i+mv}\} \\
&+ E\{(\tilde{f}_{n-1}^{i+mv})^2\}) \\
&+ p(1-p)E\{(\tilde{f}_{n-1}^{i+mv})^2\} \\
&+ p^2 E\{(\tilde{f}_{n-1}^i)^2\} \quad\quad\quad (2.10)
\end{aligned}
$$

For those inter macroblocks with redundant motion vector, the probability of receiving the primary information is $1 - p$. The probability of receiving the redundant motion vector while losing the primary information is $p(1 - p)$, and the probability of both the primary information and the redundant motion vector get lost is p^2. With all those probabilities, we can easily get equations (2.9)(2.10) for Inter macroblock with redundant motion vector.

2.3.3 JRVIR Rate Evaluation

In the RD optimization procedure, the rate of the redundant motion vector should be taken into account. For those redundant motion vectors, encoding them without exploiting the correlation among them can cost a significant number of bits. Motion vectors for neighboring macroblocks are often highly correlated, so each motion vector is predicted from vectors of nearby, and previously coded macroblocks. Therefore, the motion vector encoding procedure in H.264/AVC standard [10], which includes motion vector prediction, is adopted to encode the redundant motion vector to reduce bits. However, it is worth noticing that, in our JRVIR approach, we do not provide

24

redundant motion vector for all the inter macroblocks. For example, in Fig.2.1 only three macroblocks have redundant motion vector, when encoding the redundant motion vector for the macroblock in row 3, there are no motion vectors to predict from, because its up and left macroblocks do not have redundant motion vectors. As a result, the performance is compromised.

To determine the required rate to encode a macroblock using the JRVIR algorithm, let us assume that encoding the macroblock itself and its redundant motion vector would use R_{mb} and R_{mv} bits respectively. For encoding mode $o \in \{$Intra, Skip, Inter16×16, Inter16×8, Inter8×16, Inter$8 \times 8\}$, $R_{MB}(o)$ in (2.8) equals to R_{mb}

$$R_{MB}(o) = R_{mb} \tag{2.11}$$

For encoding mode $o \in \{$Skip_dup, Inter_dup16×16, Inter_dup16×8, Inter_dup8×16, Inter_dup$8 \times 8\}$, the value of $R_{MB}(o)$ is

$$R_{MB}(o) = R_{mb} + R_{mv} \tag{2.12}$$

2.3.4 Lagrange Multiplier Selection

The Lagrange multiplier λ_{mode} in (2.8) controls the rate-distortion trade-off. For the error-prone environment, extensive experimental evidence suggests that there is no significant performance difference between using the Lagrange multiplier tailored to the error-free or the error-prone environment. This argument has also been confirmed in [17]. So λ_{mode} is set as the one tailored to error-free environment.

$$\lambda_{mode} = 0.85 \times 2^{(QP-12)/3} \tag{2.13}$$

where QP is the quantization parameter.

2.3.5 Pseudo Code of JRVIR Algorithm

The whole mode selection process of the proposed JRVIR approach is described in Algorithm 1. It is important to note that, in the proposed JRVIR approach, the end-to-end distortion is used in the rate-distortion optimization process, and five new encoding modes are adopted. Upon the optimal encoding mode is selected, the first and second moments for all the pixels in the current macroblock are recorded based on the selected encoding mode, and those values will be recursively used in the rate-distortion optimization process of next frame. At the decoder side, if the primary slice is available,

the redundant motion vector will be discarded; when the primary slice is lost while the redundant motion vector is available, the motion vector will be used to conceal the lost region by copying into the lost macroblock the region indicated by the redundant motion vector. In general, with the correct motion vector, the concealed pixels will be much more accurate than those generated by Temporal Replacement (TR), which copies the pixels from the same positions in the previous frame.

Algorithm 1 The algorithm of mode selection in JRVIR

$RD\_cost \Leftarrow \infty$
$best\_mode \Leftarrow \infty$
for for each mode $o \in \Gamma_{JRVIR}$ **do**
  if $o \in \{\text{INTRA }\}$ **then**
    calculate D_{MB} using Equation (2.2)(2.3)(2.4)(2.5)
    calculate R_{MB} using Equation (2.11)
  else
    if $o \in \{\text{Skip, Inter}16 \times 16, \text{Inter}16 \times 8, \text{Inter}8 \times 16, \text{Inter}8 \times 8 \}$ **then**
      calculate D_{MB} using Equation (2.2)(2.3)(2.6)(2.7)
      calculate R_{MB} using Equation (2.11)
    else
      if $o \in \{\text{Skip\_dup, Inter\_dup}16 \times 16, \text{Inter\_dup}16 \times 8, \text{Inter\_dup}8 \times 16, \text{Inter\_dup}8 \times 8\}$ **then**
        calculate D_{MB} using Equation (2.2)(2.3)(2.9)(2.10)
        calculate R_{MB} using Equation (2.12)
      end if
    end if
  end if
  calculate J_{MB} using Equation (2.1)(2.13)
  if $J_{MB} < RD\_cost$ **then**
    $RD\_cost = J_{MB}$
    $best\_mode = o$
    record the value of $E\{\tilde{f}_n^i\}$ and $E\{(\tilde{f}_n^i)^2\}$
  end if
end for

2.4 Experimental Results

Our simulation setting builds on the JM14.0 H.264 codec [64], with constrained intra prediction and Context-Adaptive Binary Arithmetic Coding (CABAC) entropy coding used. Rate control mechanism in the JM codec is used with one common quantization scale to all the macroblocks of one row. Pixel level accuracy motion estimation and prediction are used. Each slice contains one row of macroblocks (22 macroblocks for the CIF (Common Intermediate Format) video sequences) for both primary and redundant

frames, and one slice per network packet is adopted, and therefore the term packet and slice are used interchangeably. The IPPP GOP structure is used, and it is assumed that the I-frame is transmitted over secure channel. A random packet-loss generator is used to drop the packets according to the required packet loss rate, except the burst packet-loss is specified explicitly. The luminance PSNR (Y-PSNR) is averaged over 200 trials to get statistical meaningful results. To evaluate the proposed JRVIR approach, we use conventional ROPE [16] and Redundant Motion Vector (RMV) [61] as benchmark.

Firstly, the frame by frame average PSNR are reported in Fig.2.2, for the three approaches, namely the JRVIR approach, ROPE and RMV. Both CIF sequence Foreman and Silent are encoded at 1 Mbps bitrate. The packet loss rate is 10%. From the figures, it is observed that for all the frames, the JRVIR frame quality is always better than that of RMV and ROPE. For the Forman sequence, for some frames the PSNR of JRVIR can be up to 2.5 dB higher than that of ROPE, and up to 8.5 dB higher than that of RMV. At the beginning of the sequence, the PSNR of JRVIR and RMV are quite similar, but with the increase of frame number, the quality gap between the two approaches increases dramatically. This phenomena indicates that, when the GOP length is small, the RMV approach can protect the video stream effectively; when the GOP length is relatively large, the RMV approach can not work properly. For Foreman, the average PSNR for JRVIR approach is 33.39 dB; it is higher than that of ROPE and RMV, which are 32.27 dB and 28.39 dB, respectively. For Silent, the average PSNR for JRVIR approach is 37.56 dB, while for ROPE and RMV it is 36.83 dB and 31.27 dB. For Foreman, the gap between JRVIR and ROPE is larger than that of Silent; this is because the movement in Foreman is more translational than that in Silent, and this leads to more inter macroblocks with redundant motion vectors to be used in the Foreman case. Interestingly, with the JRVIR approach, 8.18% of macroblocks in P-frame are intra-coded macroblocks, while 35.02% are inter macroblocks with redundant motion vector. In the ROPE approach, 18.40% macroblocks are intra macrobloks, which is more than that of JRVIR by nearly 10%.

In order to further evaluate the error-resilient performance of the JRVIR approach, we compare the video quality for different packet l oss rate in Fig.2.3, with GOP length 150 and 15 in Fig.2.3.(a) and Fig.2.3.(b), respetively. CIF Foreman sequence is used, and the target bitrate is 1Mbps. From the figures, we can see that, for different packet loss rates (0%-20%) and GOP length, the JRVIR approach can provide the best video

quality among the three approaches. In Fig.2.4 video quality versus the bitrate is presented for the three approaches. CIF Foreman sequence is used, the packet loss rate is 10%, and the GOP length is 150 and 15. In the 200 Kbps to 1 Mbps bitrate range, the proposed JRVIR approach outperforms the other two approaches, and the performance gap between JRVIR and the other two approaches increases with the bitrate. In both Fig.2.3 and Fig.2.4, it is interesting to note that, with long GOP length, ROPE can provide better video quality than RMV, while for short GOP length, RMV outperforms ROPE. This is because, in the ROPE approach, intra coding macroblocks are optimally inserted, the PSNR inside one GOP is more stable than RMV, while for the RMV approach, intra coding is not used, and consequently the PSNR inside one GOP drops incessantly.

Table 2.1: Video quality (dB) of JRVIR, RMV and ROPE for different bitrate (kbps) and packet loss rates; for ROPE the percentage of intra macroblock is provided in brackets, whereas for JRVIR the first number in brackets is the percentage of intra macroblock, the second is the percentage of macroblock with redundant motion vector.

Sequence	Rate	Method	Packet loss rate (PLR)			
			5%	10%	15%	20%
News	256	RMV	30.89	29.70	28.71	27.70
		ROPE	32.27 (2.68)	31.42 (3.56)	30.78 (4.15)	30.20 (4.62)
		JRVIR	**32.29** (1.54, 2.52)	**31.70** (1.82, 3.90)	**31.18** (2.11, 4.41)	**30.73** (2.46, 4.94)
Silent	384	RMV	30.75	29.00	28.22	27.52
		ROPE	33.56 (4.70)	32.75 (5.96)	32.10 (6.76)	31.60 (7.29)
		JRVIR	**33.76** (2.93,3.49)	**33.05** (3.43, 5.08)	**32.45** (3.95, 5.76)	**31.96** (4.54, 6.04)
Foreman	512	RMV	29.32	27.42	25.89	24.56
		ROPE	31.48 (6.78)	30.29 (9.40)	29.42 (11.32)	28.66 (13.06)
		JRVIR	**32.03** (3.84, 18.94)	**31.16** (4.84, 25.40)	**30.32** (5.54, 27.52)	**29.56** (6.90, 27.97)
Highway	1024	RMV	34.73	32.54	30.07	29.48
		ROPE	37.71 (11.68)	36.64 (15.09)	35.76 (16.52)	35.01 (18.10)
		JRVIR	**38.06** (7.10, 9.87)	**37.20** (8.80, 12.34)	**36.50** (10.25, 13.85)	**35.74** (11.70, 13.67)
Stefan	2048	RMV	25.02	21.81	19.69	18.22
		ROPE	28.31 (15.27)	26.63 (19.53)	25.49 (22.13)	24.60 (23.65)
		JRVIR	**29.54** (6.38, 19.85)	**27.50** (9.59, 18.81)	**26.09** (12.86, 16.82)	**24.99** (15.24, 14.71)

In Table.2.1, experimental results for video sequences with varies degree of movement and bitrate are reported. For all the video sequences, GOP length is 150. We can observe that, in different test environments, the proposed JRVIR approach always outperforms both ROPE and RMV approach. It is interesting to notice that, the JRVIR approach uses less intra macroblocks than ROPE so as to allocate bitrate for redundant motion vector. It is noted that for the ROPE approach, the higher the packet loss rate is, the more macroblocks are encoded with intra mode, whereas for the JRVIR approach, the total number of intra macroblocks and macroblocks with redundant motion vector increases. Table.2.1 shows that for the Foreman and Stefan sequences, nearly 20% percent of all the macroblocks are encoded with redundant motion vectors. Accordingly, the gaps between the JRVIR and ROPE approaches for these two video

Table 2.2: The time duration of encoding 30 frames for various video sequences and bitrates for JRVIR and JM software, average packet loss rate 10% is used.

Sequence	bitrate (Kbps)	JRVIR (second)	JM 14.0 (second)
News	256	41.19	40.79
Silent	384	40.51	39.35
Foreman	512	42.86	40.97
Highway	1024	42.61	41.63
Stefan	2048	42.25	40.87

sequences are relatively larger than other sequences.

The actual network loss behavior has been addressed by many papers, and it is agreed that Internet packet loss often exhibits finite temporal dependency, which means if current packet is lost, the next packet is also likely to be lost. This leads to burst packet losses, with average burst length of two for the Internet [65]. Therefore, besides i.i.d. (independent and identically distributed) random packet loss model, we also use burst loss model for simulation, and as indicated in [65], we set the average burst length as two. In practical burst loss environments, the transmission order of the primary and redundant packets would affect the performance. In our simulations, all the redundant packets of one frame are transmitted after the last primary packet of this frame; therefore, there is no interleaving delay. In Fig.2.5, the PSNR versus bitrate curves in burst loss environments are plotted. The results are similar with that in the i.i.d. case, and the proposed JRVIR approach can provide best video quality among the three approaches. This makes us conclude that, the error resilient performance of the proposed JRVIR approach is robust in different error distribution models.

In Table.2.2, we compare the encoding time of JRVIR with JM 14.0. In order to have a fair comparison, we use the same configuration file for the two approaches. It is interesting to see that, the time costs for the two approaches are quite similar. In all cases, JRVIR costs less than 5% extra encoding time; this makes the JRVIR approach suitable for the real-time hand-device applications, where the battery capacity is usually the bottleneck. This is because, in the H.264/AVC encoding process, the motion estimation step is the main time-consuming task, so in comparison with this step, the end-to-end distortion calculation and new mode selection task costs much less time.

2.5 Conclusions

In this section, a joint redundant motion vector and intra macroblock refreshment approach has been proposed to combat packet loss. Besides the traditional skip, inter and intra mode, we have added a set of new modes, which are inter coding modes with redundant motion vector. the rec Given the packet loss rate and the channel bitrate, constructed distortion at the decoder side and the total bitrate for each mode are estimated at the encoder during the mode selection process. Based on the estimated end-to-end Rate-Distortion (RD) cost, the optimal encoding mode is selected. Equipped with the two tools, namely intra macroblock refreshment and redundant motion vector, experimental results show that the proposed approach outperforms that of using them separately.

It is worth reporting that the work reported in this section has led to the following publication:

1. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Joint Redundant Motion Vector and Intra Macroblock Refreshment for Video Transmission, EURASIP Journal on Image and Video Processing, 2011:12, doi:10.1186/1687-5281-2011-12
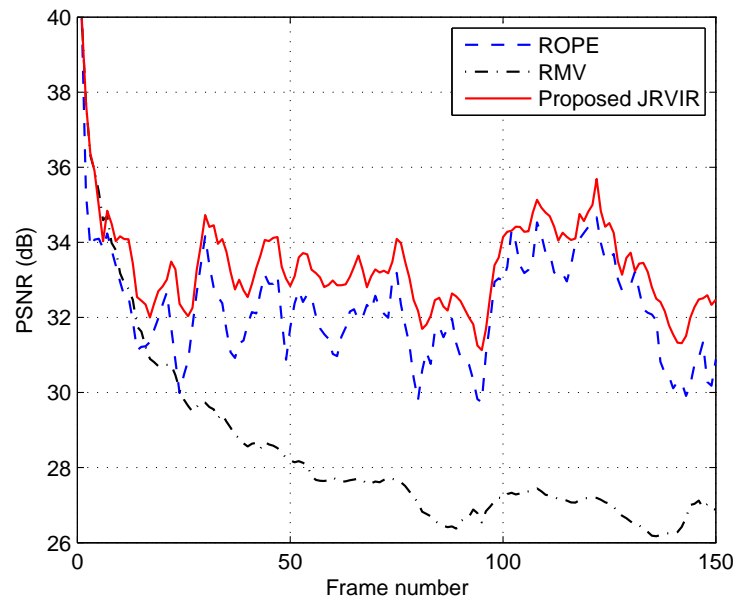
(a) Foreman
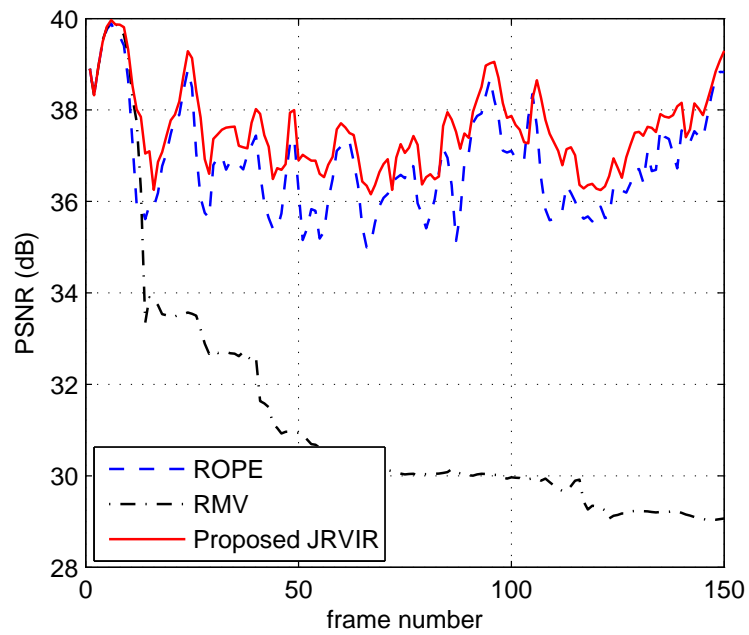
(b) Silent

Figure 2.2: Frame by frame comparison with bitrate 1 Mbps; average packet loss rate 10%; (a)CIF Forman, (b) CIF Silent.
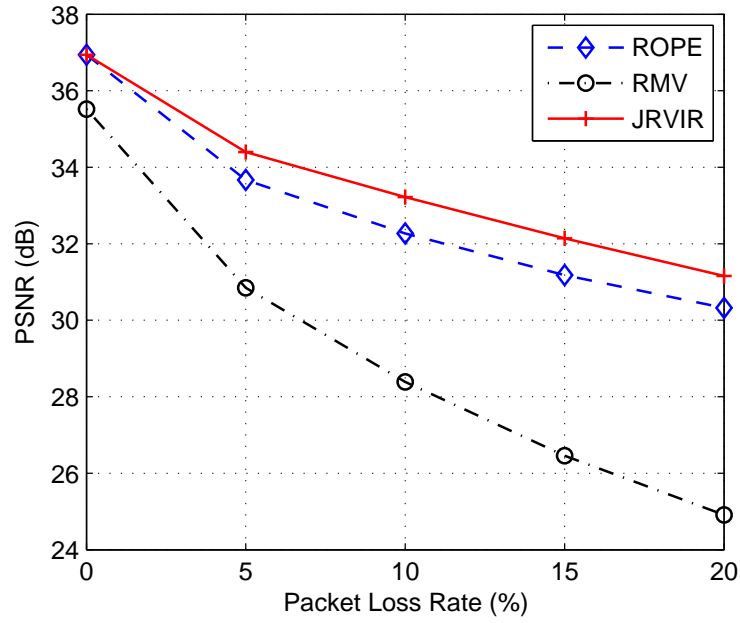
(a) GOP length 150
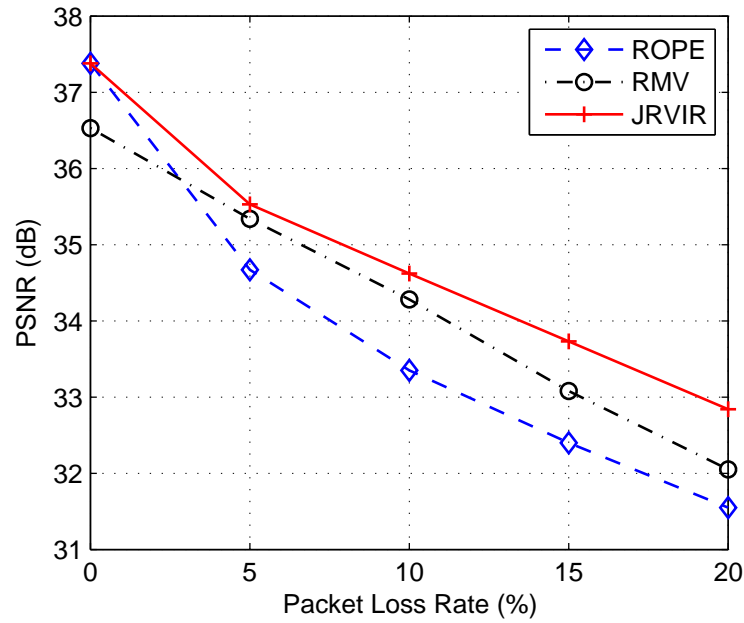
(b) GOP length 15

Figure 2.3: Average PSNR comparison under different packet loss rates; CIF Foreman sequence is used; target bitrate is 1 Mbps; (a) GOP length 150, (b) GOP length 15.
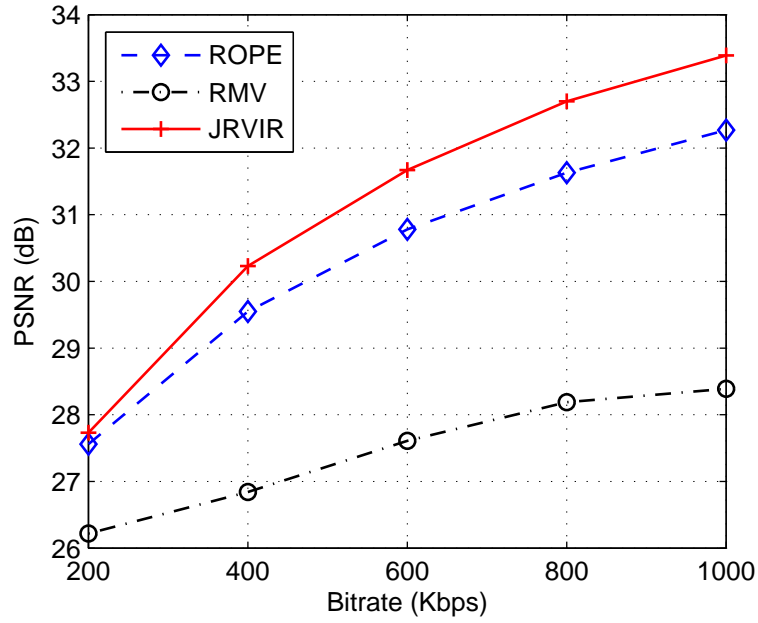
(a) GOP length 150
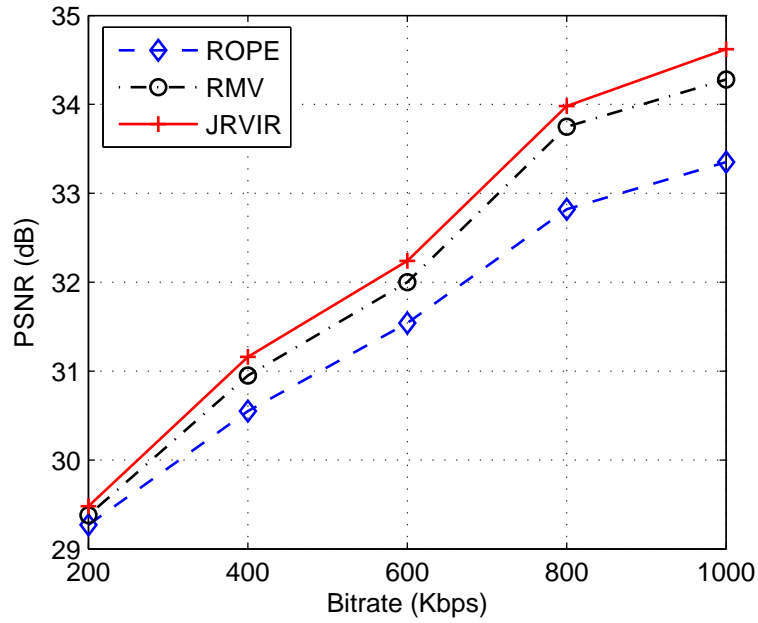
(b) GOP length 15

Figure 2.4: Average PSNR comparison under different bitrate; packet loss rate is 10%; CIF Foreman sequence is used; (a) GOP length 150, (b) GOP length 15.
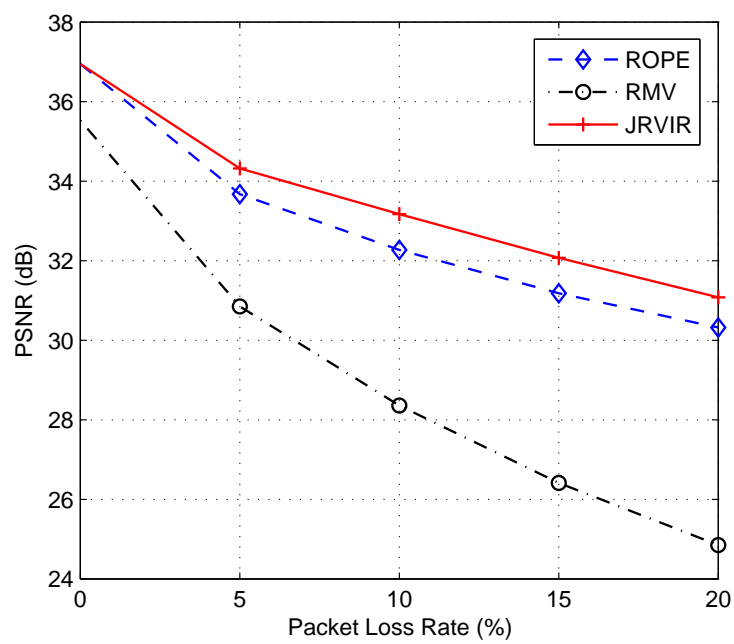
Figure 2.5: Average PSNR versus packet loss rate in burst loss environments; the average burst length is 2; CIF Foreman sequence is used; target bitrate is 1 Mbps; GOP length 150.

Chapter 3

Redundant Video Coding with End-to-end Rate-distortion Optimized at Macroblock Level

Intra macroblock refreshment [15–17] is an effective approach for error resilient video coding. It can be employed to weaken the inter-picture dependency due to inter prediction, and eventually, cut-off the error propagations. Redundant coding is another effective tool for robust video communication over lossy network. In [6], redundant information is allocated in slice level. In [36], redundant coding is optimized in macroblock level. However, in order to optimally tune the redundancy, this approach needs all the motion vector information in one GOP, which leads to a delay of one GOP. Consequently, this work cannot be applied in real-time applications, such as video conference.

Intra-macroblock refreshment can stop errors in the previous frames, while redundant coding is a way of preventing errors in the future frames. In order to take advantage of the two approaches, in the previous section, we propose to jointly use intra coding and redundant motion vector. In this section we propose another approach, by adding two new encoding modes, namely inter coding with redundant macroblock and intra coding with redundant macroblock, in addition to the conventional Intra and Inter coding modes. This approach is called Hybrid Redundant Macroblock and Intra-macroblock Refreshment (HRMIR). The redundant version macroblock is encoded with lower quality and rate, which is implemented by scaling the Quantization Parameter (QP). The selection of coding modes and the parameters for coding the redundant version of the macroblock are determined by the rate-distortion optimization procedure. It is worth noticing that the loss-aware end-to-end expected distortion is used for

the RD optimization, and the end-to-end distortion is calculated with the ROPE [16] method. Since calculating the end-to-end distortion with the ROPE method causes no additional delay, the proposed approach is suitable for real-time applications.

The rest of the section is organized as follows. In Section 3.1 the proposed HRMIR approach is introduced. In Section 3.2 extensive simulation results are given, which validate our approach. Finally, some conclusions are drawn in Section 3.3.

3.1 Proposed HRMIR Approach

As redundant coding and intra macroblock refreshment are both powerful tools for error resiliency video communication, in the proposed approach, they are hybridly applied to further protect the video stream. With the Hybrid Redundant Macroblock and Intra macroblock Refreshment (HRMIR) approach, all the macroblocks of one frame are divided into four types, namely intra macroblock, inter macroblock, inter macroblock with redundant version and intra macroblock with redundant version. The redundant version macroblocks are encapsulated in the redundant picture. It is important to note that, the concept of redundant slice is part of the H.264/AVC standard. In order to make the proposed approach fully compatible with the H.264/AVC standard, for those macroblocks without redundant version, SKIP mode could be used. Let us take macroblocks in Fig.3.1 as an example, suppose that the last macroblock in the first row is an inter macroblock with redundant version, accordingly, the redundant macroblock is stored in the redundant picture. Therefore, for macroblock with redundant version, if the macroblock in the primary picture is lost due to packet loss, the redundant version can be used to replace the macroblock. On the contrary, for intra macroblock and inter macroblock without redundant version, there will be no redundant information to be sent in the redundant picture.

It is worth noticing that, in general, the redundant version macroblock is encoded with lower bit rate than primary one, so the video quality is also lower than primary one. In our approach this is implemented by setting a relative larger Quantization Parameter (QP) for redundant version macroblock. Like the selection of the coding type for each macroblock, the selection of the appropriate QP value for redundant macroblock is also optimized in the end-to-end RD optimization process. Fig.3.2 shows the QP value for redundant frame in the Foreman CIF sequence, where the QP of primary macroblock is 22. In order to present all information in one figure, we use
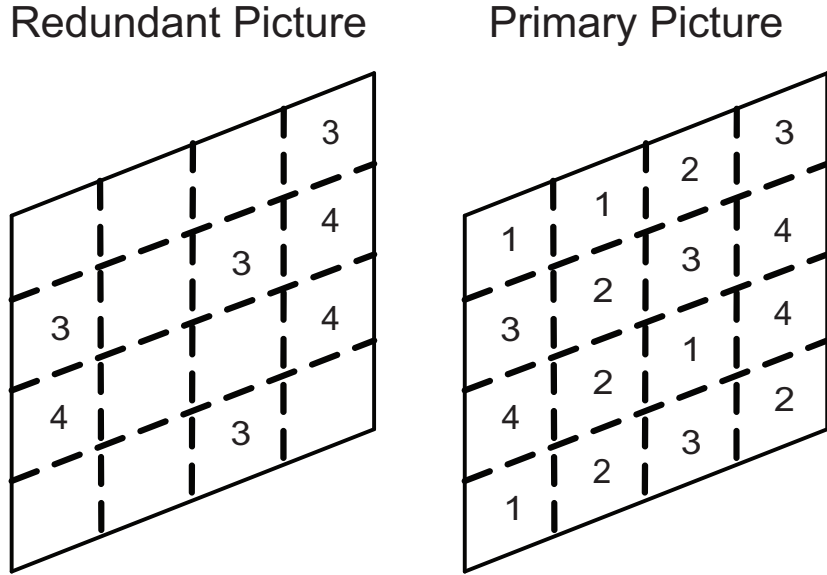
Figure 3.1: Four types of macroblocks in one frame: 1 stands for inter macroblock, 2 stands for intra macroblock, 3 stands for inter macroblock with redundant version and 4 stands for intra macroblock with redundant version; the redundant version macroblocks are encapsulated in the redundant picture.

positive number for inter macroblock and negative number for intra macroblock. The valid QP range is (1-51) in H.264/AVC, so we use 60 to denote inter macroblock without redundant version and -60 to denote intra macroblock without redundant version. For example, if a macroblock in Fig.3.2 has a value -34, this means it is an intra macroblock with QP 34; for a macroblock with value 34, it is an inter macroblock with QP 34. It can be seen that most of the background areas are encoded with inter coding without redundant version, because these areas are relatively static, and with the temporal replacement concealment algorithm losing these areas will not lead to huge distortion. On the contrary, the parts of foreground, which is the foreman face area in this frame, are strongly protected with intra coding and/or redundant coding. Note both the marcoblock type and QP value is optimized in the RD optimization process, which is presented in the next section.

3.1.1 HRMIR Rate-distortion Optimization

As in the other encoding approaches, in the HRMIR rate-distortion optimization process, the encoder selects the coding option O^* for the current macroblock, so that the Lagrangian cost function is minimized.

$$O^* = \underset{o \in \Gamma_{\mathrm{HRMIR}}}{\arg\min} \left(D_{MB}(o) + \lambda_{\mathrm{mode}} R_{MB}(o) \right) \tag{3.1}$$
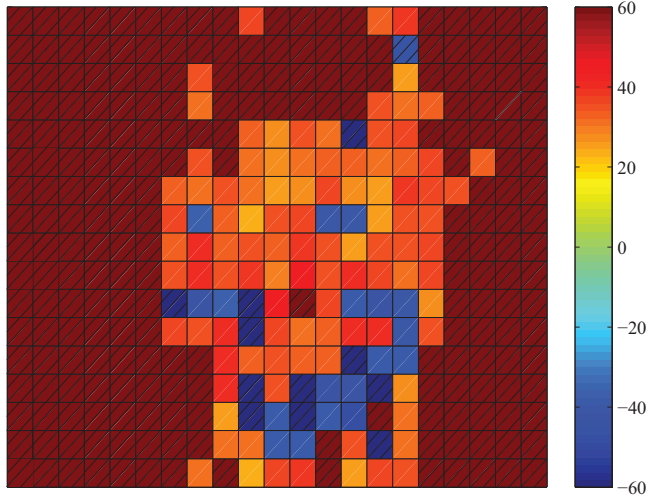
Figure 3.2: Macroblock level QP value of redundant coding for one frame in the Foreman CIF sequence; positive number for inter macroblock and negative number for intra macroblock; we use 60 and hatching to denote inter macroblock without redundant version, -60 and hatching to denote intra macroblock without redundant version.

where $D_{MB}(o)$ is the expected end-to-end distortion for mode o, $R_{MB}(o)$ is the rate for this mode and λ_{mode} is the Lagrangian multiplier. Γ_{HRMIR} is a set of encoding options which includes all encoding modes. For the original ROPE approach, the available encoding modes includes intra mode I and inter mode P, so $\Gamma_{\mathrm{ROPE}} = \{I, P\}$. However, in our HRMIR approach, there are two new modes; they are intra mode with redundant version macroblock and inter mode with redundant version macroblock. For simplicity, let us use I_r^u and P_r^v to denote the two new modes, respectively, with r standing for redundant coding, u representing the candidate QP value in the intra redundant coding and v representing the candidate QP value in the inter redundant coding. Therefore, for the HRMIR approach, the set of encoding options become $\Gamma_{\mathrm{HRMIR}} = \{I, P, I_r^u, P_r^v\}$. In general, the QP value of redundant coding is larger than that of primary coding. Let us use QP_I and QP_P to denote the primary QP value of intra and inter coding, respectively. In the redundant coding, candidate QP value is $u \in \{u | QP_I \leq u \leq 51\}$ and $v \in \{v | QP_P \leq v \leq 51\}$, where 51 is the maximum QP value in H.264/AVC [10].

3.1.2 HRMIR End-to-end Distortion and Rate

When calculating the expected end-to-end distortion, we can still use the equations (2.4)(2.5) for intra macroblock without redundant coding, and equations (2.6)(2.7) for

inter macroblock without redundant coding. For intra macroblock with redundant coding, first and second moments of the decoder reconstruction are as follows.

$$E\{\tilde{f}_n^i\} = (1-p)\hat{f}_n^i + p(1-p)\hat{f}_n^{i,u}$$
$$+ p^2 E\{\tilde{f}_{n-1}^i\} \tag{3.2}$$
$$E\{(\tilde{f}_n^i)^2\} = (1-p)(\hat{f}_n^i)^2 + p(1-p)(\hat{f}_n^{i,u})^2$$
$$+ p^2 E\{(\tilde{f}_{n-1}^i)^2\} \tag{3.3}$$

where in the primary coding f_n^i is quantized to $\hat{f}_n^i$, and in the redundant coding it is quantized to $\hat{f}_n^{i,u}$, here u is the redundant QP value.

Similarly, for inter macroblock with redundant coding, first and second moments of the decoder reconstruction are as follows.

$$E\{\tilde{f}_n^i\} = (1-p)(\hat{e}_n^i + E\{\tilde{f}_{n-1}^{i+mv}\})$$
$$+ p(1-p)(\hat{e}_n^{i,v} + E\{\tilde{f}_{n-1}^{i+mv(v)}\})$$
$$+ p^2 E\{\tilde{f}_{n-1}^i\} \tag{3.4}$$
$$E\{(\tilde{f}_n^i)^2\} = (1-p)((\hat{e}_n^i)^2 + 2\hat{e}_n^i E\{\tilde{f}_{n-1}^{i+mv}\}$$
$$+ E\{(\tilde{f}_{n-1}^{i+mv})^2\})$$
$$+ p(1-p)((\hat{e}_n^{i,v})^2 + 2\hat{e}_n^{i,v} E\{\tilde{f}_{n-1}^{i+mv(v)}\}$$
$$+ E\{(\tilde{f}_{n-1}^{i+mv(v)})^2\})$$
$$+ p^2 E\{(\tilde{f}_{n-1}^i)^2\} \tag{3.5}$$

where in the primary coding, pixel i is predicted from pixel $i+mv$ in the previous frame, the prediction residual e_n^i is quantized to $\hat{e}_n^i$. In the redundant coding, the redundant QP value is v, pixel i is predicted from pixel $i + mv(v)$ in the previous frame, the prediction residual e_n^i is quantized to $\hat{e}_n^{i,v}$.

For those intra and inter macroblocks with redundant coding, the probability of receiving the primary macorblock is $1 - p$. The probability of receiving the redundant macroblock while losing the primary information is $p(1 - p)$, and the probability of losing both the primary and redundant macroblocks is p^2. With all those possibilities, we can easily get equations (3.2)(3.3)(3.4)(3.5) for macroblock with redundant version. It is important to note that when the macroblock is encoded with redundant version, namely $o \in \{I_r^u, P_r^v\}$, the total bit rate $R_{MB}(o)$ is calculated by summing up the bit rate used for both primary and redundant coding.

39

3.1.3 Lagrange Multiplier Selection

The Lagrange multiplier λ_{mode} in (3.1) controls the rate-distortion trade-off. For the error-prone environment, extensive experimental evidence suggests that there is no significant performance difference between using the Lagrange multiplier tailored to the error-free or the error-prone environment. This argument has also been confirmed in [17]. So λ_{mode} is set as the one tailored to error-free environment.

$$\lambda_{mode} = 0.85 \times 2^{(QP-12)/3} \tag{3.6}$$

where QP is the quantization parameter.

3.1.4 Computational Complexity Reduction

In the HRMIR rate-distortion optimization procedure, in order to find the optimal QP value for redundant coding, we need to calculate the rate-distortion cost for all possible redundant QP values; therefore, the computation complexity is tremendous. For example, let us assume the primary QP value is 22, in the Rate-Distortion Optimization (RDO) procedure described in Sect.3.1.1, the encoding options are $\Gamma_{\mathrm{HRMIR}} = \{I, P, I_r^u, P_r^v\}$, then both I_r^u and P_r^v have $(51 - 22 + 1)$ possible redundant QP values, here 51 is the maximum QP value in H.264/AVC. Therefore, Γ_{HRMIR} includes 62 encoding options (both I_r^u and P_r^v have 30 QP values plus intra/inter coding without redundant version).

By lowing the number of encoding options, the computation complexity will be reduced. Let us set the redundant QP increase step as QP_{step}, then the candidate QP value would be $u \in \{u|u = QP_I + K \times QP_{step}, u \leq 51, K = 0, 1, 2...\}$ and $v \in \{v|v = QP_P + K \times QP_{step}, v \leq 51, K = 0, 1, 2...\}$.

In Fig.3.3 the trade-off between PSNR and computation complexity is reported. It is observed that when the value of QP_{step} is set as 5 and 10, the PSNR is lower than that when the QP_{step} is 1. However, the PSNR decrease is very limited. The computation overhead for the $QP_{step} = 5$ case is nearly 1/5 of that for the $QP_{step} = 1$ case, but the resulting decrease of PSNR is less than 0.3dB. Even when the QP_{step} value is set to 10, the PSNR penalty is less than 0.5dB. The indication of this property of HRMIR is significant, which means it is possible to deploy this approach in hand-device, where the computation resource is limited, by setting relatively large QP_{step} value.
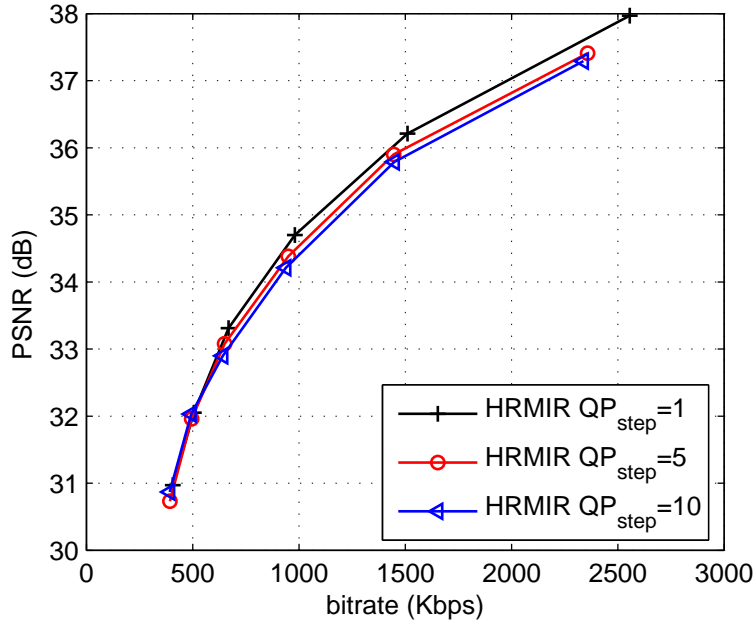
Figure 3.3: Average PSNR versus bit rate for the Foreman sequence; QP_{step} of HRMIR is set to 1, 5, 10; PLR is set to 10%, and GOP length is 30.

3.2   Experimental Results

Our simulation setting builds on the JM14.0 H.264 codec [64]. We use constrained intra prediction and CABAC for entropy coding, and fixed QP value of primary slice is used for all of our simulations. One row of macroblocks per slice is used to create slices. For each sequence, only the first frame is coded as I-frame, and the rest are coded as P-frames, the reference frame number is 1. In order to have a fair comparison with the Optimal Intra approach [16], it is assumed that the I-frame is transmitted over secure channel. We use the average luminance PSNR to assess the objective video quality, the mean squared error (MSE) is averaged over 200 trials, then the value of PSNR is calculated based on the averaged mse. A random packet-loss generator is used to drop the packets according to the required packet loss rate. For the lost slices, temporal replacement concealment is used, which means the pixel value of lost slice is copied from the same position in the previous frame. To evaluate the proposed HRMIR approach, extensive experiments have been conducted, and as benchmark, we use conventional Optimal Intra refreshment [16] and Redundant Slice Multiple Description Coding (RS-MDC) [6] for comparison.

In the first set of experiment, frame by frame average PNSR is reported for Foreman

and Bus CIF video sequences. We compare HRMIR results with Optimal Intra [16] and RS-MDC [6]. In this experiment, constant QP value is used for the primary picture. For the HRMIR approach, QP is set to 22 and 28 for Foreman and Bus respectively, while for the other two approaches, the encoded bitrate is close to but no less than the that of HRMIR approach. In Fig.3.4 full pixel accuracy motion estimation (ME) is used, whereas in Fig.3.5 motion estimation with 1/4 pixel accuracy is adopted. In both full pixel and sub-pixel motion estimation environments, the video quality of HRMIR and RS-MDC are similar at the beginning of several video frames for both the Foreman and Bus sequences. However, the video quality of RS-MDC decreases much faster than that of HRMIR; therefore, HRMIR outperforms RS-MDC significantly with frame number increasing. This result indicates that for those P-frames relatively far away from the Intra frame, only providing redundant coding is not enough to protect the video quality effectively. Meanwhile, when comparing HRMIR with Optimal Intra, for most of the frames, PSNR of HRMIR is higher than that of Optimal Intra. Another advantage of the HRMIR approach is that the video quality for each frame is more stable than the other two approaches, which is an essential characteristic of subjective high-quality video. When the encoder adopts sub-pixel ME, the accuracy of the end-to-end distortion calculated with the ROPE [16] method is compromised, and eventually, the optimal procedure in Sect.3.1.1 becomes sub-optimal. However, comparing results in Fig.3.4 with that in Fig.3.5, it is found that in both full pixel ME and sub-pixel ME environments, HRMIR outperforms Optimal Intra and RS-MDC, and the superiority of HRMIR over the other two approaches remains almost unchanged in the sub-pixel ME environment. Therefore, in the following experiments, we adopt the sub-pixel ME with the purpose of good performance in the sense of rate-distortion.

Fig.3.6 shows the video quality versus bit rates for CIF video sequences Foreman and Bus. Different QP values are selected in order to span a considerable range of coding rates. In Fig.3.6, we fix the average Packet Loss Rate (PLR) as 10%, GOP length is set to 15 and 30. It is observed that when GOP is 15, HRMIR has slight advantage over RS-MDC, whereas when the GOP is 30, HRMIR outperforms RS-MDC significantly. In Fig.3.7, we fix the GOP length as 30, PLR is set to 5% and 10%. It is interesting to see that when the PLR is 10% the superiority of HRMIR over RS-MDC is larger than the case that when average PLR is 5%. This phenomenon is because with long GOP and high packet loss rate, only providing redundant information cannot protect the video

quality properly. Furthermore, for both the Foreman and Bus sequences, the HRMIR provides much higher PSNR than Optimal Intra in all the simulation environments. Let us take the Bus sequence for example, when PLR is 5% and GOP is 30, PSNR of HRMIR is about 4 dB higher than Optimal Intra with bitrate 2 Mbps. Note that in both Fig.3.6 and Fig.3.7, when the bitrate is low, the PSNR of HRMIR and RS-MDC are nearly same, this is because in this case, very few Intra macroblocks are inserted, which makes HRMIR approach similar as RS-MDC approach. Furthermore, as the QP values of different macroblocks in the proposed HRMIR approach are not identical, additional bits are needed to encode the residual QP value.

In all the previous experiments, the channel packet loss rate is assumed to be available at the encoder, this can be implemented with the Real Time Control Protocol (RTCP) [66]. However, in practical situation, feedback packet loss rate information may be delayed from the decoder. Therefore, the packet loss rate used by the encoder in its RD optimization process may not be exactly identical to the actual packet loss rate. To further evaluate the performances of the proposed HRMIR approach at the case when the estimated packet loss rate does not match the actual one, we use 10% as packet loss rate in the RD optimization process, whereas, the actual packet loss rate is varied from 0 to 20%. In Fig.3.8, the HRMIR, Optimal Intra and RS-MDC approaches are all optimized for 10% packet loss rate. The encoded bitrate of HRMIR is 1.48 Mbps, whereas for the other two approaches, the encoded bitrate is close to but no less than the that of HRMIR approach. In the actual PLR range of [0, 20]%, the PSNR of HRMIR is the highest among the the three approaches, which means when there is PLR mismatch, the HRMIR still can provide best video quality among the three approaches. Meanwhile, the gap between HRMIR and RS-MDC increases with actual PLR; therefore, when actual packet loss rate is high, RS-MDC fails to protect the video quality properly.

In Fig.3.9 we study how Intra macroblocks are allocated in two different encoding approaches. CIF sequence Foreman is used, QP is set to 28, and the first 50 frames are used. Interestingly, the total percentage of intra macroblocks (both Intra macroblocks with and without redundant coding) increases with the PLR in both the Optimal Intra and HRMIR approaches. This can be explained in the following manner, with high packet loss rate the possibility of propagated mismatch error is high, then more Intra macroblocks are required to cut off the mismatch propagation. Meanwhile, with the

Table 3.1: Percentage of Intra macroblocks for HRMIR and Optimal Intra, QP is 28, first 50 frames are used, PLR is set to 3%, 5%, 10% and 20%.

video	apprpach	3%	5%	10%	20%
Foreman	HRMIR	0.71%	1.02%	2.14%	5.87%
	Optimal Intra	13.86%	20.31%	33.18%	48.01%
Bus	HRMIR	2.04%	3.66%	9.38%	25.61%
	Optimal Intra	53.41%	64.91%	78.07%	89.49%
Mobile	HRMIR	0.55%	0.99%	3.04%	9.59%
	Optimal Intra	26.53%	41.27%	66.72%	84.69%

same packet loss rate, the HRMIR approach allocates much less Intra macroblocks than Optimal Intra. This is because there are two tools available for error resilient coding with the HRMIR approach. Therefore, for some macroblocks, providing redundant coding leads to better usage of bitrate resource than Intra coding. More statistics information about Intra macroblock allocation can be found in Table.3.1.

Many papers [65, 67, 68] have addressed the actual network loss behavior, and most of them agree that Internet packet loss often exhibits finite temporal dependency, which means if current packet is lost, then the next packet is also likely to be lost. This leads to burst packets loss [65], the average burst length for the Internet is two. Therefore, besides i.i.d. random packet loss model, we also use burst loss model for simulation, and as indicated in [65], we set the average burst length as two. In Fig.3.10, the PSNR versus bitrate curves in burst loss environments are plotted. The results are similar with that in the i.i.d. case, and the proposed HRMIR approach can provide best video quality among the three approaches. The error resilient performance of proposed HRMIR approach is robust on different error distribution models.

3.3   Conclusions

In this section, a novel hybrid redundant macroblock and intra macroblock refreshment approach has been proposed to combat packet loss. In the proposed approach, redundant coding and/or Intra coding are optimally allocated in macroblock level. Whether to use redundant coding and/or Intra coding and the quantization parameter of the redundant coding are all determined in the end-to-end rate-distortion optimization procedure. It is worth mentioning that, in the proposed approach, only information from the previously encoded frames are used to calculate the end-to-end distortion in the

rate-distortion optimization process; therefore, no additional delay is caused, making the proposed approach suitable for real-time applications such as video conference. Extensive experimental results show that the proposed method provides better performance than other error-resilient source coding approaches. The performance gap between the proposed approach and the Optimal Intra Refreshment is huge, and in some simulation environments, the proposed approach can provide 4 dB higher PSNR than the conventional Optimal Intra Refreshment with the same bitrate.

It is worth reporting that the work reported in this section has led to the following publication:

1. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Error Resilient Video Coding with End-to-End Rate-Distortion Optimized at Macroblock Level, EURASIP Journal on Advances in Signal Processing, 2011:80, doi:10.1186/1687-6180-2011-80
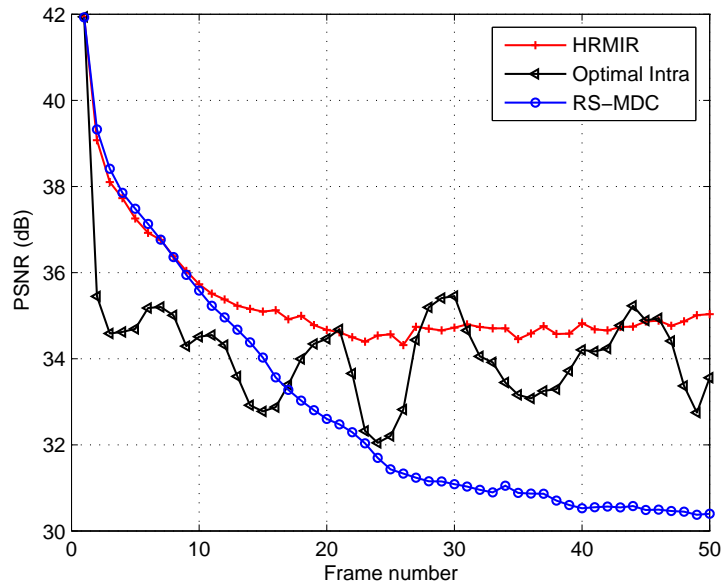
a. Foreman, full pixel motion estimation
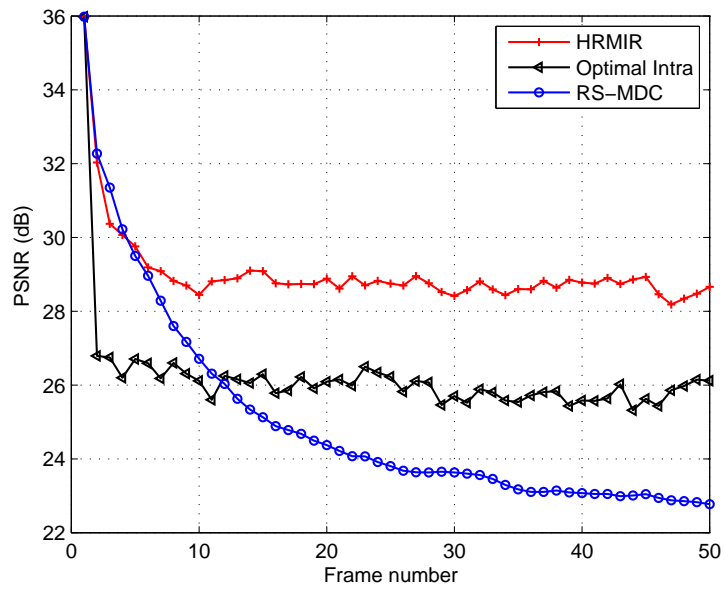
b. Bus, full pixel motion estimation

Figure 3.4: Frame by frame average PSNR comparison for HRMIR, Optimal Intra and RS-MDC; average PLR is 10%, full-pixel accuracy motion estimation; (a) Forman CIF 30 fps, 2.12 Mbps, (b) Bus CIF 30 fps, 2.88 Mbps.
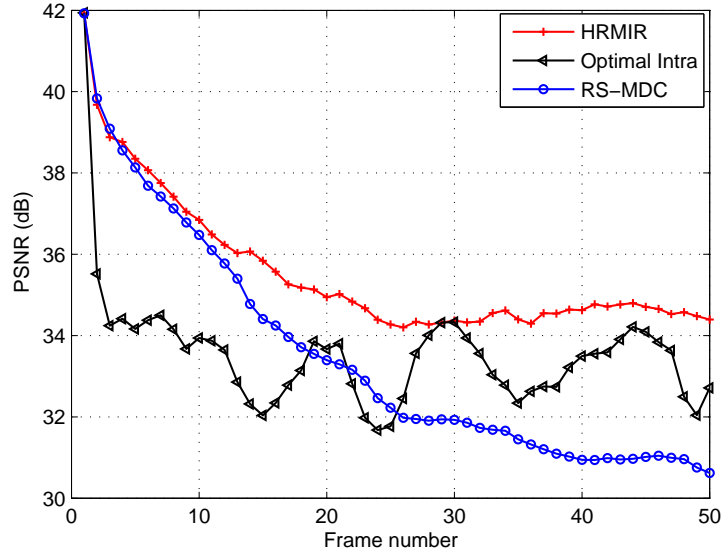
a. Foreman, 1/4-pixel motion estimation
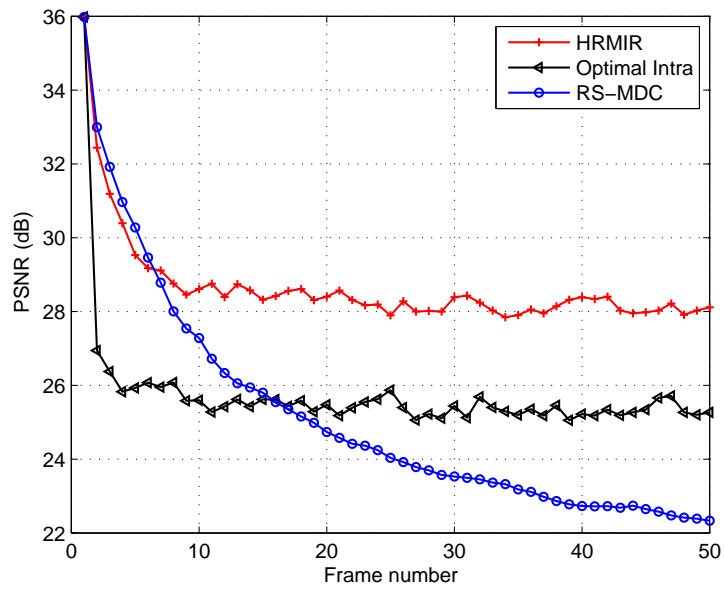
b. Bus, 1/4-pixel motion estimation

Figure 3.5: Frame by frame average PSNR comparison for HRMIR, Optimal Intra and RS-MDC, average PLR is 10%, 1/4-pixel accuracy motion estimation; (a) Foreman CIF 30 fps, 1.48 Mbps, (b) Bus CIF 30fps, 1.92 Mbps.
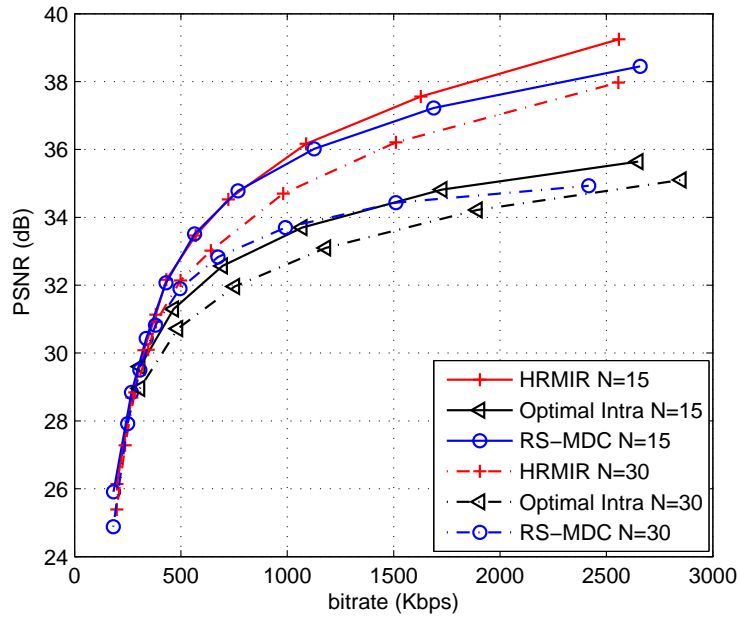
a. Foreman
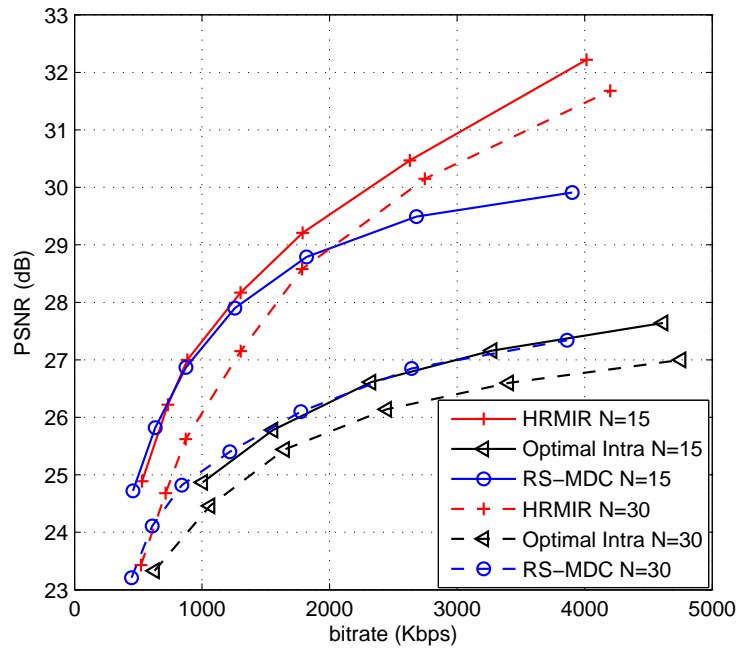
b. Bus

Figure 3.6: Average PSNR versus bit rate for HRMIR, Optimal Intra and RS-MDC; PLR is 10%; GOP length N = 15 and 30; (a) CIF Foreman sequence, (b) CIF Bus sequence.
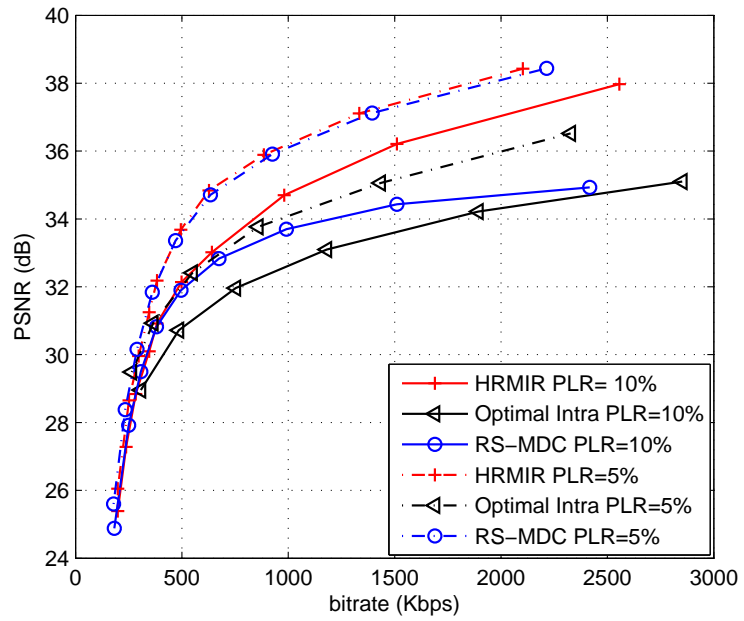
a. Foreman
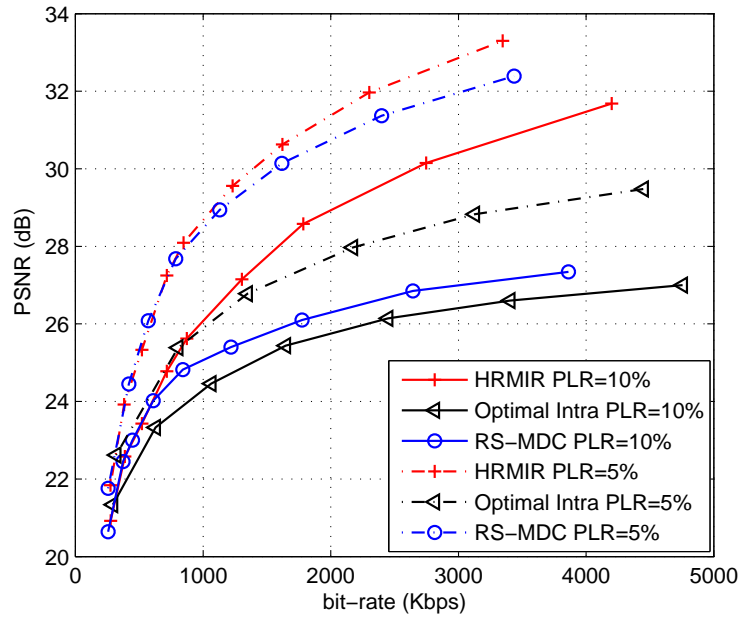
b. Bus

Figure 3.7: Average PSNR versus bit rate for HRMIR, Optimal Intra and RS-MDC; PLR is 5% and 10%; GOP length N = 30; (a) CIF Foreman sequence, (b) CIF Bus sequence.
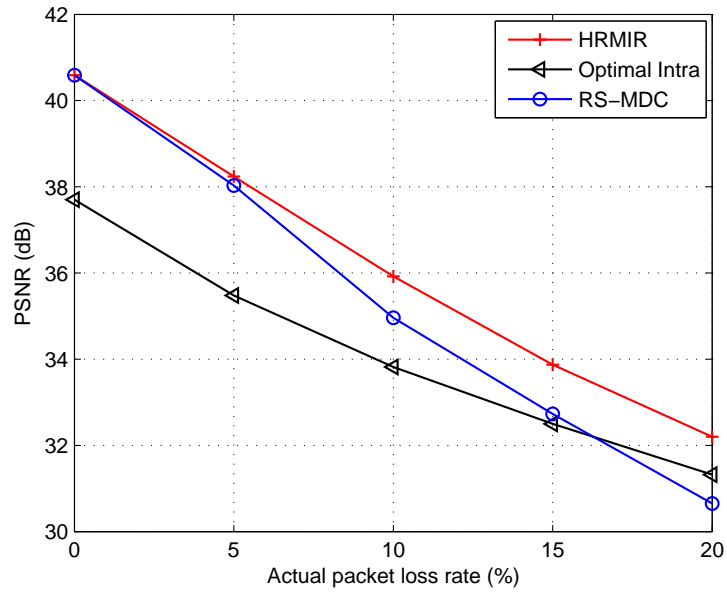
Figure 3.8: Performance comparison for HRMIR, Optimal Intra and RS-MDC when there is PLR mismatch between encoding stage and practical network situation; Foreman sequence; GOP length is 30; the estimated PLR is 10%, while the actual PLR is varied from 0 to 20%; bitrate is 1.48 Mbps.
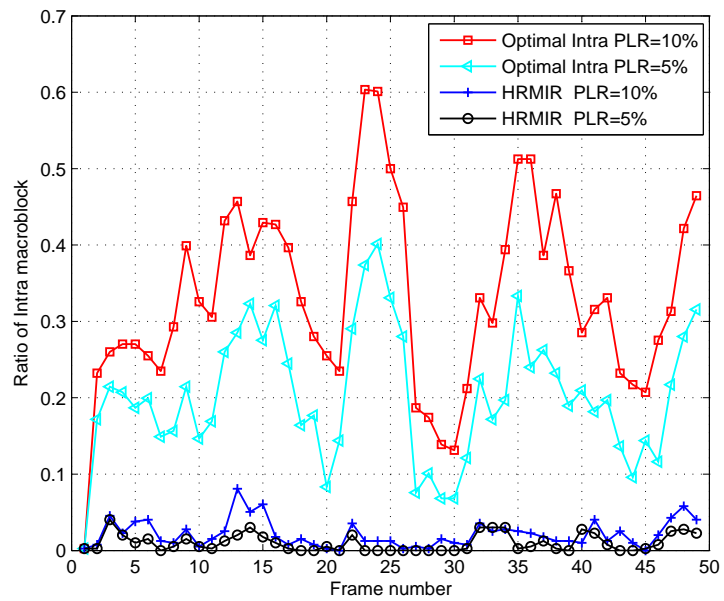
Figure 3.9: Percentage of Intra macroblock for HRMIR and Optimal Intra with PLR 5% and 10%; Foreman Sequence; QP is 28.

50
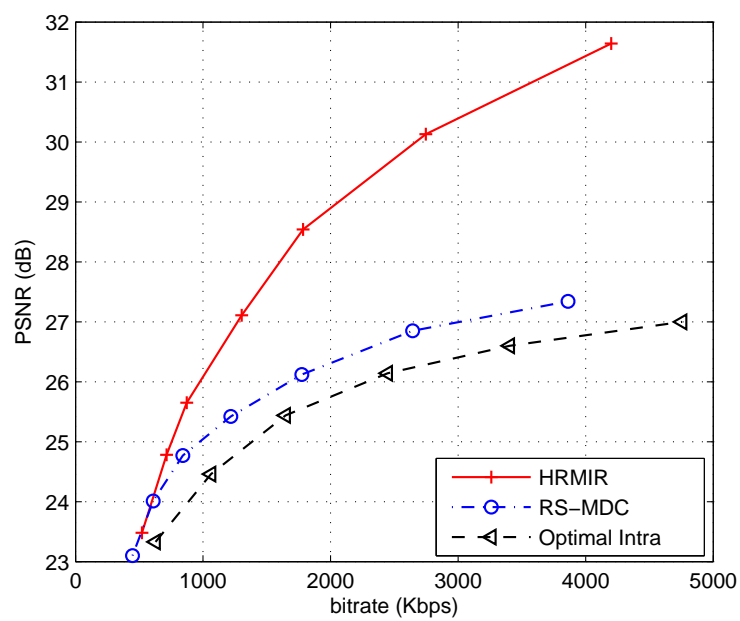
Figure 3.10: Performance comparison for HRMIR, Optimal Intra and RS-MDC when the packet loss is burst; PLR is 10%; burst length is two; Bus sequence is used; GOP length is 30.

Chapter 4

Dynamic Sub-GOP Forward Error Correction Code for Real-time Video Applications

Many error-resilient techniques have been developed to meet the requirements of video communication over lossy networks [69]. These techniques include intra macroblock (MB) refreshment [16][18], Automatic Repeat reQuest (ARQ)[70][71] and feedback-based Reference Picture Selection (RPS)[72], redundant picture coding with equal or lower quality [6][35], multiple description coding (MDC) [34][6], and Forward Error Correction (FEC) coding [56][55][1]. Among those error resilient approaches, intra macroblock refreshment, redundant picture coding and MDC cause no additional delay, making them suitable for delay constrained applications. However, for the Intra MB refreshment approach, since the coding efficiency of intra mode is typically several times lower than inter mode, the coding efficiency is compromised dramatically. For the redundant picture coding and MDC, when the redundant version is used to replace the primary one or some of the descriptions are lost during transmission, there would be mismatch error, and the mismatch error will propagate all over the GOP. ARQ and RPS usually cause long delay because of the network round-trip time, and consequently they cannot be employed for real-time applications. For the FEC approaches, the delay depends on the channel coding block size. In [56], the Reed-Solomon (RS) coding block includes the whole GOP, and one GOP of delay is caused. In [55], the RS coding block contains one block of packets (BOP) generated from different frames, and unequal loss protections are allocated for different packets based on both the frame position in the GOP and the data partition it belongs to. For this approach, one BOP of delay is caused, and the delay depends on the length of the BOP. Furthermore, as the packets

from one BOP are divided into two RS coding blocks based on the data partitioning type, the performance of the RS code is compromised. In [1], the RS code is at frame level, and no FEC coding delay is created. However, for the frame level FEC approach, usually the source packet number is not big enough for the FEC code to be efficient.

In this section, a Dynamic Sub-GOP FEC (DSGF) coding approach is proposed, and in this approach systematic Reed-Solomon erasure code is used to protect the video packets in real-time mode, while allowing to provide an error free version of the reference frame to stop the propagation error. As a block-based error correcting code, on the one hand, enlarging the block size can enhance the performance of the Reed-Solomon codes; on the other hand, large block size leads to long delay which is not tolerable for real-time video applications. In the proposed approach, in order to enlarge the RS coding block size, all frames in one sub-GOP are used as one RS coding block. The length of the sub-GOP is dynamically tuned, according to the sub-GOP position, the probability of packet loss, and other encoding parameters, so as to minimize the expected total distortion of this GOP. On the encoder side, for the systematic RS code, the data is left unchanged and the parity packets are appended; therefore, there is no encoding delay. Meanwhile, at the receiver end, to decode and display one frame in the sub-GOP, the video decoder only needs packets belonging to this frame. If some packets of this frame get lost during transmission, error concealment is applied to conceal the lost packets. In this manner, the decoder does not need to wait for all the packets belonging to this sub-GOP. Therefore, there is no delay on the decoder side. Later, when the transmission of all packets of this sub-GOP is finished, the systematic RS decoder would try to recover the lost packets. If enough packets are received, the RS decoder will be able to recover all the lost packets of this sub-GOP, and the video decoder will re-decode this sub-GOP with all the received and recovered packets, updating the reference frame, so the concealment distortion would not propagate to later frames.

The rest of this section is organized as follows. A brief review of systematic RS code is provided in Section 4.1. In Section 4.2, firstly the frame level Evenly FEC approach is introduced; this approach is used as a benchmark for the real-time FEC coding. Later, the proposed DSGF approach is presented in detail. In Section 4.3 some simulation results validating the proposed approach are given. Finally, some conclusions are drawn in Section 4.4.

4.1 Systematic Reed-Solomon Erasure Code

In this section, we will briefly recall some concepts and notations about systematic Reed-Solomon (RS) erasure code, which will be used for the DSGF approach. The systematic RS erasure code has been widely used as FEC code to protect data packets against losses in packet erasure networks. In RS (N, K) code, for every K source packets, $(N - K)$ parity packets are introduced to make up a codeword of packets. As long as a client receives at least K out of the N packets, it can recover all the source packets. If the received packet number is less than K, the received source packets can still be used, because they have been kept intact by the systematic RS encoding process. In general, for the same code rate K/N, increasing the value of K would enhance the performance of RS code.

One important parameter for the systematic RS code, that we will need is the remaining packet loss rate after the RS correction, p'. So, for example, for the Bernoulli i.i.d. packet loss model, this parameter is determined by the value of N, K, and the average network packet loss rate, p. For the Markov burst packet loss model, p' is also impacted by other parameters of the Markov model, i.e., the average burst loss length.

4.1.1 Bernoulli i.i.d. Packet Loss Model

In the Bernoulli i.i.d. packet loss model, p' could be evaluated as

$$p' = \frac{\sum_{i=1}^{K} i\ p_{rs}(i)}{K} \tag{4.1}$$

with $p_{rs}(i)$ representing the probability of still having i unrecoverable source packets after RS correction, from now on we will refer to those packets as unrecoverable lost packets. To evaluate $p_{rs}(i)$, let us use $p_s(n)$ and $p_r(n)$ to denote the probability of losing n packets before decoding the RS code among the source packets and parity packets, respectively.

$$p_s(n) = \binom{K}{n}(1-p)^{K-n}p^n \tag{4.2}$$

$$p_r(n) = \binom{N-K}{n}(1-p)^{N-K-n}p^n \tag{4.3}$$

Since having i unrecoverable lost packets is caused by losing i source packets, and at the same time losing more than $N - K - i$ RS parity packets, then the probability of this event is

$$p_{rs}(i) = \begin{cases} p_s(i)\ P_r(N-K-i+1) & \text{for } i \leq N - K \\ p_s(i) & \text{for } i > N - K \end{cases} \tag{4.4}$$

where $P_r(j)$ is used to denote that no less than j RS parity packets are lost, this could be evaluated based on Equation (4.3) as

$$P_r(j) = \sum_{n=j}^{N-K} p_r(n) \tag{4.5}$$

4.1.2 Burst Packet Loss Model

For the burst loss model, we will use the Gilbert two-state model because it is one of the most common models used for multimedia transmission simulation. For this model, the formula to calculate the remaining packet loss rate, p', after the RS correction was presented in [73]. It is worth noticing that, in Gilbert two-state model, p' is not only determined by the value of N, K and p, but also influenced by the average burst length, and p' can be evaluated as follows:

$$
\begin{aligned}
p' = & \frac{p}{K} \sum_{i=1}^{K} iR(i,K) \sum_{j=max(0,N-K+1-i)}^{N-K} R(j+1, N-K+1) \\
& \frac{1-p}{K} \sum_{i=1}^{K-1} (K-i)S(i,K) \sum_{j=0}^{K-i-1} S(j+1, N-K+1)
\end{aligned}
\tag{4.6}
$$

where $R(m,n)$ denotes the probability that $m-1$ consecutive packet losses occur following a packet loss, and $S(m,n)$ denotes the probability that $m-1$ consecutive packets arrive following one packet arrival. For the detailed procedure of calculating $R(m,n)$ and $S(m,n)$, please refer to [73].

4.2 Real-Time FEC Video Transmission Approaches

Since our objective is to design FEC video transmission system for real-time applications while minimizing the delay caused by the encoding stage; therefore B-frame will not be used, so we will use the IPPP GOP structure. It is also important to note that, the most commonly used applications for real-time system are video telephony with low latency requirements. This application typically uses the baseline profile of H.264/AVC, where only I-frames and P-frames are used [10]. To make the RS code efficient, fixed length slice scheme, in terms of byte, is used to create slices. The slice length is decided by the Maximum Transmission Unit (MTU) of the underlying networks. With this method, as many MBs are put into one slice as possible under the constraint that the slice length is no more than the target length; therefore, the length
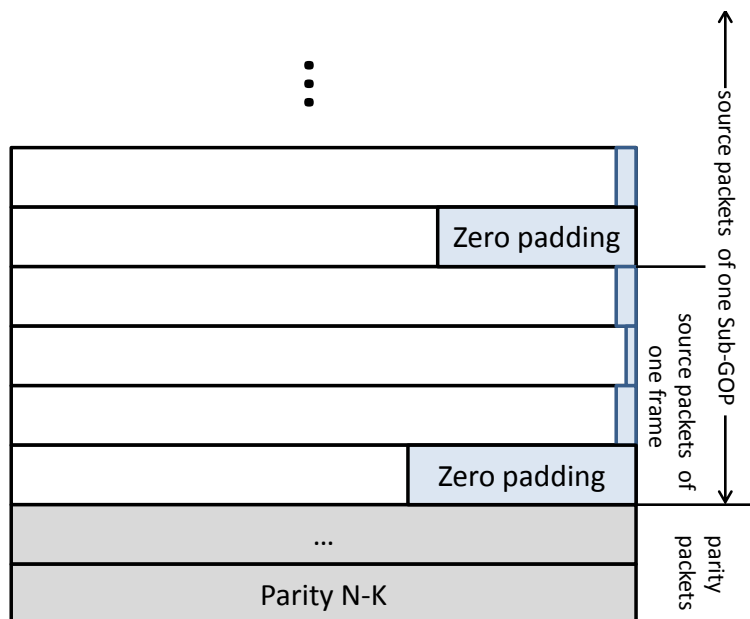
Figure 4.1: Packet padding method for the RS coding; H.264 fixed slice length method is used; the slice length is nearly same except for the last one.

of all the slices except the last ones in each frame is very close to the target slice length. As shown in Fig.4.1, for the slices other than the last slice, only very few zero bytes are padded, whereas for the last slice of one frame, usually more dummy zero bytes are padded. The length of each RS protection packet (i.e., parity packet) is the same as the target slice length. In this section, the term packet and slice are used interchangeably, as one packet per slice packetization method is adopted.

4.2.1 Frame-Level Evenly FEC

For real-time FEC video packet protection, one common approach is to perform RS coding in frame level, which means that the RS coding block contains data packets from the same video frame. Under this constraint, RS coding does not introduce any additional delay. Let us assume the GOP length is L frames, and the i-th frame has $K(i)$ source packets and $R(i)$ RS parity packets. If we want to provide even protection for all the GOP frames, and taking into account that, in general, $K(i)$ does not change largely, then $R(i)/K(i)$ needs to be almost constant over all the GOP's frames. However, taking into account that $R(i)$ should be an integer and that $K(i)$

Table 4.1: The remaining packet loss rate after RS code correction with $\mu = 0.2$, where RS coding block size is $K = 5, 10, 15, 20, 30$, network packet loss rate is $p = 5\%, 10\%, 15\%$.

p	$K = 5$	$K = 10$	$K = 15$	$K = 20$	$K = 30$
5%	1.13%	0.51%	0.25%	0.13%	0.04%
10%	4.10%	3.03%	2.38%	1.93%	1.32%
15%	8.34%	7.62%	7.20%	6.91%	6.47%

may vary from frame to frame, so we can write $R(i)$ as

$$
R(i) = \begin{cases} \lceil \mu K(1) \rceil & \text{if } i == 1 \\ \left\lceil \mu \sum_{k=1}^{i} K(k) \right\rceil - \sum_{k=1}^{i-1} R(k) & \text{if } i > 1 \end{cases} \tag{4.7}
$$

where $\mu = (N - K)/K$ is the parity packet rate of RS coding, and operation $\lceil X \rceil$ is used to get the minimum integer number greater than or equal to X. From now on we name this approach as Evenly FEC.

4.2.2 Dynamic sub-GOP FEC Coding

For a probability of packet loss p, and transmitting packet number $N = K + \mu K$, to be able to recover pN losses, we need to insert μK redundant packets, with $\mu K \geq pN$ or equivalently $\mu \geq \frac{p}{1-p}$. In the limit case, according to the law of large number, when $N \to \infty$, then μK could be as small as pN, namely $\mu K \approx pN$, which means that the inserted redundancy could be as small as $\mu = \frac{p}{1-p}$. In practical situations, $N \to \infty$ is impossible, in this case with the same parity packet rate $\mu = (N - K)/K$, the larger the value of K is, the higher the performance of RS code can be. Table.4.1 lists the remaining packet loss rate, p', after the RS code correction, for different values of K. This table demonstrates that for the same packet loss rate and redundancy, the smaller the RS coding block the lower the performance of the RS codes.

Motivated by this fact, we propose to encompass packets from a sub-GOP of frames to one RS coding block to enlarge the value of K. Fig.4.2 shows one example of how to generate sub-GOPs and allocate RS parity packets at the end of each sub-GOP. In addition, in order to meet the real-time constraint, we use the systematic RS code, so the source packets are intact in the RS coding process. Therefore, at the receiver side, the video decoder only needs packets belonging to one frame to decode and display that frame. If some packets of this frame are lost during transmission, error concealment is used to conceal them.
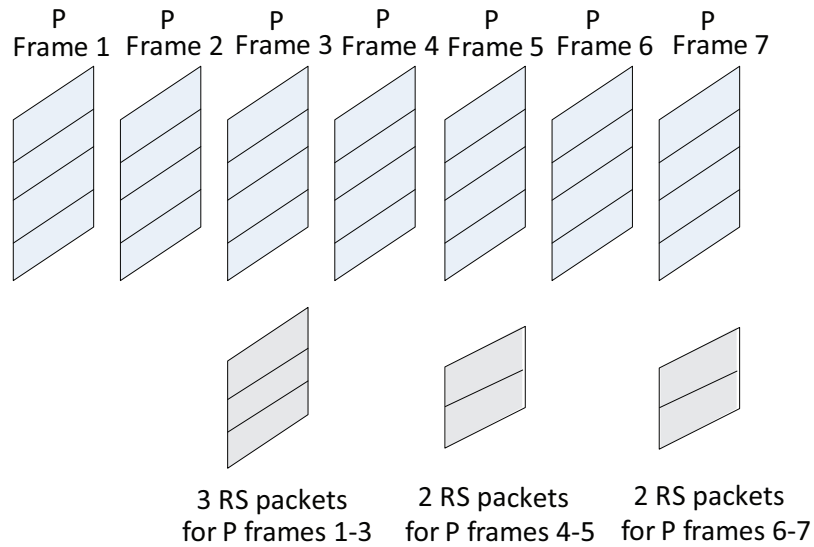
Figure 4.2: One example of RS parity packets allocation for the dynamic sub-GOP FEC coding approach.

For easy illustration, let us take one example, we use the second sub-GOP in Fig.4.2, i.e., Frames four and five. In this case, the sub-GOP contains two frames and each frame generates four packets. The redundancy due to the RS coding is 25%, which means for the eight source packets of this sub-GOP, $(10, 8)$ RS code is applied. In one sub-GOP, when the first frame is encoded by the video encoder, immediately the encoded four packets are transmitted over the network. Due to network failure, let us assume two packets among these four packets are dropped. Then upon receiving the two packets, the video decoder will decode and display this frame, and for the lost packets, error concealment is used. In this manner, no additional delay is introduced. Later, the second frame is encoded, generating another four source packets, meanwhile, as the second frame is the last frame in this sub-GOP, two RS parity packets are generated for the eight source packets of this sub-GOP. Suppose this time the four source packets and the two RS parity packets successfully arrive at the receiver side without any loss. In this case, at the receiver side, the $(10, 8)$ RS decoder will be able to recover the two lost packets, so the video decoder will re-decode the first frame of this sub-GOP with the two received source packets and the two recovered packets, and update the reference frame buffer. In this case there will be no distortion in the second frame, no distortion would propagate to the incoming frames. However, if the Evenly FEC approach is adopted, with the same amount of redundancy, for each frame $(5, 4)$ RS

code will be used. The RS code would not be able to recover the two lost packets of the first frame, eventually the concealment distortion propagates to all the following frames, and severely degrades the video quality at the receiver side. In this illustrative simplified example, both the Evenly FEC and DSGF approach use the same amount of RS parity packets, and in both cases, no additional delay is introduced. However, the advantage of the DSGF approach is obvious, because this approach is able to restrict concealment distortion in very few frames.

4.2.3 Optimal sub-GOP Size and RS Packet Allocation

As described above, using the RS protection in sub-GOP level could be better than in frame level, but the problem of how to divide frames into sub-GOPs and how to allocate the RS parity packets among all the sub-GOPs will be addressed in the following. To tackle this problem we have to note that, on the one hand, if the sub-GOP includes too few frames, the value of K for the RS code will not be large enough to make the RS code efficient. On the other hand, if the sub-GOP includes too many frames, as the RS correction codes will not be available until the last frame of this sub-GOP, the quality of those frames before the last frame would degrade significantly. Consequently, the sub-GOP length should be properly tuned to increase the efficiency.

In general, I-frame generates much more bits than P-frame, and therefore more source packets are produced for I-frame. In our DSGF approach, for the I-frame we provide RS protection in frame level, the same as Evenly FEC approach, whereas for the P-frame we allocate RS parity packets in sub-GOP level. Our objective is to optimally allocate sub-GOP and RS parity packets and minimize the expected total distortion of this GOP. To do the optimal allocation we need to know the detailed information of this GOP, including the slice number in each frame, the concealment distortion caused by losing each slice, and how the distortion propagates. However, those information are not available for real-time on-the-fly transmission system. In light of such circumstance, we established a model to represent these information. The model parameters include the number of P-frames in one GOP, L, and the average number of slices in each P-frame, S. For simplicity, let us assume the value of S is unchanged, this assumption is more accurate for low motion video with little change in the content. The expected concealment distortion of losing one packet is $\bar{d}$, the distortion in current frame propagates to the following frames, and the attenuation

function of the distortion is $f(n)$. This means if the concealment distortion of one slice is $\bar{d}$, it will propagate to the following frames and become $f(n)\bar{d}$ after n frames. For the sake of simplicity the function $f(n) = \alpha^{n-1}$ ($0 < \alpha < 1$) is employed, this expression approximates, at low levels of attenuation, the function $f(n) = (1 + \lambda_1 n)^{-1}$ and $f(n) = (1 + \lambda_2 n)^{-1/2}$ reported in [74] and [75], respectively, α, λ_1 and λ_2 being parameters to be selected. Let us assume that distortion caused by losing slices are uncorrelated, and in this case, the total expected distortion for the whole GOP is the sum of all the expected distortions caused by individual slices. The assumption on slice concealment distortion uncorrelation is reasonable. In fact, concealment distortions, can be considered as uncorrelated with the pixel values, then concealment distortions caused by losing different slices can also be considered as uncorrelated. The additive distortion model has been verified experimentally in [6]. For the P-frames in one GOP the total allocated RS parity packet number is $R = \mu SL$, here μ is the parity packet rate of RS coding. We use $R(i)$ to denote the number of RS parity packet for P-frame i.

$$\sum_{i=1}^{L} R(i) \leq R \tag{4.8}$$

Fig.4.2 shows one example of how RS parity packets are allocated. We assume there are totally t positions where we insert RS parity packets, with frame number r_1, r_2,..., r_t, whereas other positions have no RS parity packets. The number of RS parity packets are $R(r_1)$,$R(r_2)$,..., $R(r_t)$. In the example of Fig.4.2, we allocate RS parity packets in the three positions ($t = 3$), the 3 positions are $r_1 = 3$, $r_2 = 5$, $r_3 = 7$, and RS packet number is $R(r_1) = 3$, $R(r_2) = 2$, $R(r_3) = 2$. The RS parity packets allocated under the Frame r_{m+1} are used to protect the frames from $r_m + 1$ to r_{m+1}. Therefore, the parameters of RS (N, K) code for this sub-GOP are $N = (r_{m+1} - r_m)S + R(r_{m+1})$ and $K = (r_{m+1} - r_m)S$. We use $\bar{D}(r_m + 1, r_{m+1})$ to denote the expected distortion caused by losing packets from Frame $r_m + 1$ to r_{m+1}. It is important to note that $\bar{D}(r_m + 1, r_{m+1})$ does not only include the distortion in frames from $r_m + 1$ to r_{m+1}, but also accounts for the propagated distortion in subsequent frames:

$$\bar{D}(r_m + 1, r_{m+1}) = \bar{D}_i(r_m + 1, r_{m+1} - 1) + \bar{D}_p(r_{m+1}) \tag{4.9}$$

As described in Equation (4.9), the distortion $\bar{D}(r_m + 1, r_{m+1})$ is caused by two parts: $\bar{D}_i(r_m + 1, r_{m+1} - 1)$ is the expected distortion within frames $r_m + 1$ to $r_{m+1} - 1$. For those frames, the recovering capability of RS code cannot be used by the time when

those frames are decoded and displayed. The subscript i means that $\bar{D}_i(r_m+1, r_{m+1}-1)$ accounts for the *internal* distortion and their propagated distortion only within frames $r_m + 1$ to $r_{m+1} - 1$ due to the eventual losses in the frames. The error propagation to frames outside this sub-GOP is not accounted in this term. $\bar{D}_p(r_{m+1})$ is the sum of expected distortion in Frame r_{m+1} and it also account for the propagated distortion in the subsequent frames. The subscript p denotes that this term includes the *propagated* distortion to frames beyond this sub-GOP. Note that by the time of decoding and displaying Frame r_{m+1}, the RS code would try to recover the lost source packets within this sub-GOP, but when the packet loss number in this sub-GOP is beyond the recovery capability of RS code, the RS code would not be able to recover the lost packets.

For the term $\bar{D}_i(r_m + 1, r_{m+1} - 1)$, as from Frame $r_m + 1$ to $r_{m+1} - 1$ the expected number of lost packets for each frame is pS, and by taking the error propagation inside this sub-GOP into consideration, we get

$$\bar{D}_i(r_m + 1, r_{m+1} - 1) = \sum_{i=1}^{r_{m+1}-r_m-1} \phi(i) \, p \, S \, \bar{d} \tag{4.10}$$

where $\phi(i) = \sum_{n=0}^{i-1} f(n)$.

Now let us evaluate the term $\bar{D}_p(r_{m+1})$, to do this we have to note that from Frame $r_m + 1$ to Frame r_{m+1} there are $(r_{m+1} - r_m)S$ source packets, while the RS parity packet number is $R(r_{m+1})$. Firstly, let us evaluate the expected distortion in Frame r_{m+1}, $\bar{D}_l(r_{m+1})$, taking into account the error propagation from previous frames inside this sub-GOP:

$$\bar{D}_l(r_{m+1}) = \frac{\bar{\Gamma}}{r_{m+1} - r_m} \phi(r_{m+1} - r_m)\bar{d} \tag{4.11}$$

with $\bar{\Gamma}$ being the expected number of unrecoverable lost packets among Frame $r_m + 1$ to Frame r_{m+1}, and this is $\bar{\Gamma} = (r_{m+1} - r_m)p'S$, where p' is the remaining packet loss rate after the RS correction. The detailed process to calculate p' has been given in Section 4.1 for both i.i.d. and burst model. Therefore, the expected number of unrecoverable lost packets in each frame would be $\bar{\Gamma}/(r_{m+1} - r_m)$. Given that the distortion in Frame r_{m+1} will propagate to the end of this GOP; therefore

$$\bar{D}_p(r_{m+1}) = \bar{D}_l(r_{m+1})\phi(L - r_{m+1} + 1) \tag{4.12}$$

At this point, $\bar{D}_i(r_m + 1, r_{m+1} - 1)$ and $\bar{D}_p(r_{m+1})$ can be used to calculate the expected distortion caused by each sub-GOP using Equation (4.9). By adding up the

expected distortion caused by each sub-GOP, we evaluate the total expected distortion for the whole GOP, $\bar{D}_{total}$, as in Equation (4.13)

$$\bar{D}_{total} = \begin{cases} \bar{D}(1, i_1) + \sum_{m=1}^{t-1} & \bar{D}(r_m + 1, r_{m+1}) \\ & \text{for } r_t \equiv L \\ \bar{D}(1, i_1) + \sum_{m=1}^{t-1} & \bar{D}(r_m + 1, r_{m+1}) + \bar{D}_i(r_t + 1, L) \\ & \text{for } r_t < L \end{cases} \tag{4.13}$$

where condition $r_t \equiv L$ means that the last sub-GOP has RS parity packets, whereas $r_t < L$ means there are no RS parity packets for it.

Finally, the optimization problem can be formulated as the following constrained minimization:

$$\begin{cases} \min & \bar{D}_{total} \\ \text{subject to} & \sum_{i=1}^{N} R(i) \leq R \end{cases} \tag{4.14}$$

4.2.4 Greedy Algorithm for Fast RS Parity Packet Allocation

It is computational prohibitive to get the global optimal solution for Equation (4.14). When one GOP includes L P-frames, and the RS parity packet number for P-frames is R, there are totally $\binom{L+R-1}{R}$ possible allocation solutions. For example, if the GOP length is $L = 30$ and the number of RS parity packet is $R = 40$, there would be $\binom{69}{40} = 2.39 \times 10^{19}$ possible allocation solutions. Obviously, calculating the value of D_{total} using Equation (4.13) for all the 2.39×10^{19} allocation patterns is impossible. Since it is computational prohibitive to get the global optimal solution, we propose to use a greedy algorithm to get a sub-optimal RS parity packet allocation. In this algorithm, each time one RS parity packet is allocated, by trying to allocate this packet for all possible L positions, while calculating the value of D_{total} using Equation (4.13) for all these positions, then the algorithm choose to allocate the RS packet to the position which makes D_{total} minimum. Suppose adding the RS parity to Frame j would make D_{total} minimum, then $R(j) = R(j) + 1$. By iterating the previous steps R times, all the parity packets will be allocated. With this greedy algorithm, allocating R RS parity packets to L P-frames will have a computational complexity order of $O(RL)$, which is much less than $\binom{L+R-1}{R}$. The detailed procedure of greedy RS parity packet allocation algorithm is shown in Algorithm.2.

Fig.4.3 shows two practical examples of how to divide P-frames into sub-GOP and allocate the RS parity packets among all the sub-GOPs with the greedy algorithm. These results have been obtained by assuming that one GOP has 30 P-frames, each P-frame includes 5 or 10 slices, the value of α is 1, the packet loss rate is 5% with i.i.d.

Algorithm 2 Greedy RS parity packets allocation algorithm
$$\begin{aligned}
&\textbf{for } i = 1 \text{ to L } \textbf{do} \\
&\quad R(i) \Leftarrow 0 \\
&\textbf{end for} \\
&\textbf{for } j = 1 \text{ to R } \textbf{do} \\
&\quad index \Leftarrow 0 \\
&\quad distortion \Leftarrow \infty \\
&\quad \textbf{for } i = 1 \text{ to L } \textbf{do} \\
&\quad\quad R(i) \Leftarrow R(i) + 1 \\
&\quad\quad \text{calculate } D_{total} \text{ using Equation (4.13)} \\
&\quad\quad R(i) \Leftarrow R(i) - 1 \\
&\quad\quad \textbf{if } D_{total} \leq distortion \textbf{ then} \\
&\quad\quad\quad index \Leftarrow i \\
&\quad\quad\quad distortion \Leftarrow D_{total} \\
&\quad\quad \textbf{end if} \\
&\quad \textbf{end for} \\
&\quad R(index) \Leftarrow R(index) + 1 \\
&\textbf{end for}
\end{aligned}$$

model, the parity packet rate is 20%. It is interesting to find some regular patterns behind the allocations. Firstly, in general, the P-frames at the beginning of the GOP have more RS parity packets than those at the end of the GOP. In Fig.4.3.(a), the first 2 sub-GOPs have 4 RS parity packets for each sub-GOP, the subsequent sub-GOP has 3 RS parity packets, and so on. For the last frame in the GOP, no RS parity packets are allocated. This is because any distortion in the front frames will propagate to the following frames, and usually losing one packet in the front frame would lead to more distortion for the whole GOP than losing one in the end. Therefore, it is reasonable to allocate more RS parity packets to the frames at the beginning of GOP. Secondly, it is important to note that at the beginning of the GOP, one sub-GOP usually contains more frames than the sub-GOP in the end of the GOP. In Fig.4.3.(a), the first 8 sub-GOPs include 3 frames, the 9th sub-GOP contains 2 frames, while the 10th and 11th sub-GOPs contain only one frame. This is also because the distortion propagation paths in the frames at the beginning of a GOP are long. So putting more frames into one sub-GOP can make the value of K large, which means that the RS code can recover the lost packets with higher probability, and eventually effectively cut down the error propagation. Thirdly, comparing results in Fig.4.3.(a) with Fig.4.3.(b), the average sub-GOP length in Fig.4.3.(a) is larger than that in Fig.4.3.(b). This is because the number of slices in each frame, S, is large in Fig.4.3.(b), and there is no need to put as many frames into one sub-GOP as in Fig.4.3.(a).

4.2.5 Adaptive Estimation of Attenuation Factor (α)

As described above, in the sub-GOP and RS packet allocation process, we need to know the distortion attenuation function $f(n) = \alpha^{n-1}$ $(0 < \alpha < 1)$. However, for various video sequences the attenuation factor α is different, at the same time, for one specific video sequence the parameter α changes with time. Therefore one possible solution to determine α would be to adaptively estimate its value at the end of each GOP, and to use this α for the next GOP. The detailed process is as follows. While encoding the GOP, its slices will be decoded, and only one slice is assumed to be lost during the decoding process. So if we assume this slice is in Frame k, then, and because of the error propagation, the distortion due to this emulated loss will be $d(k)$, $d(k + 1)$, $d(k+2)$, ..., $d(L)$, here L is the length of the GOP. These distortions could be obtained by comparing the decoded sequence with the emulated loss with the error free version. At the end of the GOP, α could be evaluated as $\frac{1}{L-k}\sum_{i=k+1}^{L}\frac{d(i)}{d(i-1)}$. Given the fact that the attenuation factor changes slowly, the estimated α could be used to do the RS allocation of the following GOP. And the value of α could be updated at the end of each GOP by running the previous procedure for each current GOP. More accurate estimation of α could be obtained by emulating the loss of more than one slice.

4.3 Experimental Results

Our simulation setting is built on the JM14.0 [64] H.264 codec. CIF video sequence Foreman, Bus and Stefan are used for the simulations. We select these three sequences, because they represent different motion characteristics, Foreman has moderate movement and video texture, Bus has fast and translational movement, whereas Stefan has fast movement with different motion directions. The GOP structure is IPPP, and GOP length is 30 frames. The reference frame number is 1, in other words, only the previous frame is used for prediction. One slice is transmitted in one packet, taking the MTU of wireless network into account, we set the target slice length as 400 byte. We use the average luminance PSNR to assess the objective video quality, which is obtained by evaluating the Mean Squared Error (MSE) over all the frames and over 200 trials, then the average PSNR is calculated based on the averaged mse. In order to have a fair comparison, we compare the proposed DSGF approach with RS-MDC [6] and Evenly FEC approach, because all those approaches meet the real-time constraint and cause

no additional delay. In the following simulations, we assume the packet loss follows the i.i.d. model, unless burst packet loss model is explicitly specified. In Fig.4.4, we compare the effects of using adaptive α value with fixed α value. For the fixed α case, we choose $\alpha = 1$, which means the distortion will propagate without attenuation. It is interesting to note the gap between the two curves is rather narrow (always less than 0.5dB), especially in high bitrate. Taking into consideration the huge computational resource for adaptively estimating the α value, we use $\alpha = 1$ in all the following simulations for simplicity. It is important to point out that with the adaptive α estimation method described in Sect.4.2.5, the performance of the proposed approach is expected to be further enhanced.

In the first set of simulations, we study the effects of allocating different parity packet rates for RS code. The network packet loss is an i.i.d. random process; for the same packet loss rate $p = 5\%$, we try different μ, including 15%, 20%, 25% and 30%. We do simulations with various quantization parameters (QP) to span a considerable bitrate range. Fig.4.5 shows the PSNR versus bitrate with different RS parity packet rates μ. The PSNR curve for parity packet rate 15% is much lower than other cases in all bitrate. In intermediate and high bitrate, the PSNR curves for $\mu = 20\%$, 25% and 30% are very close, while in low bitrate, higher redundant rate can provide better PSNR. This is because in low bitrate, the slice number in each frame is small, which makes the performance of RS low, and high RS parity packet rate is required to compensate this. In general, the PSNR curves for redundant rate 20%, 25% and 30% are similar, consequently, in the following simulations, we use parity packet rates 20% for 5% packet loss. Similarly, in later simulations, 40% RS parity packet rate is used for 10% packet loss.

Fig.4.6 and Fig.4.7 compare the performance of different approaches in terms of PSNR versus bitrate and for random packet loss rate 5% and 10%, respetively. As mentioned above, for DSFG and Evenly FEC for the packet loss rate 5% and 10%, the RS coding redundancy is 20% and 40%, respectively, whereas for the RS-MDC, the redundancy is tuned as described in [6]. The proposed DSGF approach always outperforms RS-MDC and Evenly FEC in all the simulation environments. For all the Foreman, Bus and Stefan sequences, the gain over Evenly FEC can be more than 2 dB in low bitrate when packet loss rate is 5%, and the gain over RS-MDC could be over 4 dB in high bitrate when packet loss rate is 10%. It is very interesting to note

that the gap between the DSGF and Evenly FEC is larger in low bitrate than in high bitrate. This is because in high bitrate, more packets are generated for each frame, and consequently, for the frame level Evenly FEC approach, the value of K is relatively large to make the RS coding efficient.

In Fig.4.8, for the Foreman and Stefan sequences, the PSNR of each frame in one GOP are plotted. In each sub-GOP, the video quality degrades frame by frame gradually because of the random packet loss. However, at the end of each sub-GOP, with high probability, the RS parity packets will be able to recover all the lost packets of this sub-GOP, so the PSNR of the last frame of each sub-GOP is higher than other frames in this sub-GOP. All these factors make the video frame PSNR fluctuate, with a period same as the sub-GOP length. Nevertheless, for the majority of the frames in one GOP, PSNR of the proposed approach is higher than that of Evenly FEC approach and RS-MDC. In fact, among the 30 frames, only 6 and 3 frames have PSNR lower than that of the Evenly FEC approach for the Foreman and Stefan sequences, respectively; almost all frames have better video quality than RS-MDC, although RS-MDC has some extra bitrate. It is worth noticing that for some video frames, PSNR of the proposed approach is more than 3 dB higher than that of the Evenly FEC approach, and for the second half of the GOP, our approach outperforms the Evenly FEC approach and RS-MDC significantly. Note that for the first frame in this GOP, which is I-frame, the video quality of the proposed approach and Evenly FEC approach is same, more than 0.8 dB better than RS-MDC. This is because, the slice number in the I-frame is large, and that makes the RS code efficient, thereby providing higher PSNR than RS-MDC. It is worth noticing that similar results have been obtained for the Bus sequence. With the proposed approach, although the video quality rises and falls, this would not lead to inferior visual perception. To demonstrate that, two consecutive video frames after random packet loss are provided in Fig.4.9 for the Foreman sequence, with the $10th$ frame has 38.27 dB and the $11th$ frame has 35.20 dB of PSNR. Despite of 3 dB PSNR gap, human eyes can hardly distinguish this quality fluctuation. In order to better visualize the video quality, the most damaged area in Frame $11th$ is zoomed in. We select this area because the slice that covers this area is lost, and concealment process is invoked, moreover, there is some motion in this area. However, even in this area, as we can see, the video quality is still acceptable.

In all the previous experiments, the channel packet loss rate is assumed to be

available at the video transmitter side, this can be implemented with the Real Time Control Protocol (RTCP) [66]. However, in practical situations, feedback packet loss rate information may be delayed from the video receiver. Therefore, the packet loss rate may not be exactly identical to the actual packet loss rate. To further evaluate the performances of the proposed DSGF approach in this scenario, we assume the packet loss rate is 10%; therefore the redundancy of the RS-MDC is tailored for 10% packet loss rate, as proposed in [6]. This will lead to 36.5% of redundancy with QP 26 and GOP length 30, and it is worth noticing that, this amount of redundancy is optimal for RS-MDC for this specific transmission scenario. In order to have fair comparison with other approaches, this amount of redundancy and the same QP has been used for Evenly FEC and the DSGF approach, this will generate the same total bitrate for the three approaches. The actual packet loss rate is varied from 0 to 20%. In Fig.4.10, the video qualities of the three approaches under different packet loss rates are plotted. As we can see, the PSNR of the proposed DSGF approach is the highest among the three approaches, which means when there is packet loss rate fluctuation, the proposed approach can still provide the best video quality. Meanwhile, the gap between the DSGF approach and Evenly FEC increases with packet loss rate, that is because in this case the RS parity packet rate μ is fixed, increasing the packet loss rate makes the redundancy relatively small comparing to the packet loss rate, and in this case, it becomes more important to group frames together in order to increase the efficiency of RS coding.

In order to validate the performance of the proposed DSGF approach in different error distribution models, some additional results are provided for real Internet packet loss pattern and Gilbert burst loss pattern. In Fig.4.11, the average PSNR versus bitrate for real Internet environments are presented, for the Foreman and Stefan sequences. The packet loss pattern for the Internet experiments specified in the file 10 of Q15-I-16r [76] is used to emulate the real Internet environments. This has an actual packet loss rate of 11.38%. From the results we could note that, in the real Internet environments, the proposed DSGF approach outperforms RS-MDC and Evenly FEC, which is similar to the results obtained in the i.i.d. packet loss environment. In Fig.4.12, we compare the proposed DSGF approach with the other two approaches in Gilbert burst loss environment. As indicated in [65], we set the average burst length as two, and since burst packet loss usually requires higher redundancy comparing to i.i.d. loss, 60%

parity packets are inserted for the 10% average packet loss rate for both the DSGF and Evenly FEC approaches. In burst loss environment, the DSGF approach outperforms the Evenly FEC significantly, with an average gain of more than 2 dB, being higher than that in i.i.d. case (Fig.4.7.a). This is because, in burst loss environment, consecutive packets tend to be lost together, in this case FEC coding at sub-GOP level can mitigate this kind of losses more efficiently than Evenly FEC. It is also noted that, in burst loss environment, the gain over the RS-MDC is lower than that in i.i.d. case (Fig.4.7.a), this is due to the packet arrangement in RS-MDC. In fact, in RS-MDC, for each frame, redundant packets are grouped together and sent sequentially before the primary packets, in this case, the probability of losing both primary and redundant packets for the same video content becomes quite low. Nevertheless, in all the bitrate higher than 500 kbps, the DSGF approach provides much higher PSNR than RS-MDC, and the gap increases dramatically with bitrate.

4.4 Conclusions

In this section, a real-time FEC video transmission approach has been proposed. We firstly presented the general idea of this approach, then the theoretical model for creating sub-GOP and allocating FEC protection packets was given. With this model, the allocation problem becames a constrained optimization problem. To resolve it, a fast greedy algorithm was proposed. In order to validate the proposed approach, its performance was compared with other real-time error resilient approaches, such as RS-MDC and Evenly FEC. Experimental results demonstrated that the proposed approach had considerable practical value for real-time applications.

It is worth reporting that the work reported in this section has led to the following publications and patent:

1. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Real-Time Forward Error Correction for Video Transmission, IEEE Visual Communication and Image Processing, VCIP 2011

2. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Dynamic Sub-GOP Forward Error Correction Code for Real-time Video Applications, IEEE Transactions on Multimedia, Vol.14, No.4, 2012. doi:10.1109/TMM.2012.2194274

3. Jimin Xiao, Tammam Tillo, Dynamic Sub-GOP Forward Error Correction Code

for Real-time Video Streaming, application number for China patent: 201110170067.6 (Approved, in Chinese)

(a) $S = 5$

(b) $S = 10$

Figure 4.3: RS allocation example with the greedy algorithm; packet loss rate $p = 5\%$; one GOP has 30 P-frames; RS parity packet rate $\mu = 20\%$; each frame includes S slices; (a) $S = 5$, (b) $S = 10$.
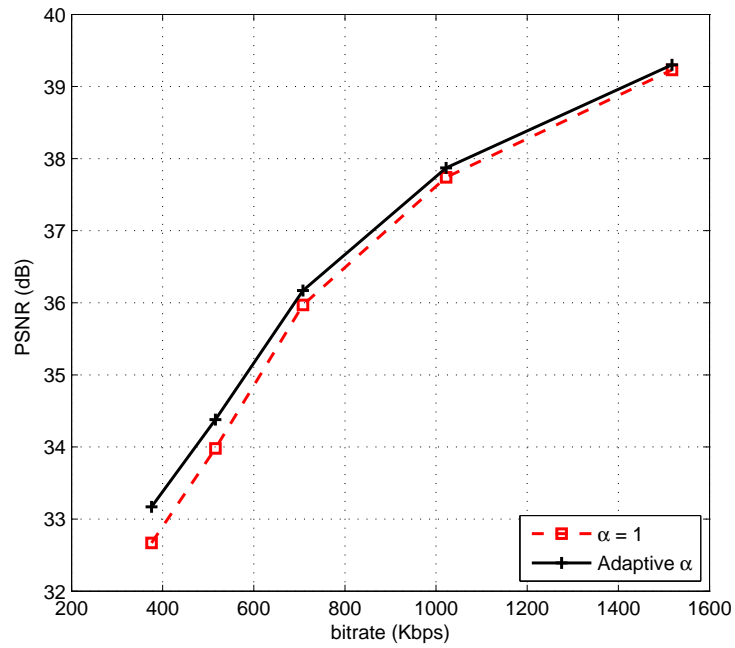
Figure 4.4: Average PSNR versus bitrate for fixed α value ($\alpha = 1$) and adaptive α for the DSGF approach; CIF Foreman sequence is used; GOP length is 30; packet loss rate is 5%, parity packet rate is 20%.
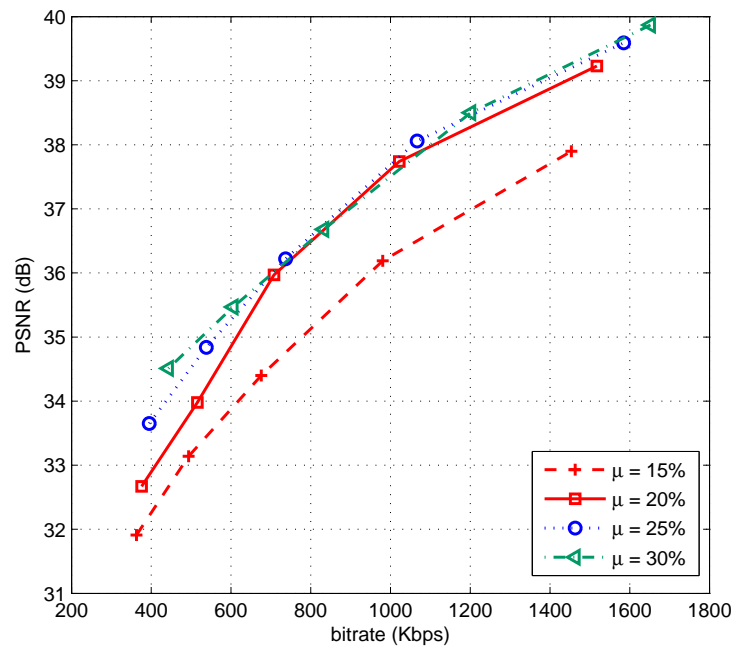
Figure 4.5: Average PSNR versus bitrate for various parity packet rate μ; CIF Foreman sequence is used; packet loss rate is 5%; RS parity packet rate μ includes 15%, 20%, 25% and 30%.
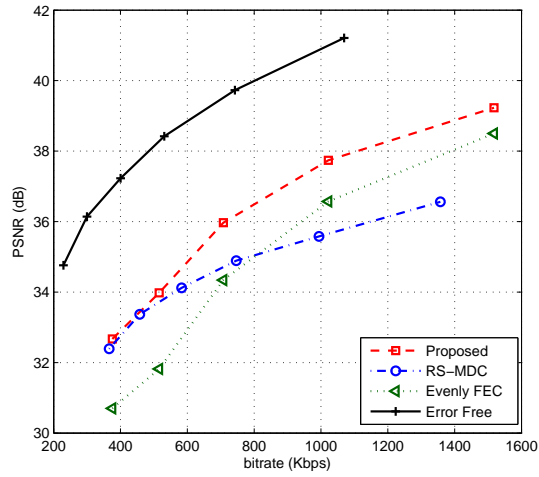
(a) Foreman
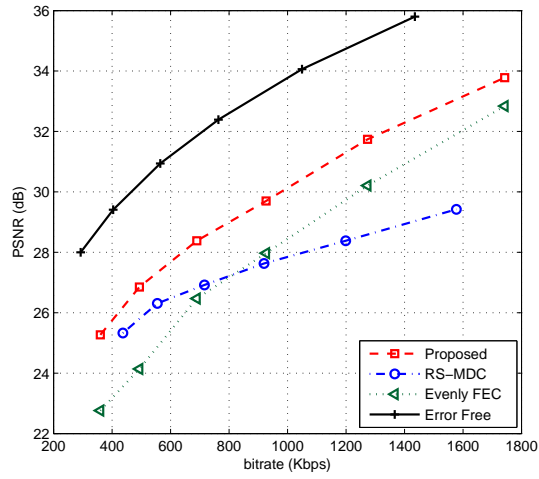
(a) Bus
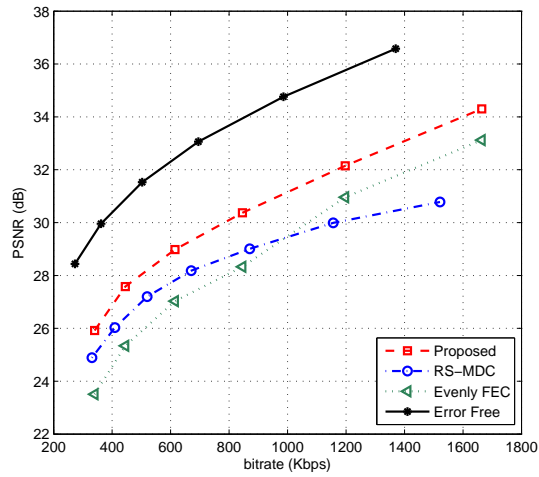
(c) Stefan

Figure 4.6: Average PSNR versus bitrate curves; the network packet loss rate is 5% and the parity packet rate μ is 20%.
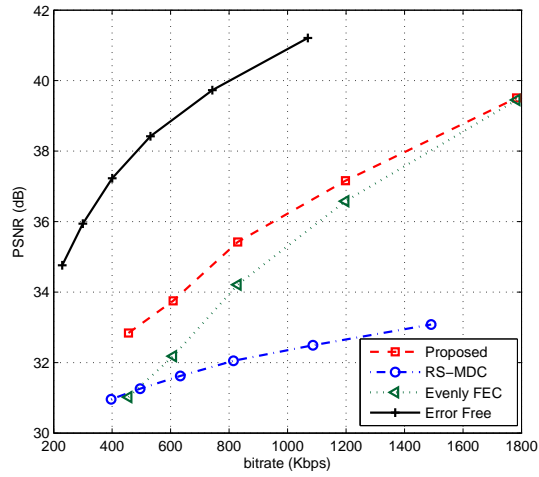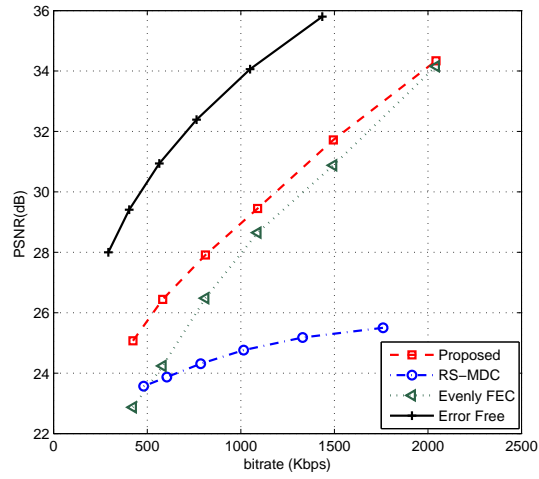
(a) Foreman

(a) Bus
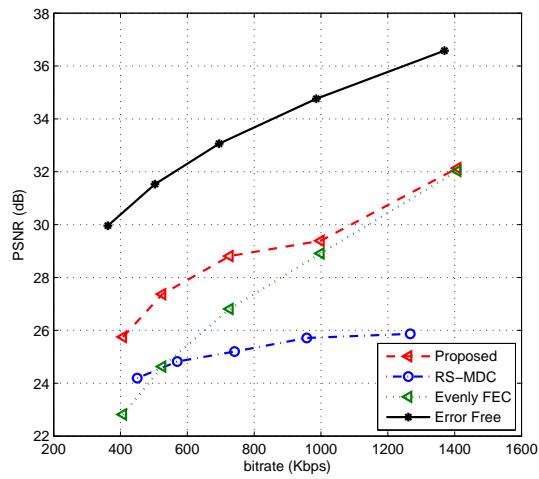
(c) Stefan

Figure 4.7: Average PSNR versus bitrate curves; the network packet loss rate is 10% and the parity packet rate μ is 40%.
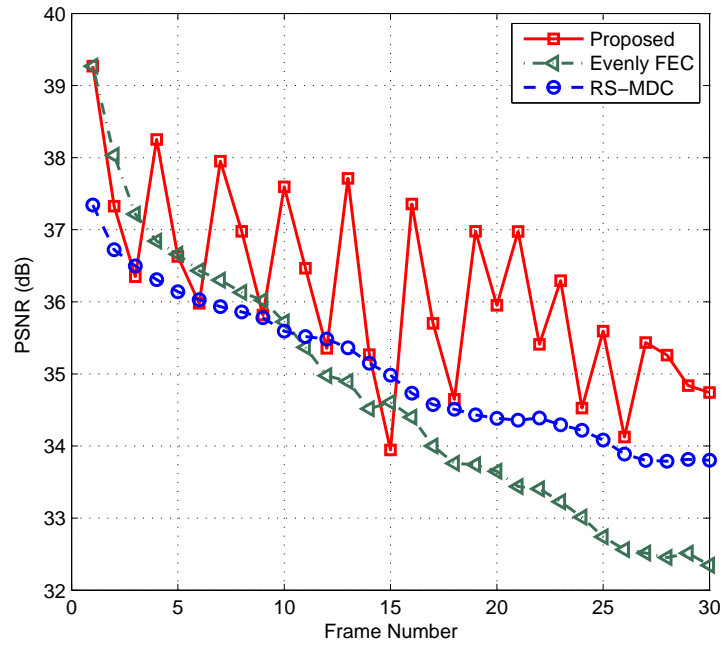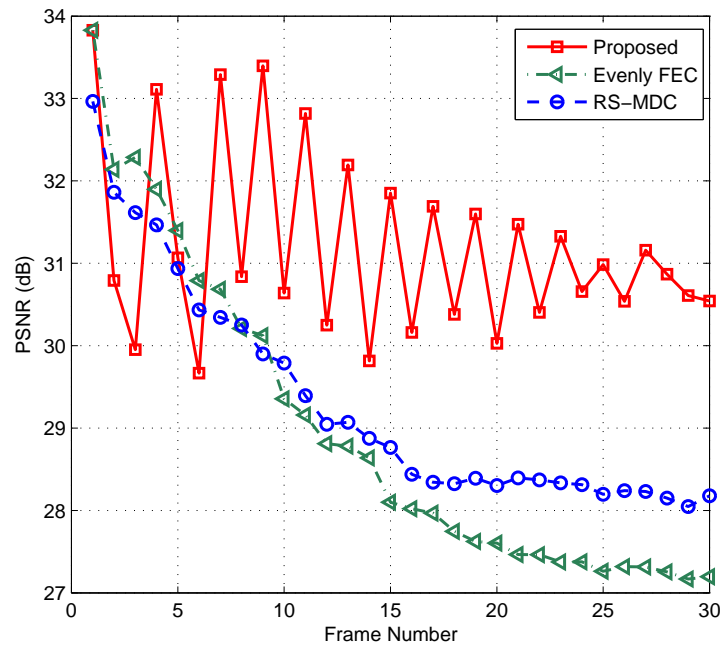
(a) Foreman

(b) Stefan

Figure 4.8: Video quality versus frame number in one GOP with length 30; Packet loss rate is 5%, parity packet rate μ is 20%; (a) CIF Foreman sequence; QP = 26; bitrate for the proposed approach and the Evenly FEC approach is 707.9 Kbps, for RS-MDC is 746.5 Kbps; (b) CIF Stefan sequence; QP = 32; bitrate for the proposed approach and the Evenly FEC approach is 845.4 Kbps; for RS-MDC is 870.5 Kbps.

(a) the 10*th* video frame with PSNR 38.27dB

(b) the 11*th* video frame with PSNR 35.20dB
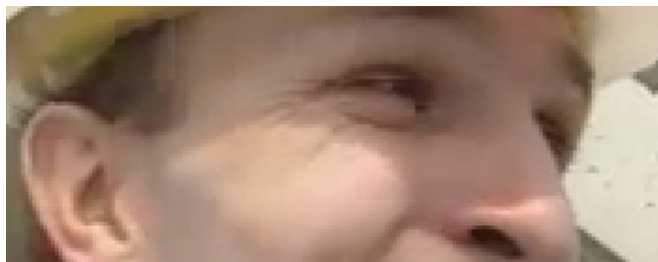
(c) zoom in the most damaged area in Frame 11*th*.

Figure 4.9: Two consecutive frame in one decoded sequence after random packet loss (p=5%), (a) 10*th* frame,with PSNR 38.27 dB; (b) 11*th* frame,with PSNR 35.20 dB; (c) zoom in the most damaged area in Frame 11.
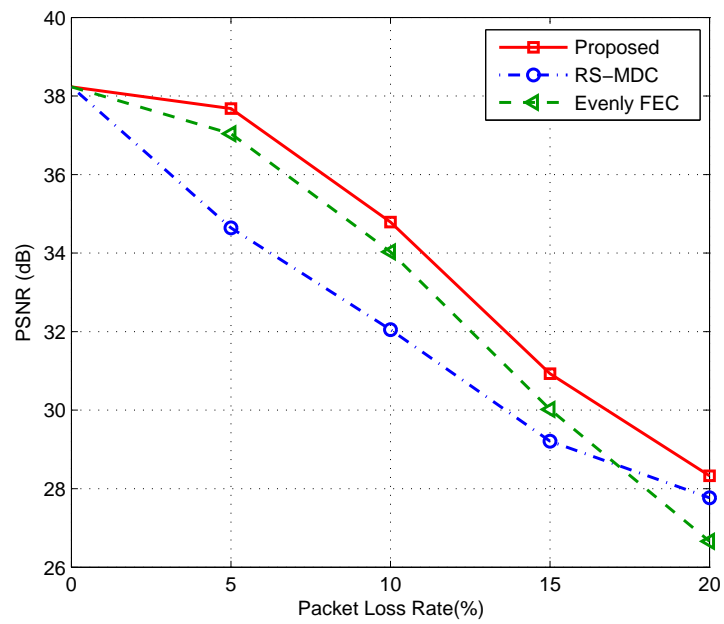
Figure 4.10: Video quality versus different packet loss rate; CIF Foreman sequence is used; QP is 26; GOP length 30; the redundancy of the three approaches is $\mu = 36.5\%$.
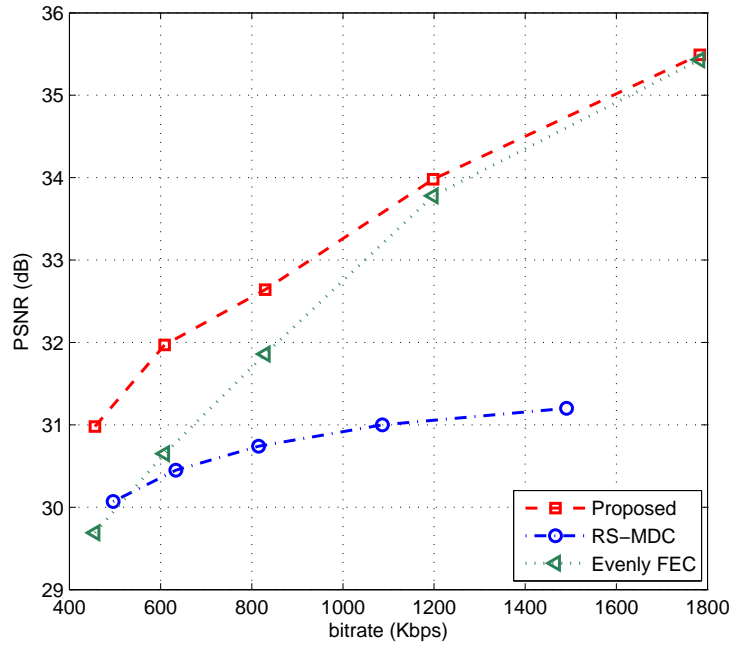
(a) Foreman
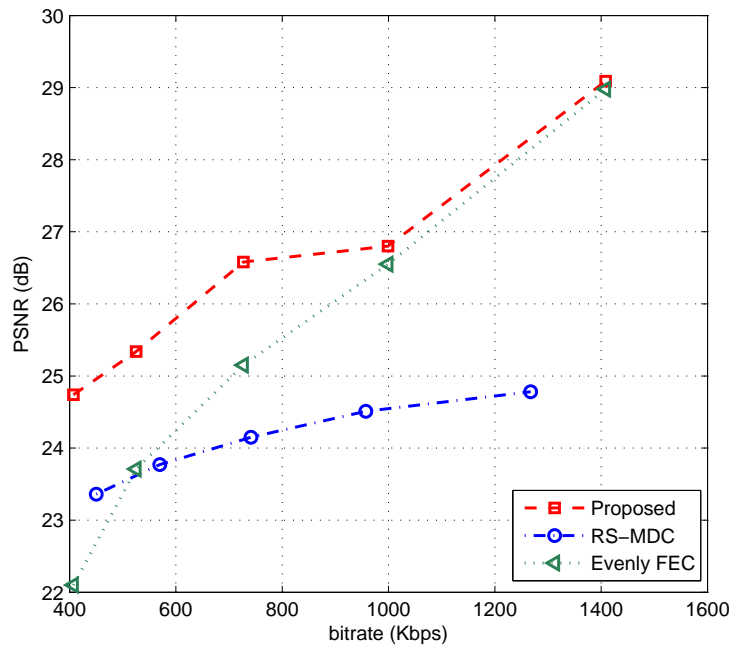
(b) Stefan

Figure 4.11: Average PSNR versus bitrate curves; the network packet loss follows the pattern in file 10 of Q15-I-16r; μ is 40%; (a) Foreman sequence,(b) Stefan sequence.
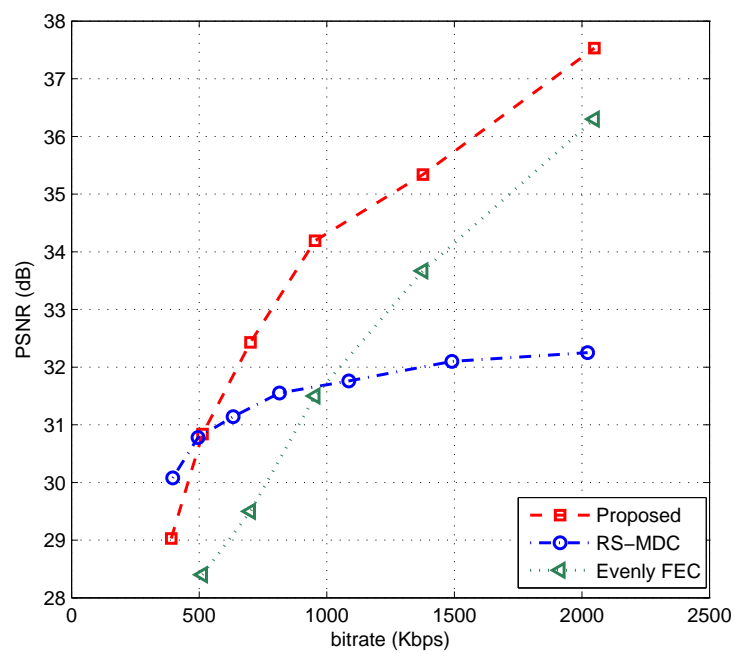
Figure 4.12: Average PSNR versus bitrate curves for the Foreman sequence; the packet loss rate is 10%; average burst length is two; μ is 60%.

Chapter 5

A Real-time Error Resilient Video Streaming Scheme Exploiting the Late- and Early-arrival Packets

When the video packets are transmitted over lossy networks, some packets are dropped by the underlying network facilities, and they will never reach the destinations, thus we call them *physical lost* packets. Moreover, some packets may arrive at the destination after long delay, which is larger than the maximum allowed end-to-end delay for the applications. In general, these packets are also regarded as lost packets by the video applications. To distinguish these packets from the *physical lost* packets, in the following, we will refer to them as *late-arrival* packets. Meanwhile, for some video packets, the transmission delay is short, and they may arrive at the destination before the display deadline of their temporal-previous frames, in this article those are called *early-arrival* packets. It is worth recalling that the end-to-end delay constraint is an application-dependent parameter, so for example, for real-time video conferencing/telephony applications, the acceptable end-to-end delay is between 150 ms and 400 ms according to ITU-T G.114 [77].

In this section, a Real-time Video Streaming scheme exploiting the Late- and Early-arrival packets (RVS-LE) in an optimal fashion is proposed. In the proposed approach, we are targeting real-time applications with stringent end-to-end delay constraint, e.g., delay less than one Round-Trip Time (RTT), thus retransmission is not considered. The proposed RVS-LE approach includes two mechanisms. One is to use these packets to update the reference frames to make them more consistent with the encoder side,

and this will eventually reduce the error propagations. The other mechanism is to use these packets to increase the chance of successfully decoding the RS code. It is worth mentioning that exploiting the *late-arrival* packets to update the reference and stop error propagations was also used in [38, 39]. In this section, the *late-arrival* packets will be jointly optimized with the FEC code in an optimal way. To further exploit the *late-* and *early-arrival* packets, sub-GOP based systematic Reed-Solomon (RS) code, which is proposed in Chapter 4 to improve the error correction performance of the RS code, is also used to increase the probability of successfully decoding the current frame. Finally, in the proposed RVS-LE approach, the size and the parity packet number for each sub-GOP are optimally tuned, taking into consideration the maximum end-to-end delay, the network conditions, which accounts for the *physical lost* packets, *late-* and *early-arrival* packets to minimize the total distortion, under the constraints of the inserted redundancy and the maximum end-to-end delay. It is worth noticing that in the previous section, the network transmission delay, the *late-arrival* packets were not taken into consideration during the sub-GOP and parity allocation process, which makes them less optimal for practical video streaming applications, since in practical applications, the video packets take half RTT delay to reach the destinations.

The rest of the section is organized as follows. In Section 5.1, the proposed RVS-LE approach is presented. Later, the frame level Evenly FEC approach is introduced; this approach is used as a benchmark for the proposed RVS-LE coding. In Section 5.2 some simulation results validating the proposed approach are given. Finally, some conclusions are drawn in Section 5.3.

5.1 Proposed Video Streaming Approach

The same as the DSGF approach introduced in the previous section, the B-frames will not be used in order to minimize the delay in the video encoding process, so the IPPP GOP structure will be used. Similarly, to make the RS code efficient, fixed length slice scheme in term of byte, will be used to create slices, the slice length is decided by the Maximum Transmission Unit (MTU) of the underlying networks. It is important to note that, in the proposed scheme, the length of the packets used to encapsulate the RS parity symbols is similar to the length of the packets used to encapsulate video slice data. For detailed information of creating slice and RS packets, please refer to Fig.4.1.

5.1.1 RVS-LE Approach: A Case Study

In the proposed approach, all the video packets arriving at the destination with different delay are exploited, including *late-* and *early-arrival* packets. To do this, we propose to incorporate packets from a Sub-GOP, which usually contains more than one video frame, to one RS coding block, and add parity packets at the end of the sub-GOP. In this case, both *late-* and *early-arrival* packets could be used by the RS code to recover the still unavailable packets.

Fig.5.1.a shows one example of how to exploit the *late-* and *early-arrival* packets, where slices of one sub-GOP (3 frames) are used as one RS coding block, and the RS parity packets are allocated and appended at the end of each sub-GOP, thus RS code $(15, 12)$ is used for this sub-GOP. Here packets 3 and 7 are *late-arrival* packets, because they arrive at the destination later than the display deadlines of the frames they belong to; packets 6, 10, 11, 12, 13 and 14 are *early-arrival* packets, as they arrive at the destination before the display deadlines of their temporal-previous frames. The decoder will decode the first frame with slice 3 being concealed. However, this packet arrives at the destination by the display deadline of the second frame, so in order to stop the error propagations caused by the injected errors in the prediction loop by the concealment stage, the first frame will be re-decoded with the newly received slice. It is worth indicating that the updated version of the first frame will not be displayed, nevertheless, it will be used to update the reference buffer, so as to stop the propagated distortion. By the display deadline of frame 2, slice 7, which belongs to frame 2, does not arrive at the destination. However, with four early-arrival packets, i.e., 10, 12, 13, 14, in addition to the other arrived packets belonging to the current sub-GOP, which totally amounts to twelve packets, the RS code will be able to recover packet 7. This allows to correctly decode frame 2 and recover packet 9, i.e., the lost packet in frame 3. In this case, for frame 2 and 3, there is no concealment distortion propagated from the first frame, and no distortion will propagate to the following frames.

Fig.5.1.b is another example that serves to demonstrate how the *late-* and *early-arrival* packets are exploited between different sub-GOPs. In this example, by the display deadline of frame 3, the amount of received packets for the first sub-GOP is not enough to recover all its source packets. Thus, there will be concealment distortion for the lost packets 3, 9. For the *later-arrival* packet 7, it will arrive at the destination
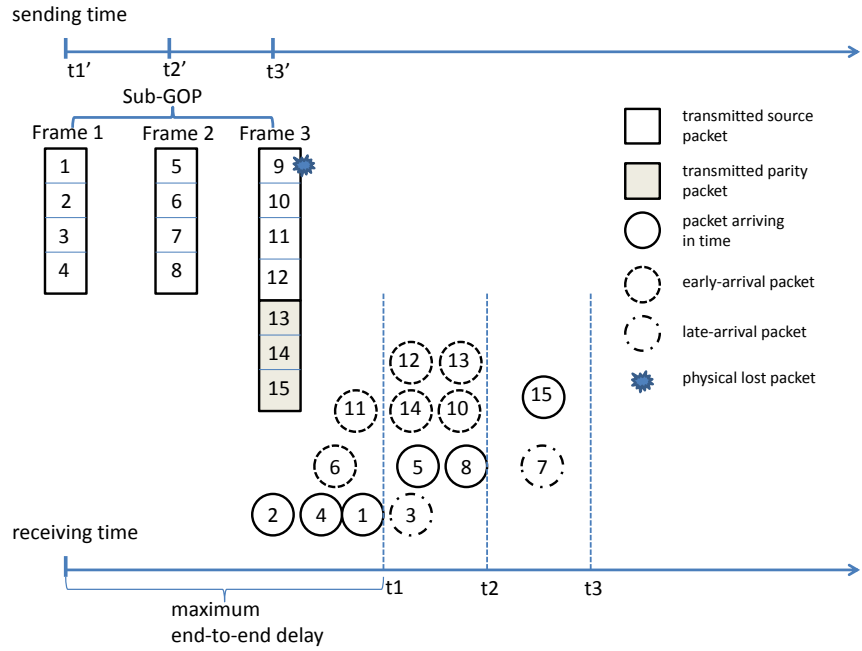
a. example with one sub-GOP
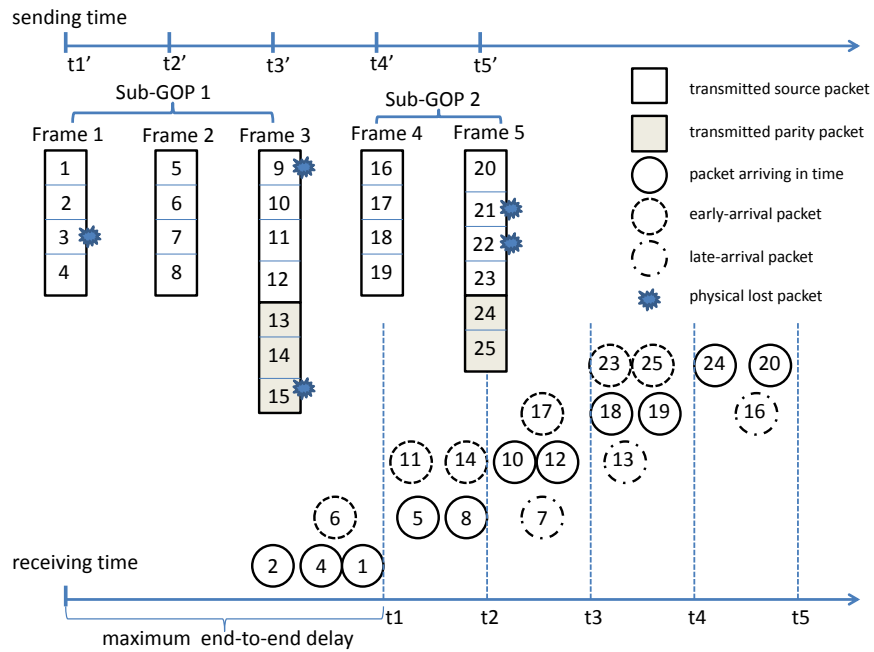
b. example with two sub-GOPs

Figure 5.1: Examples of sub-GOP and RS parity packets allocation and packet transmission delay for the proposed approach; the I-frame (with index 0) is not shown in the figure; $t1$, $t2$, $t3$, $t4$, $t5$ are the display deadline for frame 1, frame 2, frame 3, frame 4 and frame 5, respectively; $t1'$, $t2'$, $t3'$, $t4'$, $t5'$ are the sending time for frame 1, frame 2, frame 3, frame 4 and frame 5, respectively.

before the display deadline of frame 3. Thus before decoding frame 3, this slice will be re-decoded and updated. It is worth noticing that the *early-arrival* packet 17 belongs to the second sub-GOP, and it cannot be used by the RS code of the first sub-GOP. Later, packet 13 arrives before the deadline of the fourth frame ($t4$), and with this newly received packet, the total received packets for the first sub-GOP becomes twelve, which allows the RS-decoder to recover all the lost packets of the first sub-GOP, and consequently, the video decoder will re-decode all the video frames of the first sub-GOP using the previously arrived and RS-recovered packets, and following that the reference buffer will be updated. Thus, the concealment distortion for losing slice 3 and 9 will stop propagating to the following frames, which means that the mismatch caused by the concealed slice 3 may only affect frames 2 and 3, whereas the concealed slice 9 only affects frame 3.

As demonstrated in the above examples, the general principal of the proposed RVS-LE approach could be described as follows. On the receiver side, all the received packets of the current frame-to-be-displayed will be decoded and displayed at the display deadline. If some packets are not available then the RS-decoder will try to recover these missing packets using the already received packets. Note that the already received packets may include the *late-arrival* packets of the temporal-previous frames, the *early-arrival* packets of the following frames and the received parity packets of the current sub-GOP. Finally, for all the cases where the RS-decoder fails to recover the missed packets of the current frame, the error concealment will be invoked. It is also important to note that in some cases, at the end of the current sub-GOP the total number of received packets may not be enough to make the RS decoder recover all the missed packets, this sub-GOP will be described in the following as *pending* sub-GOP, in other words, the decoding process of these sub-GOPs will be regarded as uncompleted. Moreover, the *pending* sub-GOP category will also include those sub-GOPs that are completely decoded (all their packets arrived or recovered); however, their previous sub-GOPs are *pending*, because in this case the mismatch error may propagate from the previous sub-GOPs to the following ones. In order to mitigate the mismatch error, each newly received packet will be checked to see whether it belongs to any previously *pending* sub-GOPs. If one or more newly arriving packets belong to a *pending* sub-GOP, then they will be firstly used together with the previously received packets that belong to this *pending* sub-GOP, by the RS-decoder, to try to recover its missing

packets. Secondly, with the recovered missing packets and the newly received source packets, the *pending* sub-GOP will be re-decoded by the video-decoder. If the previously described stage leads to an update of the content of the previous frames then the reference buffer frame will be updated and also all the following sub-GOPs will be re-decoded and updated.

5.1.2 Optimal sub-GOP and RS Parity Packet Allocation

This section will address the problem of how to group frames into sub-GOPs, and how to allocate the RS parity packets among all the sub-GOPs in order to maximize the exploitation of the *late-* and *early-arrival* packets. It is worth noticing that, in the proposed approach, I-frames are not included in any sub-GOP, and are protected at frame level. This is because, in general, I-frames generate much more bits than P-frames, and therefore more source packets are produced for I-frames, and consequently, protecting them directly is efficient enough. While for the allocation problem of the P-frames, we have to note that, on one hand, if the sub-GOP includes too few P-frames, the *late-* and *early-arrival* packets cannot be exploited properly, and the value of K for the RS code will not be large enough to make the RS code efficient. On the other hand, if the sub-GOP includes too many P-frames, the time interval for the whole sub-GOP will be very long. Having a long time interval means that by the display deadline of many frames of the current sub-GOP; the amount of received packets of this sub-GOP will be, with high probability, too few to allow the RS-decoder to recover any unavailable packets. To optimally generate the sub-GOPs and properly allocate them the RS parity packets, we need to know some detailed information of this GOP, including the number of P-frames in one GOP, the slice number in each frame, the concealment distortion caused by losing each slice, and how the distortion propagates. However, the actual values of some of these parameters are not available for real-time on-the-fly transmission system. To overcome this limitation, we use the expected values of these parameters instead of their actual values, which will obviously make the obtained solution sub-optimal. Nevertheless, the obtained results demonstrate the effectiveness of the proposed procedure. These parameters are listed in the following.

- Instead of the actual number of slices in each P-frame, for each GOP, we will use the average number of slices per P-frame, $\overline{S}$. Therefore, to predict this value we will assume that it does not change from GOP to GOP, and consequently, for a

practical approach we will assume that the current $\overline{S}$ is similar to the actual value of the average number of slices per frame of the previous GOP, which could be obtained at the end of the encoding process of the previous GOP. This approach becomes more accurate for low motion video with little change in the content.

- The actual concealment distortion caused by losing a slice will be replaced by the expected concealment distortion of losing one slice, $\bar{d}$, we assume $\bar{d}$ to be the same for all slices. To estimate the amount of propagated distortion to the following frames, an attenuation function $f(n)$ will be used. This means that if the concealment distortion of one slice is $\bar{d}$, it will propagate to the following frames and become $f(n)\bar{d}$ after n frames. For the sake of simplicity the function $f(n) = \alpha^{n-1}$ ($0 < \alpha \leq 1$) is employed, this expression approximates, at low levels of attenuation, the function $f(n) = (1 + \lambda_1 n)^{-1}$ and $f(n) = (1 + \lambda_2 n)^{-1/2}$ reported in [74] and [75], respectively, with α, λ_1 and λ_2 being parameters that depends on the sequence itself. Moreover, in order to model multiple slice losses, let us assume that the distortion caused by losing multiple slices are uncorrelated, in this case, the total expected distortion for the whole GOP will be the sum of the expected distortions caused by losing the individual slices. The assumption that slice concealment distortion is uncorrelated is reasonable. In fact, concealment distortions, can be considered as uncorrelated with the pixel values, then concealment distortions caused by losing different slices can also be considered as uncorrelated. The additive distortion model has been verified experimentally in [6].

One more parameter that we need to take into account, while solving the allocation problem, is the maximum allowed end-to-end delay (T_{max}). In fact, this parameter will determine the probability of receiving packets at the display deadline of each frame. Let us use $p_{k,i}$ to denote the probability of receiving packets belonging to frame i at the display deadline of frame k, this probability could be evaluated as:

$$p_{k,i} = cdf\left(T_{max} + (k - i)\ T_0\right) \tag{5.1}$$

with T_0 being the time interval between two adjacent frames, e.g., for 30 fps application T_0 would be 33.3 ms; and $cdf(t)$ is the Cumulative Distribution Function (CDF) of the packet delay, which means the ratio of packets with delay less than t is $cdf(t)$,

$$P(v_{k,r_{m+1}}) = \binom{R(r_{m+1})}{v_{k,r_{m+1}}} (p_{k,r_{m+1}})^{R(r_{m+1})-v_{k,r_{m+1}}} (1 - p_{k,r_{m+1}})^{v_{k,r_{m+1}}} \tag{5.9}$$

$$\mathcal{I}_{k,m+1}(z) = \{u_{k,j}, v_{k,r_{m+1}} \mid (\sum_{j=r_m+1}^{r_{m+1}} u_{k,j}) + v_{k,r_{m+1}} = z, \forall u_{k,j} \in \mathbb{Z}^*, \forall v_{k,r_{m+1}} \in \mathbb{Z}^*\}$$

$$\tag{5.10}$$

this function is dictated by the network conditions. In (5.1) the frame index i could be either larger or smaller than k, with the condition that $T_{max} + (k - i) T_0 > 0$, so for example by the display deadline of frame k the probability of receiving packets belonging to the previous frame is $cdf(T_{max} + T_0)$, this is because the total waiting time for the packets belonging to frame $k - 1$ is $T_{max} + T_0$. Moreover, to solve the allocation problem, the maximum number of parity packets for all the P-frame in one GOP, R, need to be specified, and consequently the number of RS parity packets allocated for the ith P-frame, $R(i)$, need to satisfy the following constraint:

$$\sum_{i=1}^{L} R(i) \le R \tag{5.2}$$

with L being the number of P-frames in one GOP; it is worth noting that if for example $R(1) = R(2) = 0$ and $R(3) = 3$ then that means that the first, second, and third frames form one sub-GOP protected by three RS parity packets, this case describes the first sub-GOP in Fig.5.1.b and the sub-GOP in Fig.5.1.a.

At this stage, with all the parameters and the constraints listed above, we could tune the sub-GOP size and allocate the parity packets. To solve this problem, we assume that these packets will be inserted in totally t positions, these positions are identified by the frame indexes r_1, r_2,..., r_t, whereas for all the other P-frames no RS parity packets will be inserted. Moreover, let us assume that the number of the inserted RS parity packets is $R(r_1)$,$R(r_2)$,..., $R(r_t)$. According to this notation, the RS parity packets allocated for frame r_{m+1} are used to protect frames $[r_m + 1, r_{m+1}]$, and this

$$\bar{d}_{m+1}(k) = \sum_{z=N-K+1}^{N} \sum_{\mathcal{I}_{k,m+1}(z)} \left(\left(P(v_{k,r_{m+1}}) \prod_{j=r_m+1}^{r_{m+1}} P(u_{k,j}) \right) \left(\sum_{j=r_m+1}^{min(k,r_{m+1})} u_{k,j} f(k-j)\bar{d} \right) \right) \tag{5.11}$$

86

$$\bar{d}'_{m+1}(k) = \begin{cases} \bar{d}_{m+1}(k) & \text{if } k \le r_{m+1} \\ \sum_{z=N-K+1}^{N} \sum_{\mathcal{I}_{r_{m+1},m+1}(z)} \left(\left(P(v_{r_{m+1},r_{m+1}}) \prod_{j=r_m+1}^{r_{m+1}} P(u_{r_{m+1},j}) \right) \left(\sum_{j=r_m+1}^{r_{m+1}} u_{r_{m+1},j} f(k-j)\bar{d} \right) \right) & \text{if } k > r_{m+1} \end{cases}$$

$$(5.16)$$

sub-GOP will be indexed as the $(m+1)$ th sub-GOP [1]. Therefore, the RS code used for this sub-GOP would be $(N, K) = \left((r_{m+1} - r_m)\,\overline{S} + R(r_{m+1}), (r_{m+1} - r_m)\,\overline{S} \right)$. For the example reported in Fig.5.1.b, the allocated RS parity packets are in the following two positions $r_1 = 3$, $r_2 = 5$, i.e., $t = 2$, and $R(r_1) = 3$, $R(r_2) = 2$.

Now let us evaluate the expected distortion in frame k, with $k \ge r_m + 1$, caused by slice losses within sub-GOP $m+1$, i.e., frames $[r_m + 1, \; r_{m+1}]$. To evaluate this expected distortion, let us denote the number of source packets sent at time i and have not arrived at the receiver side at the display deadline of frame k by $u_{k,i}$, and the probability of this event by $P(u_{k,i})$. Similarly, $v_{k,r_{m+1}}$ will be used to denote the number of RS parity packets sent at time r_{m+1} and have not arrived by the display deadline of frame k, and the probability of this event is $P(v_{k,r_{m+1}})$. As previously described, the probability of receiving a packet belonging to the i-th frame, by the display deadline of the k-frame, is $p_{k,i}$; therefore, the probability $P(u_{k,i})$ becomes

$$P(u_{k,i}) = \binom{\overline{S}}{u_{k,i}} (p_{k,i})^{\overline{S}-u_{k,i}} (1-p_{k,i})^{u_{k,i}} \qquad (5.8)$$

Similarly, the probability of the event $v_{k,r_{m+1}}$ is shown in (5.9). According to the property of systematic RS code, the distortion in frame k caused by the unavailable slices within sub-GOP $m+1$, is because the total number of received packet of this sub-GOP is less than K by the display deadline of frame k. In other words, the RS decoder cannot recover the unavailable source packets within this sub-GOP. Thus, in order to evaluate the expected distortion in frame k, let us define the set $\mathcal{I}_{k,m+1}(z)$ as in (5.10). This set stands for all the possible patterns of packet reception, that have totally z unavailable packets in the $(m+1)th$ sub-GOP by the display deadline of frame k. Therefore, the expected distortion in frame k ($k \ge r_m + 1$) caused by the $(m+1)th$ sub-GOP could be formulated as (5.11). As previously defined, $f(n)$ is the distortion attenuation function, $\overline{d}$ is the expected distortion caused by each slice. It is important to note that, the term $\bar{d}_{m+1}(k)$ contains all the distortion in frame k that is caused by the $(m+1)th$ sub-GOP, including the propagated distortion from

---

[1]Note that $r_0 = 0$.

the previous frames of this sub-GOP. So (5.11) is obtained by adding up the weighted concealment and propagated distortion of all the packet loss patterns that make the RS code fail to recover the unavailable packets, where the weight factor is the probability of all the specific patterns. Consequently, the total expected distortion caused the by $(m+1)$ th sub-GOP could be obtained by summing up the caused distortion within frames $[r_m + 1, L]$ as

$$\bar{D}_{m+1} = \sum_{k=r_m+1}^{L} \bar{d}_{m+1}(k) \tag{5.12}$$

with L being the number of P-frames in one GOP. Finally, by adding up the expected distortion of all the sub-GOPs within the GOP, the total expected distortion of the whole GOP is

$$\bar{D}_{total} = \sum_{k=1}^{r_t} \bar{D}_k \tag{5.13}$$

At this stage, the optimization problem can be formulated as the following constrained minimization:

$$\begin{cases} \min & \bar{D}_{total} \\ \text{subject to} & \sum_{i=1}^{L} R(i) \leq R \end{cases} \tag{5.14}$$

5.1.3 Implementation of sub-GOP and Parity Packet Allocation

Given the variables in (5.14) are integers and the mathematical complexity of the equations, we believe that it is impossible to get a closed-form solution to the optimization problem given by (5.14). Moreover, it is computational prohibitive to numerically get the global optimal solution by trying to evaluate $\bar{D}_{total}$ for all the possible allocation patterns, and selecting the one that leads to the minimal $\bar{D}_{total}$. In fact, when one GOP includes L P-frames, and the number of RS parity packets is R, there would be $\binom{L+R-1}{R}$ possible allocation patterns to be investigated. For example, if $L = 30$ and the number of RS parity packets is 40, there would be $\binom{69}{40} = 2.39 \times 10^{19}$ possible allocation patterns. Obviously, calculating the value of D_{total} using (5.13) for all the 2.39×10^{19} allocation patterns would be unfeasible.

Since it is computational prohibitive to determine the global optimal solution, we propose to simplify this process by firstly partitioning frames into sub-GOPs, and secondly allocate the redundancy among those sub-GOPs.

- The first stage will be executed before the video encoding process, where the frames partition problem will be solved using a greedy algorithm that finds the

local optimal size for each sub-GOP. In other words, the local optimal size for the first sub-GOP will be determined and then the second sub-GOP and so on. The sub-GOP size that minimizes the expected distortion *per frame* will be selected. To do this, the algorithm will try all the possible sizes starting from 1, and for each candidate the total expected distortion caused by this sub-GOP will be evaluated using (5.12), and then it will be divided by the size of the sub-GOP, so as to obtain the expected distortion *per frame*. Finally, the sub-GOP size that leads to the smallest expected distortion *per frame* will be chosen. After that, the algorithm will determine the local optimum sizes of the following sub-GOPs by repeating the previous process.

- The second stage, that aims at allocating RS parity packets for each sub-GOP, is executed in parallel with the video encoding process. For this reason the exact number of slices per frame, $S(i)$, will be known exactly at the end of the video encoding process of each frame i. Thus, based on this knowledge, and the known positions where parity packets will be inserted (from the first stage), the actual number of parity packets will be decided on the fly, using the following equations:

$$R(r_k) = \begin{cases} \lceil \mu \sum_{i=1}^{r_1} S(i) \rceil & k = 1 \\ \lceil \mu \sum_{i=1}^{r_k} S(i) \rceil - \sum_{i=1}^{k-1} R(r_i) & k > 1 \end{cases} \qquad (5.15)$$

with $R(r_k)$ being the amount of parity packets to be inserted at frame r_k. It is worth reminding that this position has been already determined in the first stage. The operation $\lceil X \rceil$ is used to get the minimum integer number greater than or equal to X.

5.1.4 Implementation of Reference Buffer Updating

At the video decoder side, the reference buffer updating technique can efficiently stop error propagations at the expense of high computational complexity. One solution to reduce the complexity is using the slice level reference buffer updating instead of frame level updating. This approach works by keeping track of all the slices that use a previously unavailable slice as reference, those slices will be called *dependant* slices in the following. This prediction information could be obtained by exploiting the motion vectors at the decoder side. In such a way to build a tree structure that describes the prediction dependency between slices. So once a slice is re-decoded and updated, all the slices whose tree root is this node will be also re-decoded and updated. Using

this slice level updating technique can reduce the computational complexity without sacrificing the error resilient performance.

In order to further lower the computational complexity of the reference buffer updating in the RVS-LE approach, another solution is that the *late-arrival* packets can only be exploited within the current sub-GOP, whereas the *late-arrival* packets of the previous sub-GOP are simply discarded. For this low complexity solution, the sub-GOP size and the parity packet number for each sub-GOP should be allocated in a different way because the total expected distortion (5.13) needs to be calculated differently. In this case, the term $\bar{d}_{m+1}(k)$, all the distortion in frame k that is caused by the $(m+1)th$ sub-GOP, should be evaluated as (5.16) instead of (5.11). The main difference between (5.16) and (5.11) is that for any packets belonging to sub-GOP $m+1$, if they arrive later than the frame r_{m+1}, they will not be exploited for updating. The remaining process of determining the sub-GOP size and parity packet number is the same, so we name this scheme as Simplified RVS-LE.

5.1.5 A Benchmark: Frame-Level Evenly FEC

In order to evaluate the performance of the proposed RVS-LE approach, frame level FEC coding is provided as a benchmark. For the delay constraint FEC video coding, one possible approach is to perform RS coding at frame level, which means that the RS coding block contains data packets from the same frame. Let us assume that, for the i-th frame in one GOP there are $S(i)$ source packets and $R(i)$ RS parity packets, and that we want to evenly allocate the parity packets over all the GOP frames, taking into account that $R(i)$ should be an integer and the value $S(i)$ varies from frame to frame, so $R(i)$ can be evaluated as $R(i) = \lceil \mu S(i) \rceil$, where μ is the intended parity packet rate of RS coding. For this approach, the RS parity packets are evenly allocated among all the frames, from now on we name this approach as Evenly FEC. It is important to note that, when $S(i)$ and μ are small, at least one RS parity packet will be allocated. Let us take one example, $S(i) = 1$ and $\mu = 0.2$, then $R(i)$ is 1, in this case, the Evenly FEC approach degrades to duplicating the source packets.

It is worth noting that, with the fast algorithm of the proposed RVS-LE approach, the number of parity packets for each sub-GOP is proportional to the number of source packets within it. Therefore, unequal error protection scheme based on the frame position of the GOP is not applied here. This is why the Evenly FEC scheme is chosen

90

Table 5.1: Average delay and the rate of unreceived packets on time using different maximum end-to-end delay, for packet loss and delay patterns file 3,10 and 20 in Q15-I-16r.

File	average delay	the rate of unreceived packets on time (%), with different maximum end-to-end delay				
		infinity	350 ms	300 ms	250 ms	200 ms
$f3$	125 ms	3.25	3.25	3.25	3.25	3.33
$f10$	160 ms	11.38	11.68	13.41	18.16	26.56
$f20$	160 ms	20.68	21.08	22.49	27.00	33.90

as our benchmark.

5.2   Experimental Results

Our simulation setting is built on the JM14.0 [64] H.264/AVC codec. CIF video sequences Foreman, Coastguard and Stefan are used for the simulations, with various levels of motion. The GOP structure is IPPP, the frame rate is 30 frames per second, and the GOP length is 30 frames. The reference frame number is 1, in other words, only the previous frame is used for prediction. As for packetization, one slice per packet is used, and taking the MTU of wireless network into account, we set the target slice length as 400 bytes. The average luminance PSNR is used to assess the objective video quality, which is averaged over 100 trials. Packet loss and delay patterns for the Internet experiments specified in Q15-I-16r [78] are used to emulate the real Internet environments. Table 5.1 lists the average delay and the rate of unreceived packets on time using different maximum end-to-end delay, for the file 3, 10 and 20 in Q15-I-16r. Those three files are used for simulation in this section, and we will refer to them as f3, f10 and f20. It is worth noticing that the received packet rate for the *infinity* end-to-end delay is equivalent to the *physical* packet loss rate. Moreover, we assume the channel conditions are available at the encoding side. For the unrecoverable lost packets, the *motion copy* error concealment algorithm implemented in JM14.0 [64] decoder is used. In the following simulations, we set the maximum delay to 300 ms, unless otherwise noted. This value is within the acceptable maximum delay for real-time video communications, e.g., video conferences, which is 150-400 ms [77].

Given that the two parameters $\overline{S}$ and α are used in the optimization process, and because they may not be accurately determined, in the first set of simulations we will

investigate the impact of these two parameters on the final results. In Table 5.2, we report the obtained average PSNR, along with the obtained pattern of sub-GOP size, for Foreman and Coastguard sequence with different values for $\overline{S}$, with the file f10, and $\mu = 40\%$. For the Foreman sequence we used QP= 26, and we tested $\overline{S} = \{3, 6, 10\}$ with 6 being the actual value of $\overline{S}$; for the Coastguard sequence we set QP to 32, and we tested $\overline{S} = \{3, 7, 12\}$ with 7 being the actual value of $\overline{S}$. From this table we could see the impact of using inaccurate $\overline{S}$ on the final results, and in particular we could see that when the used $\overline{S}$ is smaller than the actual value then the sub-GOP size will enlarge; when the used $\overline{S}$ is larger than the actual $\overline{S}$, the sub-GOP size will diminish. This happens, because small $\overline{S}$ requires the inclusion of more frames, in each sub-GOP, to improve the performance of the RS code; for large $\overline{S}$ small sub-GOP size becomes more suitable, and in this case, the beginning frames of the sub-GOP could be protected properly for small sub-GOP size. We can also notice that when the used $\overline{S}$ is the same as the actual $\overline{S}$, it leads to the best performance, and this demonstrates the effectiveness of the proposed sub-GOP and RS parity packet allocation algorithm, and the correctness of the optimization framework. One more conclusion that we can come to, from this results, is that despite the huge mismatch between the used $\overline{S}$ and its actual value, the average PSNR impairment is limited to 0.44, and 0.1dB for Foreman and Coastguard sequence, respectively. It is important to note that for practical application the expected mismatch between the estimated and actual value of $\overline{S}$ would be less than those reported in Table 5.2. Therefore, based on these results, we could conclude that using the predicted $\overline{S}$ instead of the actual $\overline{S}$ will lead to nearly optimal performance. Secondly, for the distortion attenuation function $f(n) = \alpha^{n-1}$, Fig.5.2 shows the effects of different values of α. As shown in this figure, α will not affect the performance hugely. Based on this observation, α will be set to one, this is equivalent to say that the propagated distortion does not attenuate.

In the second set of simulations, we study the effects of different rates of the inserted parity packets, μ, on the performance. The file f10 is used in this simulation. We try different values of μ, including 25%, 30%, 40%, 50% and 60%. Simulations with various quantization parameters (QP) are carried out to span a considerable bitrate range. Fig.6.4 shows the average PSNR versus bitrate with different μ, for Foreman sequence. In general, the PSNR curve for $\mu = 25\%$ and $\mu = 30\%$ is much lower than other cases, whereas, the PSNR curves for $\mu = 60\%$ is slightly lower than that of 40% and 50%, with

92
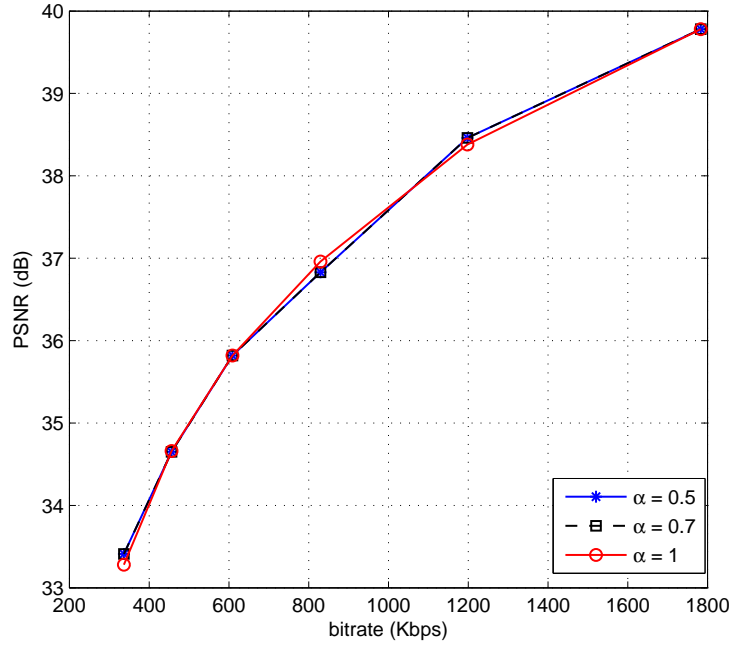
Figure 5.2: The effects the parameter α, Foreman CIF sequence, maximum delay is 300 ms.

Table 5.2: The effect of $\overline{S}$ on the optimization process

video sequence	$\overline{S}$	sub-GOP size	PSNR (dB)
Foreman	6 (actual $\overline{S}$)	5 4 4 4 4 4 3 1	37.14
(QP = 26)	3	5 5 5 5 5 4	36.70
	10	4 4 4 4 4 3 3 3	36.96
Coastguard	7 (actual $\overline{S}$)	4 4 4 4 4 4 4 1	31.44
(QP = 32)	3	5 5 5 5 5 4	31.34
	12	4 4 4 4 3 3 3 3 1	31.36
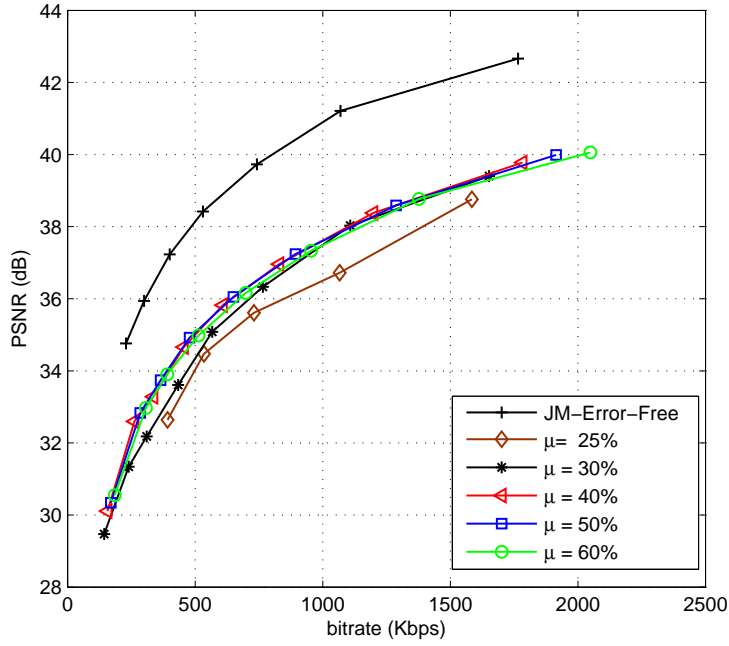
Figure 5.3: Average PSNR versus bitrate for various parity packet rates μ (redundancy), CIF Foreman sequence is used, packet loss and delay pattern file is f10.

the performance of 40% and 50% is almost equivalent. Consequently, in the following simulations, we use $\mu = 40\%$ for the file f10, because lowering the parity packet rate can improve the performance for the error free case. By doing similar simulations for different average packet loss rate, it is found that the RS parity packet rate, μ, should be proportional to the average packet loss rate, p. In this article, $\mu = \psi p + \phi$ is used, with $\psi = 2$ and $\phi = 0.2$. This kind of linear FEC redundancy allocation method was also applied in the implementation of Skype [79]. So in the following simulations, 26% and 60% RS parity packet rate are used for the file f3 and f20, respectively. It is important to note that, even with improper μ, the proposed method still outperforms the Evenly-FEC method. Fig.5.4 shows the average PSNR versus bitrate curves for f10 with improper parity packet rates $\mu = 25\%$ and $\mu = 50\%$. For the case with low redundancy $\mu = 25\%$, its gain over Evenly-FEC is larger than that with high redundancy $\mu = 50\%$. This phenomenon indicates that using the sub-GOP concept is more important when the inserted FEC redundancy is low.

In Fig.5.5, the effect of the proposed sub-GOP allocation is studied by comparing its performance with the empirical setting of the sub-GOP size as 1 and 2. In all the approaches, *late-arrival* packet update has been applied. It is important to note that,
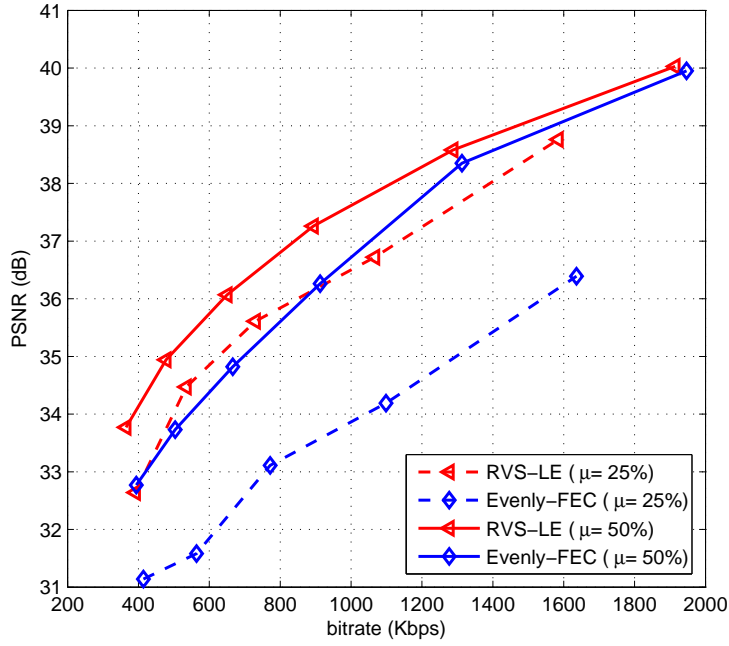
Figure 5.4: Average PSNR versus bitrate for improper parity packet rates $\mu = 25\%$ and $\mu = 50\%$, CIF Foreman sequence is used, packet loss and delay pattern file is f10.

when the sub-GOP size is 1, each frame has its own parity packets, and therefore in comparison with the sub-GOP approach the RS decoder does not have to wait for the parity packets allocated at several frames later. However, and in spite of that, its performance is not as good as that of the proposed RVS-LE. This is because when the sub-GOP size is small, the performance of the RS code is low. Also when the sub-GOP size is set to 2 the performance would be lower than RVS-LE; however, in this case the performance gap will be smaller, this is because the average sub-GOP length for the RVS-LE is 4.

In Fig.5.6, we compare the proposed approach with the Hybrid MDC with sophisticated error concealment approach proposed in [80]. In order to have fair comparison, in this simulation, Bernoulli channel model with 10% of packet loss rate, and without delay, has been used. Moreover, the RTP/UPD/IP header length 40 byte is accounted in the bitrate of the RVS-LE approach. From the reported results we can notice that the proposed approach outperforms the Hybrid MDC, by nearly 2 to 5dB, and the RS-MDC by 3 to 6 dB.

In Fig.5.7 and Fig.5.8, the average PSNR versus bitrate curves are plotted for packet loss and delay pattern in file f20 and f10, respectively. We compare the proposed RVS-

Figure 5.5: Effects of sub-GOP size, Foreman CIF sequence, QP = 28 (608.64 kbps), packet loss and delay pattern file is f10.

Figure 5.6: Average PSNR versus bitrate for RVS-LE, Hybrid-MDC and RS-MDC; Bernoulli 10% packet loss rate without delay; CIF Foreman sequence, GOP length 30.

LE with other approaches, including the simplified RVS-LE described in Section 5.1.4, Evenly-FEC, Evenly-FEC (No Update) and RS-MDC [6]. For the Evenly-FEC and Evenly-FEC (No Update), parity packets are allocated a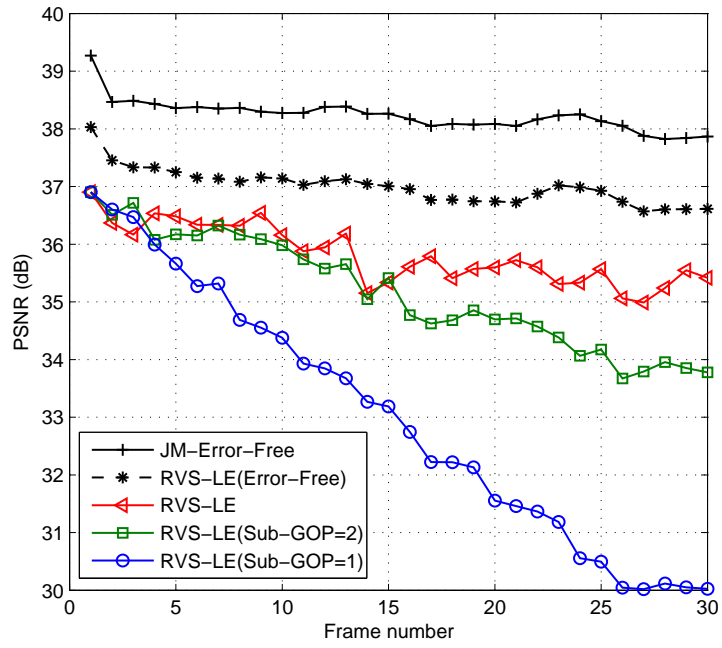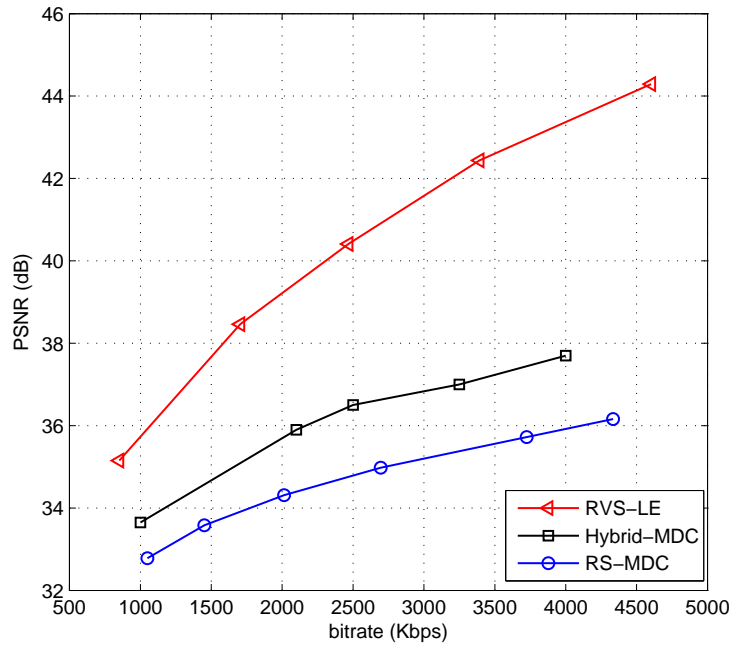s described in Section 5.1.5. However, the two approaches differ at the decoder side, with the former exploiting the *late-arrival* packets to update the reference buffer, and the latter simply discards them. As for the RS-MDC, it also discards the *late-arrival* packets. It is interesting to note that, in all cases the RVS-LE approach outperforms the RS-MDC approach significantly. Moreover, the comparison of both RVS-LE and Evenly-FEC, on one side, with the simplified RVS-LE and Evenly-FEC (No Update), on the other side, shows the importance of using all the *late-arrival* packets, even if they are belonging to previous sub-GOPs. In the whole bitrate range in Fig.5.7 and low bitrate range in Fig.5.8, the simplified RVS-LE has similar or higher performance than the Evenly-FEC, which means although the simplified RVS-LE only exploits the *late-arrival* packets of the current sub-GOP, it recovers most of the unavailable packets at low bitrate, because at this range of bitrate the simplified RVS-LE have relative large sub-GOPs. This, consequently, makes its performance comparable to those of Evenly-FEC, that exploits all the *late-arrival* packets. In Fig.5.8, and at high bitrate we notice that the performance of the simplified RVS-LE deteriorates in comparison with evenly-FEC, this is because the sub-GOPs tend to be small at high bitrate, and consequently more and more *late-arrival* packets will not be recovered. This effect does not happen in Fig.5.7, this is because the inserted parity packets are much higher, with f20 than those used for f10.

To evaluate the performance of the proposed approach, for relatively good channel condition, in Fig.5.9 we report the average PSNR versus bitrate for the packet loss and delay pattern in file f3. The reported results are for the following approaches: JM-Error-Free, RVS-LE, simplified RVS-LE, Evenly-FEC, and Evenly-FEC (No Update). It is worth noticing that for the packet loss and delay pattern in file f3, all the packets arrive at the destination before the 300 ms display deadline (this could be seen in Table 5.1); therefore the performance of RVS-LE is similar to the simplified RVS-LE, and Evenly-FEC is also similar to Evenly-FEC (No Update). Thus, in this case where no packet arrive after its display deadline, these results serve to show the effectiveness of the sub-GOP based approach in recovering the physically lost packets.

In order to validate the proposed approach with respect to the maximum allowed

end-to-end delay, in Fig.5.10 we show the average PSNR versus bitrate for different maximum end-to-end delays, namely 200, 250, 300, and 350 ms. These results have been obtained using the same parity packet rate, and the loss and delay pattern in file f10. From these results we could notice that the smaller the allowed delay is, the lower the expected PSNR would be. It is worth noticing that with 200 ms end-to-end delay, the unreceived on-time packet rate is 26.56%, this would have been a big loss rate for a traditional applications that discard the *late-arrival* packets; however, with the proposed approach we could still achieve 33.16dB at 1.2 Mbps bitrate.

Re-decoding and updating the reference buffer will increase the computational complexity at the video receiver side. In Table 5.3, the ratio of slice number that need re-decoding and updating to the total slice number is reported for the Foreman and Coastguard video sequences, where the packet loss and delay pattern is file f10, the parity packet rate is 40%, and the maximum end-to-end delay is set to 300 ms and 200 ms. Two implementation methods with different complexity are used, including RVS-LE using frame-level and slice-level reference buffer re-decoding and updating. From the table it is observed that, with 300 ms maximum end-to-end delay, the average ratio of slices that need re-decoding is 0.131 for the scheme with frame-level re-decoding and updating, whereas for the one with slice-level re-decoding is 0.063. When the maximum end-to-end delay is stringent, i.e., $T_{max} = 200$, this ratio is around 0.4 by using the slice-level updating technique. This information demonstrates that the reference buffer re-decoding and updating technique does not require tremendous computational resource, especially for advanced slice-level updating technique. Another observation is that, slice-level updating technique is especially useful at high bitrate, i.e., low QP value; this is because at high bitrate the slice number per frame is large, so frame-level updating will waste more computational resource.

5.3 Conclusions

In this section, a real-time error resilient video streaming scheme exploiting the *late-arrival* packets and the out-of-order packets has been proposed. In the proposed approach, the *late-arrival* packets are not simply discarded. They are used to boost the reconstructed video quality at the receiver side. In order to better exploit the *late-arrival* packets and the out-of-order packets, we propose to use packets of one sub-GOP, which contains a variable number of frames, as the RS coding block. Given

Table 5.3: The ratio of slice number that need re-decoding and updating to the total slice number; packet loss and delay pattern is file f10, parity packet rate is 40%, maximum end-to-end delay is 300 ms and 200 ms.

Sequence	QP	$T_{max} = 300$ ms		$T_{max} = 200$ ms	
		frame-level	slice-level	frame-level	slice-level
Foreman	22	0.137	0.057	0.813	0.368
	26	0.125	0.059	0.735	0.376
	30	0.137	0.077	0.780	0.498
Coastguard	28	0.142	0.056	0.879	0.384
	32	0.121	0.057	0.758	0.383
	36	0.127	0.074	0.706	0.480
Average		0.131	0.063	0.778	0.414

the maximum end-to-end delay, the RS parity packet rate, the network packet loss rate and the packet delay distributions, a theoretical framework is presented to calculate the optimal sizes of the sub-GOPs and the amount of parity packets for each sub-GOP, so as to achieve the best error resilient performance. Since it is computational prohibitive to get the global optimal solution for the theoretical framework, a fast algorithm is proposed for the practical applications. Meanwhile, a simplified version of the proposed scheme is also presented, where only the *late-arrival* packets within the current sub-GOP are exploited. In order to validate the proposed approach, its performance has been compared with other state-of-the-art real-time error resilient approaches in different environments. The experimental results demonstrate that the proposed approach outperforms the existing error resilient schemes significantly.

It is worth reporting that the work reported in this section has led to the following publications and patent:

1. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Real-time Video Streaming Exploiting the Late-arrival Packets, IEEE Picture Coding Symposium, PCS 2012

2. Jimin Xiao, Tammam Tillo, Chunyu Lin, Yungang Zhang, and Yao Zhao, A Real-time Video Streaming Scheme Exploiting the Late- and Early-arrival Packets, IEEE Transactions on Broadcasting, online in IEEE xplore (2013).

3. Chunyu Lin, Jimin Xiao, Tammam Tillo, Delay Constrained Video Streaming Based on Reed-Solomon Code, application number for China patent: 201110372939.7 (In Chinese)
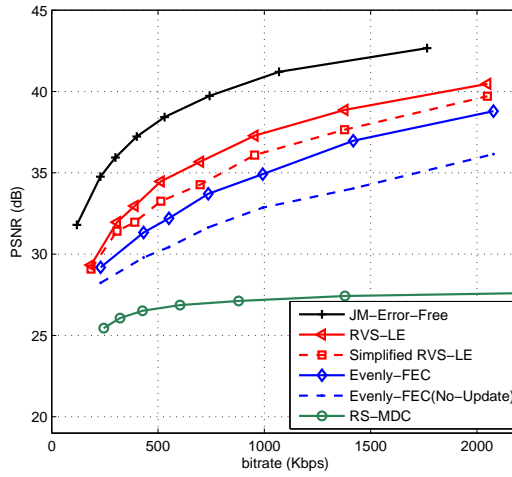
(a) Foreman
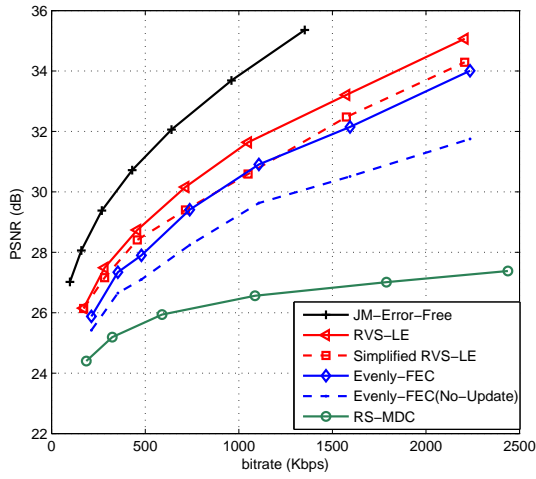
(b) Coastguard
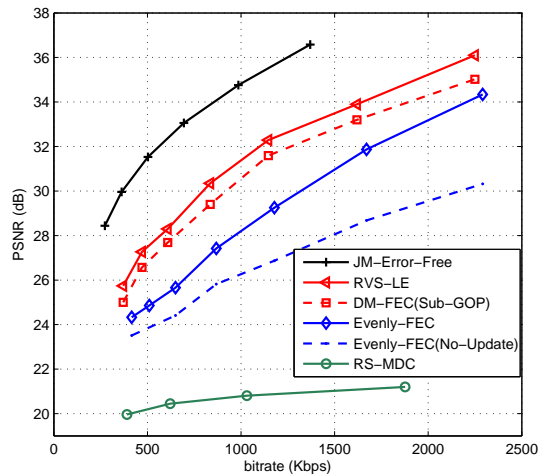
(c) Stefan

Figure 5.7: Average PSNR versus bitrate for different approaches, the packet loss and delay pattern is file f20; (a) Foreman, (b) Coastguard, (c) Stefan.
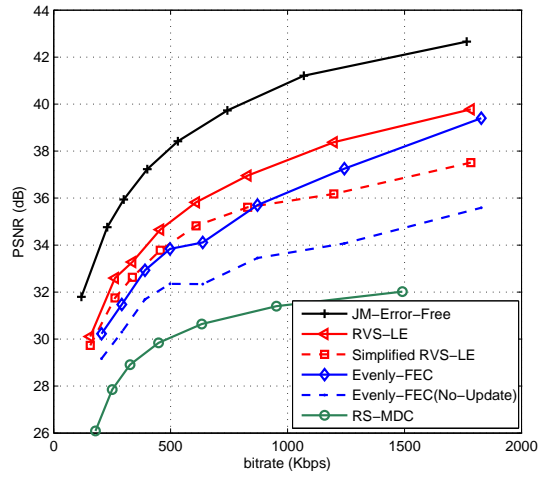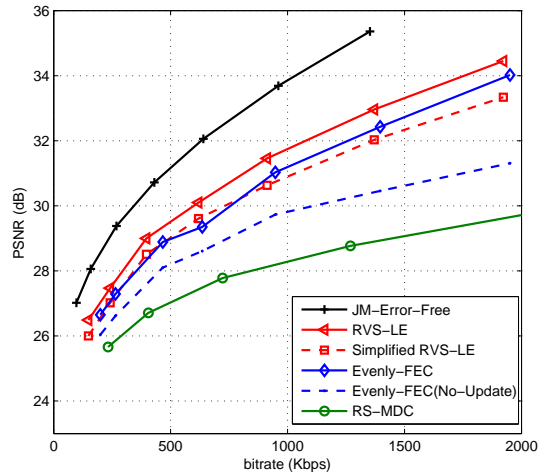
(a) Foreman
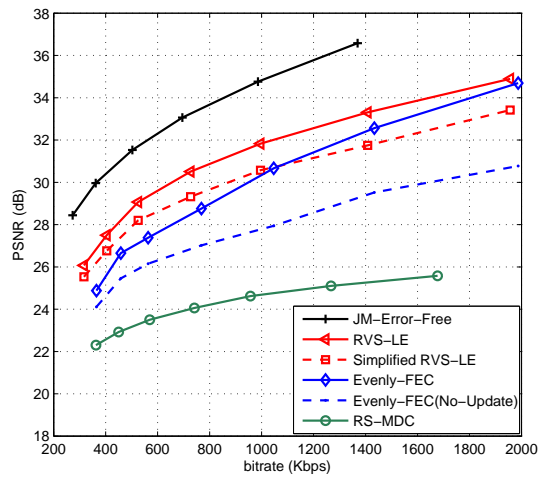
(b) Coastguard

(c) Stefan

Figure 5.8: Average PSNR versus bitrate for different approaches, the packet loss and delay pattern is file f10; (a) Foreman, (b) Coastguard, (c) Stefan.
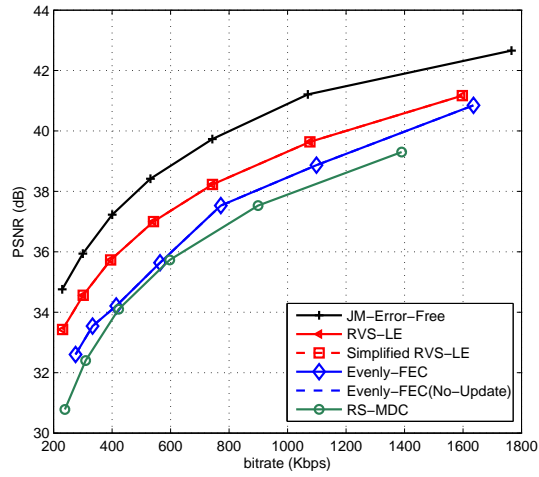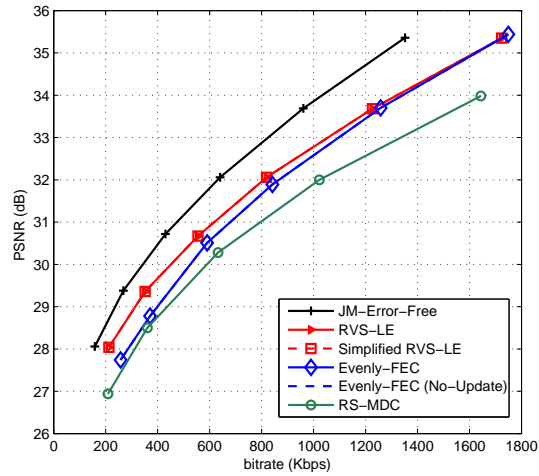
(a) Foreman
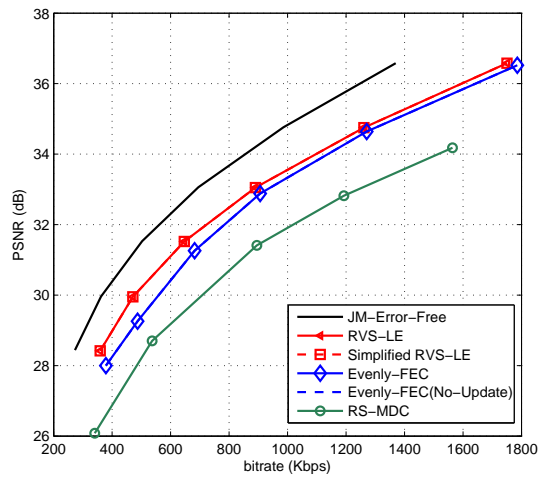
(b) Coastguard

(c) Stefan

Figure 5.9: Average PSNR versus bitrate for different approaches, the packet loss and delay pattern is file f3; (a) Foreman, (b) Coastguard, (c) Stefan.
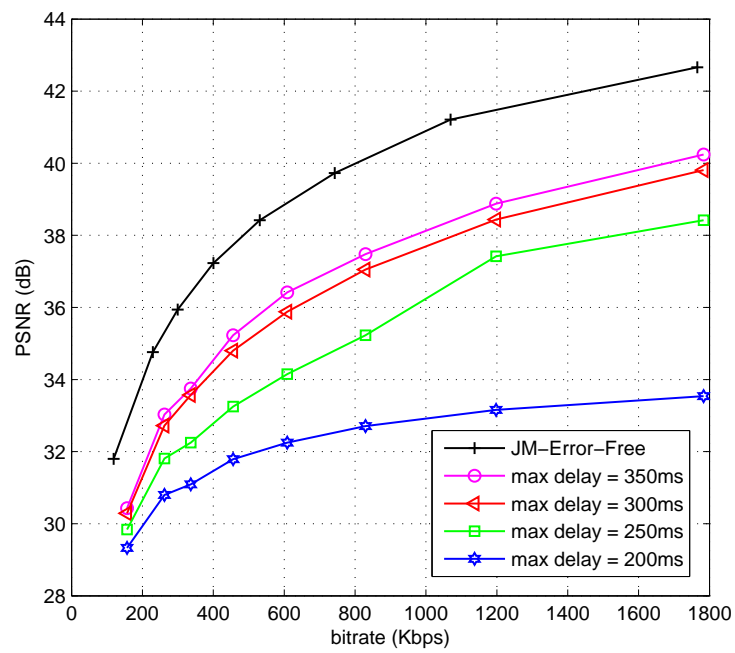
Figure 5.10: Effects of the allowed maximum delay; CIF Foreman sequence, the packet loss and delay pattern is file f10, parity packet rate is 40%.

Chapter 6

Real-Time Video Streaming Using Randomized Expanding Reed-Solomon Code

For the forward error correction coding schemes, the error correction performance and the FEC decoding delay are two contradicting requirements. On one hand, enlarging the FEC coding block size, i.e., grouping video packets from more than one video frame, can improve the error correction performance; however, this will cause some delay equivalent to the length of the FEC coding block. On the other hand, small FEC coding block size, i.e., FEC implemented at frame level, will cause no delay for waiting for the video or parity packets of the following frames; however, the error correction performance is compromised. To solve this problem, in Chapter 4, we proposed the Dynamic sub-GOP FEC Coding (DSGF) approach, where the sub-GOP, which contains video packets of more than one video frame, is used as the FEC coding block. It is found that in the DSGF approach, FEC codes can stop error propagations efficiently, thereby it could provide better overall video quality than frame level FEC, yet no delay will be caused for waiting for the video or parity packets of the following frames. However, for DSGF approach, the FEC error correction capability can only be used by the last frame of each sub-GOP, and consequently the video quality in the sub-GOP could be deteriorated, making the video quality fluctuate frame by frame.

To overcome the previous challenges, in this section, a Randomized Expanding Reed-Solomon (RE-RS) scheme is proposed for real-time video streaming applications. In the proposed RE-RS scheme, RS parity packets are allocated for each frame of the GOP. They are generated using the video packets of the current frame and all the previous frames of the current GOP. At the decoder side, the parity-check equations

of the current frame will be combined with those of all the previous frames. The lost packets will be recovered if the combination of the parity-check equations can be jointly solved. Thereby, these RS parity packets will not only help to recover the lost packets of the current frame, but also the lost packets of the previous frames. It is worth noticing that recovering the lost packets of the previous frames will not affect their timely decoding and visualization. In fact, during their decoding time they will be concealed and displayed, so their later recovering will help reducing the propagated errors. In this scheme, no video packets of the following frames will be used in the current RS coding block, thereby each frame could be decoded and displayed at its display time, and no extra delay will be needed to wait for the source and parity packets of the following frames. Moreover, for the RE-RS scheme, there will be no frame-by-frame video quality fluctuation problem that exists for the DSGF [58] approach, and more in general sub-GOP based approaches. It is worth mentioning that sliding window [48] and expanding window [49, 50] fountain code protection for Scalable Video Coding (SVC) [44] has already been proposed, where they target layered hierarchical data, i.e., SVC. To the best of our knowledge, twisting the expanding window RS code with the reference updating technique for real-time steaming of non-layered data, i.e., H.264/AVC date, is novel. In [48–50], the LT (Luby Transform) or Raptor code was used. However, with the introduction of the RaptorQ code [81] the performance of these methods would be improved, given that RaptorQ code requires less overhead.

The contribution of this section is many fold: firstly, the expanding window RS code is introduced in combination with the reference buffer updating technique for real-time video streaming applications. Secondly, in order to ensure that the equations of different windows could be jointly RS decoded, so as to increase the probability of recovering current and previous losses, we proposed a randomized RS code for the expanding window approach. Thirdly, we investigated the parity allocation problem in the new paradigm of expanding window RS code for video data, and it has been found that evenly allocating the parity packets among frames is a simple yet efficient method. Fourthly, a simplified sliding window scheme is proposed to lower the computational complexity and the memory requirement without compromising its error resilient performance too much.

The rest of the section is organized as follows. Some important RS code property is provided in Section 6.1. In Section 6.2 the proposed RE-RS scheme is presented in

detail. In Section 6.3 some experimental results validating the proposed approach are given. Finally, some conclusions are drawn in Section 6.4.

6.1   Reed-Solomon Code Property for RE-RS

In this section, we will recall some concepts and theory about systematic Reed-Solomon (RS) erasure code, which will be used for the RE-RS approach. The systematic RS erasure code has been widely studied as application layer FEC code to protect data packets against losses in packet erasure networks. In RS (N, K) code, for every K source packets, $N - K$ parity packets are generated to make up a codeword of packets, with a total length of N packets. As long as a client receives at least K out of the N packets, it can recover all the source packets. If the received packet number is less than K, the received source packets can still be used, because they have been kept intact by the systematic RS encoding process. For the RS (N, K) code, the N and K could be any positive integer under the following constraint:

$$\begin{cases} N \leq 2^m - 1 \\ K < N \end{cases} \tag{6.1}$$

where m is the number of bits in a symbol. When $N < 2^m - 1$, it is referred to as the short form of the code. In this case, $2^m - 1 - N$ zero padding packets are added to the K source packets, which makes the total number of packets $2^m - 1 + K - N$ before RS coding, we will call this the full length source code. The RS code will add $N - K$ parity packets, which makes the total packet number $2^m - 1$. After encoding, the padded zero packets are removed to form a so called shortened Reed-Solomon code, whereas at the decoder side these zeros are re-inserted. Let us assume the systematic RS code is $C = (c_1, c_2, \ldots, c_n)$ where $n = 2^m - 1$, and among them t symbols are lost at the decoder side with their indexes being i_1, i_2, $\ldots$, i_t. Thus, $c_{i_1} = X_1$, $c_{i_2} = X_2$, $\ldots$, $c_{i_t} = X_t$ are the t lost variables. The decoder will try to recover the lost packets by solving the parity-check equations:

$$CH^T = 0 \tag{6.2}$$

where H is the parity-check matrix, and it could be denoted as follows:

$$H = \begin{bmatrix} 1 & \alpha & \ldots & \alpha^{2^m-3} & \alpha^{2^m-2} \\ 1 & \alpha^2 & \ldots & (\alpha^2)^{2^m-3} & (\alpha^2)^{2^m-2} \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & \alpha^{N-K} & \ldots & (\alpha^{N-K})^{2^m-3} & (\alpha^{N-K})^{2^m-2} \end{bmatrix} \tag{6.3}$$

with α being the primitive element of Galois Field $GF(2^m)$. Since H is a full rank matrix and its rank is $N - K$, so (6.2) could be solved when the variable number is not more than $N - K$, which also means that it can recover up to $N - K$ erased symbols.

6.2 Randomized Expanding Reed-Solomon Scheme

For real-time FEC video streaming applications, one common approach is to perform RS coding in frame level, which means that the RS coding block contains video packets from the same video frame. Under this constraint, to recover the lost packets, the RS decoder does not need to collect source packets of many frames; therefore there will be no decoding delay. To analyze this approach, that will be used for comparison, let us assume the GOP length is L frames, and the i-th frame has $S(i)$ source packets and $R(i)$ RS parity packets. If we want to have even distribution of the parity packets among all the GOP frames, then $R(i)/S(i)$ needs to be almost constant over all the GOP's frames. In general, the number of generated slices per frame, $S(i)$, varies from frame to frame due to different level of motion level and texture complexity, and because the number of parity packets to be inserted, $R(i)$, should be integer, so we can write $R(i)$ as following:

$$R(i) = \begin{cases} \lceil \mu S(1) \rceil & \text{if } i == 1 \\ \left\lceil \mu \sum_{k=1}^{i} S(k) \right\rceil - \sum_{k=1}^{i-1} R(k) & \text{if } i > 1 \end{cases} \tag{6.4}$$

where $\mu = (N - K)/K$ is the redundant packet rate of RS code, and operation $\lceil X \rceil$ is used to get the minimum integer number greater than or equal to X. In this case, using formula (6.4) makes the average inserted redundant packet rate among several frames approach μ. Since in this approach, the RS parity packets are almost evenly allocated among all the video frames, so this will be called Evenly FEC in the following. An simplified example of the Evenly FEC is shown in Fig.6.1-(a), where for $\forall i\ S(i) = 4$, and $\mu = 0.5$.

For the Evenly FEC, there are two fundamental problems that make its error correction performance low:

1. The number of video source packets generated in each frame is small, which makes the RS code inefficient. To better visualize the effects of K, the number of source packets, on the error correction capability of the RS code, Table 4.1 lists the remaining packet loss rate, p', after the RS code correction, for different values of
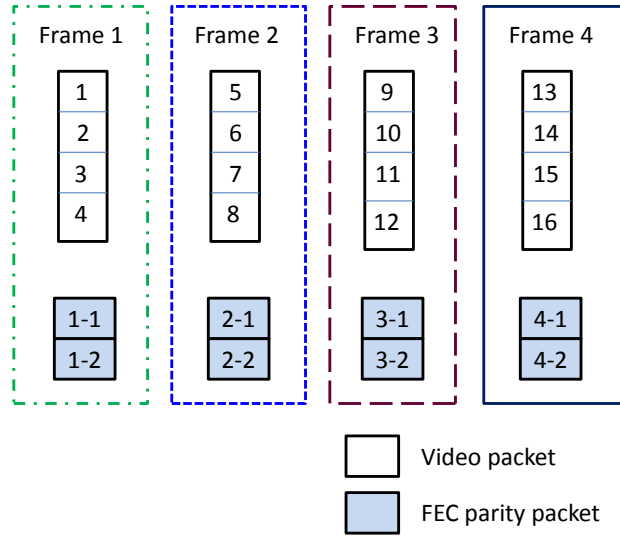
(a) Evenly FEC
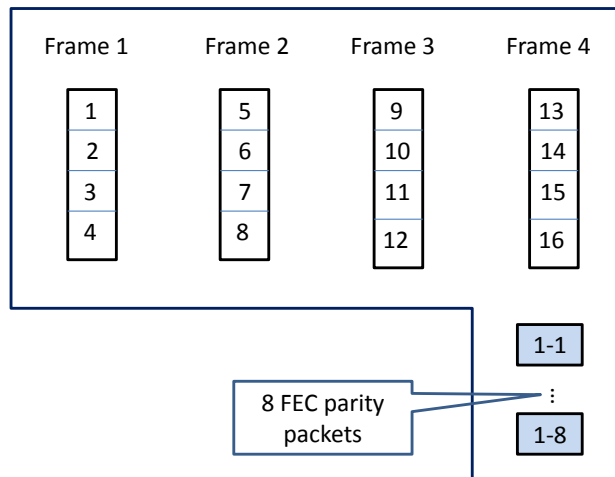
(b) sub-GOP based FEC
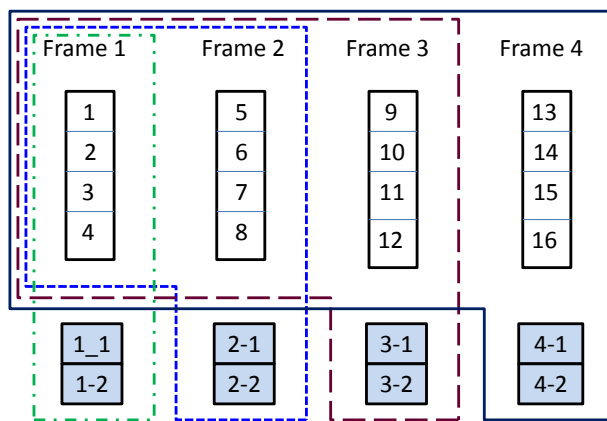
(c) proposed RE-RS scheme

Figure 6.1: Examples of different FEC schemes, where each frame has 4 video packets and redundant packet rate $\mu = 0.5$. (a) Evenly FEC; (b) sub-GOP based FEC; (c) proposed RE-RS scheme.

K. This table demonstrates that for the same packet loss rate and redundancy, the smaller the RS coding block the lower the performance of the RS codes.

2. The error correction capability of the current frame cannot help to recover the lost packets of the previous frames, which may cause the distortion of the unrecovered packets in the previous frames propagate to the current and following frames. Let us take the first two frames in Fig.6.1-(a) as an example. Let us assume three video packets in the first P-frame are lost, while other packets are received. In this case, the RS code of the first P-frame will fail to recover the three lost packets. Meanwhile, as there is no packet loss in the second frame, its error correction capability will not be exploited, and it cannot be used to help recovering the previous losses in the first P-frame. If this error correction capability could be exploited to help to recover the lost packets in the previous frames, then it might be possible to recover the three lost packets, and thereby the propagation distortion from the previous frames will be reduced.

In order to overcome the above problems of the Evenly FEC scheme, we could come out with two possible solutions that are described as follows.

6.2.1 sub-GOP Based FEC Scheme

One solution is grouping video packets of one sub-GOP, which contains more than one video frame, into one RS coding block. Fig.6.1-(b) shows one example for this case. By doing this, the K value of the RS code will be increased, and consequently this will improve the performance of the RS code. However, for this solution, one sub-GOP of delay will be caused if the video frames will be decoded and displayed at the end of its sub-GOP. On the other hand, if the sub-GOP based approach is to be used for the real-time applications where it is not possible to wait for the video or parity packets of the following frames, the average video quality might deteriorate with respect to the approach that tolerates delay. In [58], we proposed to decode and display the frames in real-time fashion, but by the end of each sub-GOP, if the lost packets could be recovered by the RS code, the reference buffer will be updated using the recovered information to stop the error propagations. In this scheme, although the overall performance could be higher than that of the Evenly FEC, the video quality fluctuates frame by frame.

6.2.2 Proposed RE-RS scheme

Another solution which we propose in this section is to use expanding RS code. This scheme is described in Fig.6.1-(c), the RS parity packets are generated using the video packets of the current frame and all the previous frames of the current GOP. At the decoder side, the parity-check equations of the current frame will be combined with those of all the previous frames. The lost packets will be recovered if the combination of the parity-check equations can be jointly solved. Thereby, these RS parity packets can not only help to recover the lost packets of the current frame, but also the lost packets of the previous frames. In this scheme, no video packets of the following frames will be used in the current FEC coding block, thereby each frame could be decoded and displayed at its proper time, and no extra delay will be caused.

To better illustrate the expanding Reed-Solomon scheme, one simplified example is drawn. Let us use the first two frames in Fig.6.1-(c), where each frame has 4 video packets and 2 RS parity packets. We assume packets 1, 2 and 3 in the first frame and packet 5 in the second frame are lost, whereas other video packets and parity packets of the two frames are received. In this case, as described in (6.2), the parity-check equations for the first frame could be simplified as following:

$$\begin{cases} X_1 + \alpha X_2 + \alpha^2 X_3 + C_1 = 0 \\ X_1 + (\alpha^2)X_2 + (\alpha^2)^2 X_3 + C_2 = 0 \end{cases} \tag{6.5}$$

and the parity-check equations for the second frame combined with that of the first frame are as following:

$$\begin{cases} X_1 + \alpha X_2 + \alpha^2 X_3 + C_1 = 0 \\ X_1 + (\alpha^2)X_2 + (\alpha^2)^2 X_3 + C_2 = 0 \\ X_1 + \alpha X_2 + \alpha^2 X_3 + \alpha^4 X_5 + C_3 = 0 \\ X_1 + (\alpha^2)X_2 + (\alpha^2)^2 X_3 + (\alpha^2)^4 X_5 + C_4 = 0 \end{cases} \tag{6.6}$$

where X_1, X_2, X_3 and X_5 denote the four lost packets; C_1, C_2, C_3 and C_4 are constant values, which are determined by the received packets. It should be mentioned that, in order to recover the lost packets, the above equations should be solved in Galois Field of $GF(2^m)$. Here we should note that in the first and third equations in (6.6), the coefficients for X_1, X_2, X_3 are the same, and this also happens for the second and fourth equations. Therefore, for (6.5) and (6.6), the rank of coefficient matrix is 2 and 3, respectively. Hence, (6.5) and (6.6) cannot be solved with three and four variables, respectively. In other words, all the four lost packets, in this example, cannot be recovered.

In order to tackle this problem, we need to increase the rank of (6.6) to 4, for this reason we propose to randomly reorder the video packets and the zero padded packets (if any) before the RS encoding stage. This is a key step to ensure that, with high probability the coefficients of the parity-check equations for different frames are independent, thereby ensure the high error correction performance of the proposed RE-RS scheme. Let us see an example where we assume that each symbol has 4 bits, or in other words, the Galois Field is $GF(2^4)$, and RS code $(16,14)$ is used [1], which means that if we have only 4 source packets, 10 zero packets will be added before generating the 2 parity packets to make the full length RS code. Let us assume for the first RS window, the randomly reordering positions for the lost packets $\{1,2,3\}$ become $\{6,3,11\}$; for the second RS window the randomly reordering positions for the lost packets $\{1,2,3,5\}$ become $\{7,1,4,12\}$. In this case, the combined RS parity-check equations for the second frame becomes:

$$\begin{cases} \alpha^5 X_1 + \alpha^2 X_2 + \alpha^{10} X_3 + C_1{}' = 0 \\ (\alpha^2)^5 X_1 + (\alpha^2)^2 X_2 + (\alpha^2)^{10} X_3 + C_2{}' = 0 \\ \alpha^6 X_1 + X_2 + \alpha^3 X_3 + \alpha^{11} X_5 + C_3{}' = 0 \\ (\alpha^2)^6 X_1 + X_2 + (\alpha^2)^3 X_3 + (\alpha^2)^{11} X_5 + C_4{}' = 0 \end{cases} \quad (6.7)$$

In the Galois Field $GF(2^4)$, the coefficients matrix for (6.7) is

$$\begin{bmatrix} \alpha^5 & \alpha^2 & \alpha^{10} & 0 \\ \alpha^{10} & \alpha^4 & \alpha^{20} & 0 \\ \alpha^6 & 1 & \alpha^3 & \alpha^{11} \\ \alpha^{12} & 1 & \alpha^6 & \alpha^{22} \end{bmatrix} = \begin{bmatrix} \alpha^5 & \alpha^2 & \alpha^{10} & 0 \\ \alpha^{10} & \alpha^4 & \alpha^5 & 0 \\ \alpha^6 & 1 & \alpha^3 & \alpha^{11} \\ \alpha^{12} & 1 & \alpha^6 & \alpha^7 \end{bmatrix}.$$

It is worth noticing that the rank of this matrix is 4, so it is full rank matrix, and the equations can be solved. Therefore, by the second frame all the four lost packets can be recovered.

6.2.3 Detailed Procedure of RE-RS Scheme

Since our objective is to design FEC video transmission system for real-time applications while minimizing the delay caused by the encoding stage; therefore B-frame will not be used, so the IPPP GOP structure will be used. This choice is also justified by the fact that video telephony, the most commonly used applications for real-time system, typically uses the baseline profile of H.264/AVC, where only I-frames and P-frames are used [10]. To make the RS code efficient, fixed length slice scheme in term of byte, will be used to create slices. In this method, the macroblocks in each frame will be

---

[1] This means that at maximal three frames could be protected in this case study.

scanned in raster-scan order and encapsulated into slices with the constraint that the size of each slice should not be more than the target length of the slices; therefore, the length of all the slices except the last ones in each frame will be very close to the target length. The last slice in each frame will be in general smaller than the target length. Thus, for slices other than the last one, only very few zero bytes are padded to reach the target packet length. For the last slice in each frame, usually more dummy zero bytes are used for padding. It is important to note that, in the proposed scheme, the length of the packets used to encapsulate the RS parity symbols is similar to the length of the packets used to encapsulate video slice data, and this latter is dictated by the Maximum Transmission Unit (MTU) of the underlying networks. So consequently, throughout this section, the term packet and slice are used interchangeably, as one packet per slice packetization method is adopted.

The RE-RS procedure at the video sender side works as following:

1. RS parity packets are allocated for each frame in the GOP using (6.4); that means the redundancy is evenly distributed among the GOP frames, and $R(i)$ is the amount of parity packets inserted for the i-th frame.

2. Video packets of the current frame and all the previous frames of the current GOP are collected. If the total video packets number is less than $2^m - 1 - R(i)$, zero padding is used. All video packets are ordered as they are generated by the H.264/AVC video encoder, where the zero padding packets are appended after the video packets. Let us take the second RS window in Fig.6.1-(c) for example, the order of the video packets is: 1, 2, 3, 4, 5, 6, 7, 8.

3. The $2^m - 1 - R(i)$ packets are randomly reordered. Let us assume that for the ith frame, the new position for the kth packet is $O_i(k)$, and $O_i(k)$ should meet the following requirements:

$$\begin{cases} O_i(k) \in [1, 2^m - 1 - R(i)] \\ O_i(k_1) \neq O_i(k_2), \forall \ k_1 \neq k_2 \end{cases} \tag{6.8}$$

Here it is important to note that for different frames, different reordering maps should be used. Moreover, in order to make the decoder work properly, the sender and receiver should have the same maps.

4. $R(i)$ RS parity packets are generated using the reordered $2^m - 1 - R(i)$ video

packets and zero padded packets (if any). Taking the second RS window in Fig.6.1-(c) as example, the generated parity packets are 2-1 and 2-2.

5. Together with the video packets of the current frame, $R(i)$ RS parity packets are transmitted to the receiver side.

6. Repeat steps 2-5 for all the video frames in the current GOP.

At the video receiver side, all the received packets of the previous frames in the current GOP will be kept in the buffer, and the decoding procedure of the RE-RS scheme will work as following:

1. The receiver will collect the video packets of the current frame, and together with the previously received and buffered packets of the current GOP, they will be reordered using the same reordering map used at the video sender side.

2. By multiplying the reordered video packets with the parity-check matrix, the parity-check equations are generated, as in (6.2), the parity-check equations for the current frame include $R(i)$ equations, and they will be kept for the RS decoding of the following frames in this GOP.

3. The parity-check equations of the current frame are combined with all the parity-check equations of the previous frames of the current GOP. The combined equations for the i-th frame will include $\sum_{k=1}^{i} R(k)$ equations.

4. If the combined equations could be solved, and consequently if it allows to recover some of the non-recovered packets in the previous frames, then the frames that these packets belong to and the following frames will be video re-decoded with all the recovered packets, and the reference buffer will be updated with the newly decoded information. It is worth mentioning that the reference buffer updating technique was also used to exploit the late arrival packets to stop error propagations in [38, 39]. If the combined equations cannot be solved, then the current video frame will be video decoded with all the received video packets and the non-arrived slices will be concealed, then the reference buffer will be updated.

5. Repeat all the above process for all the video frames in one GOP.

6.2.4 Performance of Randomly Reordering

The performance of the proposed RE-RS scheme critically depends on removing the linear dependency between the parity-check equations by randomly reordering the source packets. In order to show that the randomly reordering is good for this task, let us take a case study, where the parity packet number for each frame is 1, the lost packet number in the first frame is n, and in the following $n-1$ frames, there is no packet loss, which means that their parity packets will be used to recover the lost packets in the first frame. We assume that for the expanding window of the u-th frame, the position of the v-th lost packet after randomly reordering becomes $i_{u,v}$ ($1 \leq u \leq n$, $1 \leq v \leq n$) with $1 \leq i_{u,v} \leq 2^m - 1$, then the coefficient matrix of the combined parity-check equations of the first n frames is:

$$
\begin{bmatrix}
\alpha^{i_{1,1}-1} & \alpha^{i_{1,2}-1} & \ldots & \alpha^{i_{1,n}-1} \\
\alpha^{i_{2,1}-1} & \alpha^{i_{2,2}-1} & \ldots & \alpha^{i_{2,n}-1} \\
\vdots & \vdots & \vdots & \vdots \\
\alpha^{i_{n,1}-1} & \alpha^{i_{n,2}-1} & \ldots & \alpha^{i_{n,n}-1}
\end{bmatrix} .
\tag{6.9}
$$

Typically $n \ll 2^m - 1$, which means that the elements of the generated matrix by the randomly reordering process could be regarded as being i.i.d. selected from the Galois Field $GF(2^m)$. According to [82], the probability that this matrix is full rank is $\prod_{i=1}^{n} \left(1 - (2^m - 1)^{-i} \right)$. So for example with $m = 8$ and $n \in [1, 10]$, this probability is almost 0.9961. From this we could conclude that using the randomly reordering process can remove the linear dependency between the parity-check equations with high probability.

Based on the above finding, we could conclude that it is reasonable to assume that the randomization process achieves its objective, and this will also be demonstrated by the results obtained with the video sequences in the experimental section. From now on, we will assume that the combined parity-check equations are linearly independent, this means that if we have "proper" redundant packet rate, all the lost packets could be recovered by waiting for several frames so as to accumulate enough parity packets to solve the equations. In the following, we will study the time interval needed to wait so as to recover all the lost packets. To do this, let i represent the index of the current frame and $d(k)$ to denote the lost parity and video packet number for frame k, with $1 \leq k \leq i$. To recover all the lost video packets by the decoding time of the i-th frame,

the set $\Phi = \{d(k), k \in [1, i]\}$ should satisfy the constraint:

$$C(i) = \{\Phi | \sum_{j=0}^{t} d(i-j) \le \sum_{j=0}^{t} R(i-j), \forall t \in [0, i-1]\} \qquad (6.10)$$

where $R(k)$ is the number of RS parity packets for the k-th frame as previously defined. The reason behind this equation is two fold, the first is that for $\forall t \in [0, i-1]$ the lost packets among frames $[i-t, i]$ could only be recovered by using the RS parity packets allocated for these frames, but not the previous parity packets, i.e., those frames before the $(i-t)$-th frame; second, the number of the lost packets should be less than the allocated RS parity packets among frames $[i-t, i]$. Later on, the error correction capability of the following frames could also be used to recover the lost packets in frames $[1, i]$, thereby, all the lost packets in frames $[1, i]$ could be recovered by the time of decoding the j-th frame with $j > i$, if the following condition is satisfied:

$$C'(i, j) = C(i) \bigcup C(i+1) \ldots \bigcup C(j). \qquad (6.11)$$

This is because for $\forall k \in [i, j]$, $C(k)$ insures that all the lost packets among frames $[1, k]$ could be recovered. To numerically evaluate the upper bound performance of the randomization process for parameter setting $\{S, \mu, p\}$, let us define the probability that the set $\{d(k), k \in [1, j]\}$ meets the constraint $C'(i, j)$ by $P(C'(i, j))$. At this point now let us evaluate the probability $P(C'(i, j))$ for a few cases, where the slice number per frame, redundant packet rate and the i.i.d. average packet loss rate $\{S, \mu, p\}$ are $\{5, 0.2, 0.1\}$ and $\{10, 0.2, 0.05\}$, and 10000 trials have been carried out for 10 frames. Fig.6.2 shows the value of $P(C'(i, j))$ for this simulation. It is observed that for the same value of i, the larger the value of j is, the higher probability $P(C'(i, j))$ could be. This is because for larger j, there are more RS parity packets in the following frames that could help to recover the lost packets among frames $[1, i]$. Moreover, we could notice in Fig.6.2-(b), where the average packet loss rate is relatively small in comparison with the redundant packet rate, and a large number of packets per frame are generated, it is almost certain that all the lost packets in previous frames $[1, j-3]$ can be recovered by frame j. In Fig.6.2-(a) the average packet loss rate is relatively high and the number of packet per frame is small, then more time is needed to fully recover the lost packets.

From practical point of view, it should be mentioned that, sending the reordering maps to the receiver side costs some bitrate. Thus, the same reordering maps could be
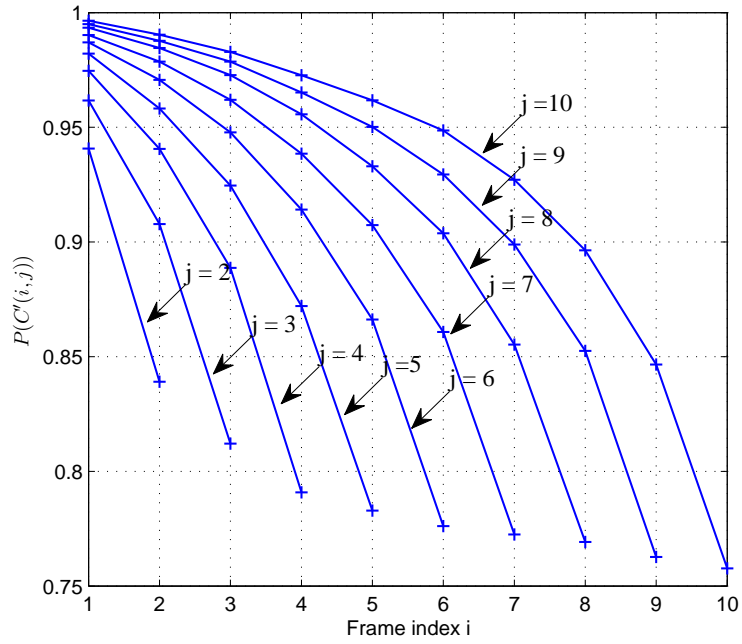
(a)
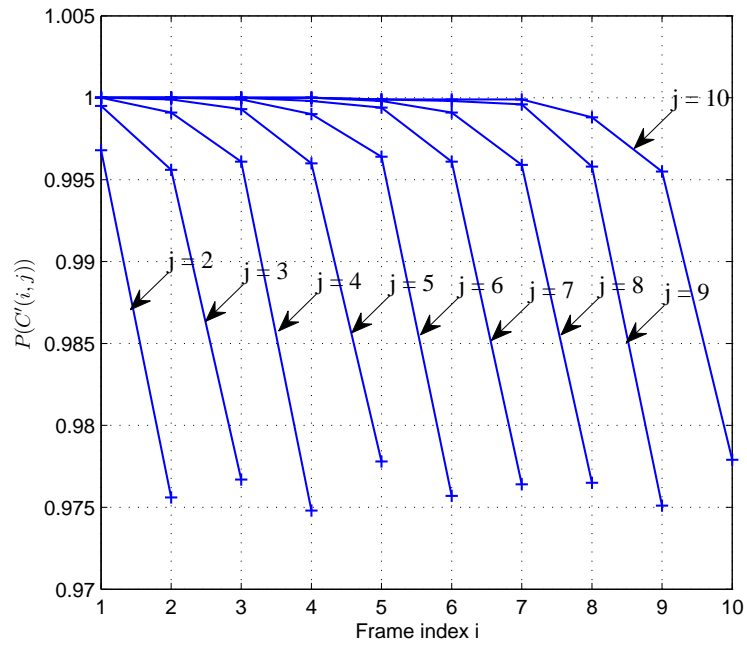
(b)

Figure 6.2: The value of $P(C'(i,j))$ for different i, j; (a) each frame has 5 slices and 1 parity packet, packet loss model is i.i.d., with packet loss rate 10%; (b) each frame has 10 slices and 2 parity packet, packet loss model is i.i.d., with packet loss rate 5%.

used for different GOPs. In this case, the overall bitrate cost of sending the reordering maps could be neglected.

6.2.5 Why Evenly Allocating Parity Packets?

As described in Section.6.2, in the proposed RE-RS scheme, the allocated parity packets are evenly distributed among all the frames using (6.4). In this section, the reason for allocating the parity packets in this fashion will be explained. To simplify the problem, some assumptions will be used: each P-frame has the same number of slices; the mismatch distortion caused by losing each slice is the same; and proper redundant packet rate is used, which means that after certain number of frames, all the lost packets could be recovered. At this point, let us assume a hypothetical scenario in which some packets are lost among frames $[i_1, i_1 + t]$ whereas other packets of the GOP are received intact. Given the assumption that a proper amount of redundancy is inserted, these lost packets could be recovered after w frames. This means that the concealment distortion and propagated distortion will affect the frames $[i_1, i_1 + t + w - 1]$. If now hypothetically we assume that the same pattern of errors affects frames $[i_2, i_2 + t]$, $i_2 \neq i_1$, and we want to insert a certain amount of redundancy to recover these losses, then a question will rise: whether $[i_1, i_1 + t]$ or $[i_2, i_2 + t]$ should be protected more? To answer this, if we take the previous assumptions into consideration, i.e., the frames $[i_1, i_1 + t]$ and $[i_2, i_2 + t]$ have the same number of slices, and each slice lose leads to the same amount of mismatch distortion, and the two groups of frames are randomly chosen, then we could conclude that neither $[i_1, i_1 + t]$ nor $[i_2, i_2 + t]$ should be favored in terms of redundancy; therefore the two groups of frames should be treated equally. In other words, the two groups of frames should have the same pattern of redundancy. So now if we generalize this for different t, then we reach a further conclusion that the pattern of redundancy should be uniform. In Fig.6.2.(b) we could see that having uniform redundancy will lead to constant error propagation window, in other words, if $i \leq j - 3$, then $P(C'(i, j)) \approx 1$. This means that the distortion will only propagate for no more than 3 frames no matter the frame position within one GOP is.

Another way to explain this is to use the iterative method. Let us use the simplified video distortion model, where in one GOP there are L P-frames, each P-frame has S Slices, the distortion caused by losing each slice is same, which is $\bar{d}$. The FEC redundant packet rate is μ, which means for the SL source packets, $R = \mu SL$ total parity packets

117

will be used. Note that this model does not consider the I-frame, because the I-frame, generally, has large number of slices, which makes the FEC performance high, so the probability of recovering all lost packets is very high. We also assume that the distortion of losing one slice will propagate to the following frames without attenuation until the lost slice is able to be recovered by the expanding FEC. The total expected distortion for the L P-frames is $\bar{D}_{total}$. The optimal allocation problem can be formulated as the following constrained minimization:

$$
\begin{cases}
\min & \bar{D}_{total} \\
\text{subject to} & \sum_{i=1}^{L} R(i) \leq R
\end{cases}
\tag{6.14}
$$

As shown in [58], for the R parity packets, there are totally $\binom{L+R-1}{R}$ possible allocation solutions. Obviously, calculating the value of $\bar{D}_{total}$ for all the $\binom{L+R-1}{R}$ allocation patterns is impossible. Since it is computational prohibitive to get the global optimal solution, an iterative method is used to get the allocation which is close to the optimal solution.

The details of the iterative method are described in Algorithm.3. First, the R parity packets are randomly allocated to the L positions. Then, the algorithm will try to find the position where reducing one parity packet can have the lowest distortion impact and the position where adding one parity packet can lead to the greatest distortion impact. One parity packet is moved from the lowest impact position to the greatest impact position. This method is iterated for many times until it reaches a stable state, where the *iteration_num* is initialized as R in Algorithm.3. Finally, the parity allocation algorithm will reach one state which is nearly optimal allocation. It is worth noticing that since it is complex to get $\bar{D}_{total}$ using the probability model, thus it is evaluated using 400 random packet loss patterns.

Using the above iterative method, it is found that the final allocation is very close to evenly allocating the parity packets. To demonstrate this, $\bar{D}_{total}$ versus variance of $R(i)$ for four different parameter sets are reported in Fig.6.3. Each point represents one different allocation pattern with its total parity packet number equals to R. It is observed that, minimizing the value of $\bar{D}_{total}$ requires small variance of $R(i)$, which means that evenly allocating the parity packets can lead to the best error resilient performance.

Algorithm 3 iterative method for allocating the parity packets

Randomly allocate the parity packets, with $\sum_{i=1}^{L} R(i) = R$
$D_1 \Leftarrow \infty$, $D_2 \Leftarrow \infty$
$pos_1 \Leftarrow 0$, $pos_2 \Leftarrow 0$
initialize the iteration number $iteration\_num$
$index \Leftarrow 1$
if $index \leq iteration\_num$ **then**
  for $i = 1$ to L **do**
    $R(i) \Leftarrow R(i) - 1$
    $D_t \Leftarrow$ calculate $\bar{D}_{total}$ with the new allocation
    if $D_t \leq D_1$ **then**
      $D_1 \Leftarrow D_t$
      $pos_1 \Leftarrow i$
    end if
    $R(i) \Leftarrow R(i) + 1$
  end for
  for $i = 1$ to L **do**
    $R(i) \Leftarrow R(i) + 1$
    $D_t \Leftarrow$ calculate $\bar{D}_{total}$ with the new allocation
    if $D_t \leq D_2$ **then**
      $D_2 \Leftarrow D_t$
      $pos_2 \Leftarrow i$
    end if
    $R(i) \Leftarrow R(i) - 1$
  end for
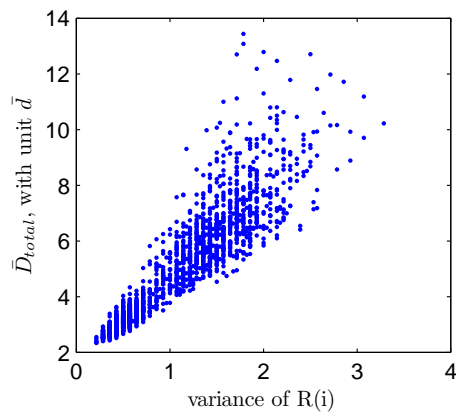 $R(pos_1) \Leftarrow R(pos_1) - 1$
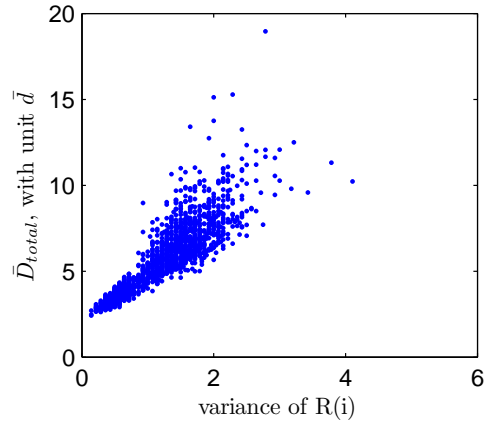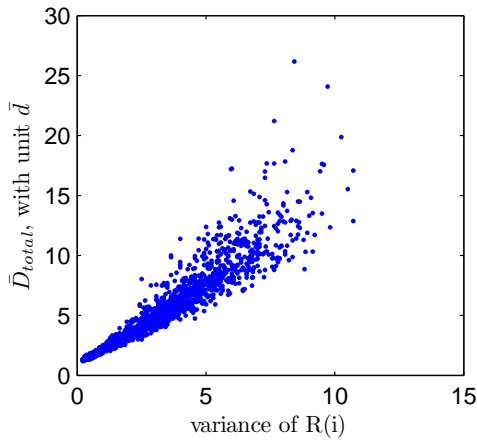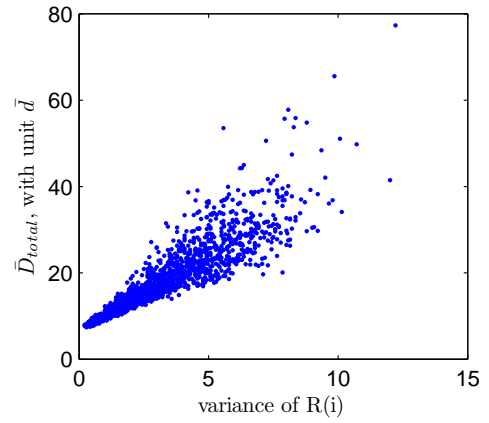 $R(pos_2) \Leftarrow R(pos_2) + 1$
  $index \Leftarrow index + 1$
end if

Figure 6.3: $\bar{D}_{total}$ versus variance of $R(i)$; number of P-frames in one GOP is $L = 29$, other parameters including (a) $\{S, \mu, p\} = \{10, 0.2, 0.05\}$, (b) $\{S, \mu, p\} = \{5, 0.4, 0.1\}$, (c) $\{S, \mu, p\} = \{10, 0.4, 0.1\}$, (d) $\{S, \mu, p\} = \{10, 0.4, 0.15\}$.

6.2.6 Sliding Window RS code: A Simplified Solution

The full version of the proposed scheme requires storing all the video slices and parity packets, so in order to lower the computational complexity and the memory requirement for both the RS encoding/decoding and the reference updating process, one simplified scheme is proposed, where instead of using the expanding window RS code, sliding window RS code is adopted. In other words, the RS parity packets will be generated using the video packets of the current frame and several frames before the current frame, for example, W frames, where W refers to the sliding window size. Accordingly, at the decoder side, the parity packets in the current frame can help to recover the lost packets within its window. The sliding window scheme is based on the assumption that when proper among of parity packets are inserted, for the current frame i, with high probability all the lost packets before this window, i.e., before frame $i - W + 1$, are already recovered, so whether or not the RS coding block includes packets before frame $i - W + 1$ will make no difference. Otherwise, it will fail to recover the lost packets within this window. So this scheme will sacrifice the error resilient performance slightly. For example, for the reported case in Fig.6.2.(b), if a sliding window approach is used, with $W \geq 3$, then the performance will not be sacrificed too much. Moreover, the experimental results reported in Section 6.3 also show that if proper sliding window size is used, its performance gap in comparison with full expanding window approach is not big.

In the proposed system, the decoding process differs from conventional systematic RS decoding, due to the use of reordering of source packets. It also involves joint decoding/error correction of codeword packets of both the current and prior frames. So the fast algorithm of decoder implementation will be essential for power-constrained devices, and this work is left for future research.

6.3 Experimental Results

Our experimental setting is built on the JM14.0 [64] H.264/AVC codec. CIF video sequence Paris, Foreman, Bus, Stefan and Mobile are used for the simulations. We selected these sequences, because they represent different motion and texture characteristics. The GOP structure is IPPP with 30 frames, the beginning 90 frames of each sequence are used for simulation unless otherwise noted. The reference frame number

is one, in other words, only the previous frame is used for prediction. One slice is transmitted in one packet, taking the MTU of networks into account, we set the target slice length as 400 bytes [83] unless otherwise noted. Since at high bitrate, the slice number of one GOP could be large, 10-bit per RS symbol is used, namely Galois Field of $GF(2^{10})$. So the value of N for the RS code (N, K) could be up to 1024, and the value of K depends on N and the per frame parity packet number evaluated using (6.4) in Section.6.2. We use the average luminance Peak Signal-to-Noise Ratio (PSNR) to assess the objective video quality, which is obtained by evaluating the Mean Squared Error (MSE) over all the frames and over 200 trials, then the average PSNR is calculated based on the averaged mse. It is worth mentioning that in all the following reported results, the bitrate includes both the video and parity packets. In our previous work [58], it was shown that the performance of Dynamic sub-GOP FEC Coding (DSGF) approach is higher than many state-of-the-art approaches. Therefore, to have fair comparison we compare our results with DSGF [58] and Evenly FEC, both of which meet the real-time constraint and cause no additional delay.

In the first set of simulations, we study the effects of allocating different redundant packet rates for RS code. The network packet loss is i.i.d. random packet loss model; for the same average packet loss rate $p = 10\%$, we try different RS redundant packet rates, μ, including $\{0.3, 0.4, 0.5, 0.6\}$. We do simulations with various quantization parameters (QP) to span a considerable bitrate range. Fig.6.4 shows the average PSNR versus bitrate curves with different RS redundant packet rates μ. In general, the PSNR curve for redundant packet rate 0.3 is much lower than other cases. The PSNR curves for $\mu = \{0.4, 0.5\}$ are very close; while in low bitrate, higher redundant rate, $\mu = 0.5$, can provide slightly better performance than that of $\mu = 0.4$, and vice versa, in high bitrate, lower redundant rate, $\mu = 0.4$, is slightly better. This is because in low bitrate, the slice number in each frame is small, which makes the performance of RS code low, and high RS redundant packet rate is required to compensate for this. For the PSNR curve of $\mu = 0.6$, although at low bitrate its performance is similar as that of $\mu = \{0.4, 0.5\}$, it is less performing in high bitrate. It it worth indicating that for a fixed total bitrate, higher redundancy means less bitrate could be used for the video date and vice versa; that is why having too high redundant packet rate cannot provide the best performance. In general, the PSNR curves for redundant rate $\{0.3, 0.4\}$ are similar; consequently, in the following simulations, we use RS redundant
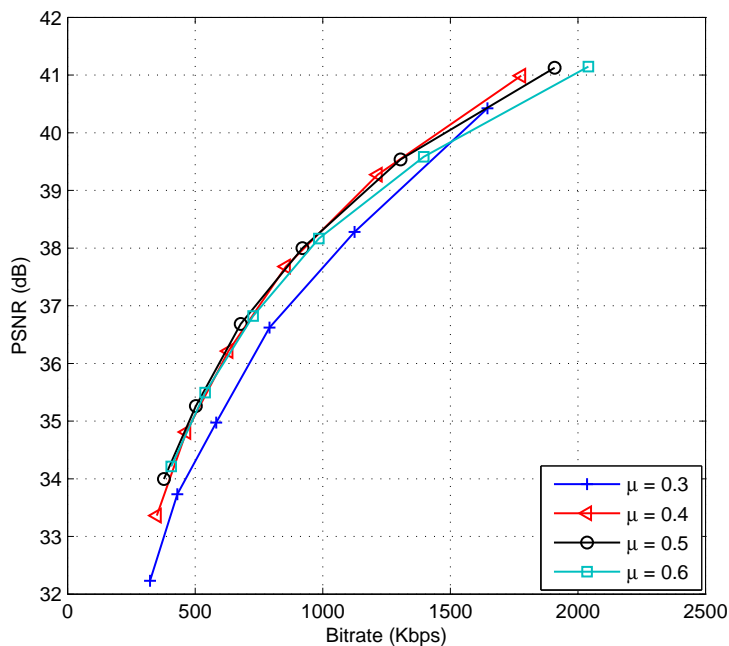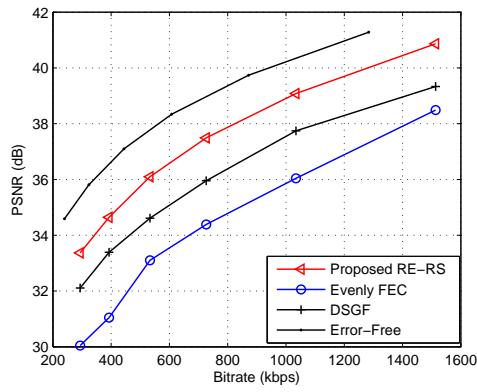
Figure 6.4: Average PSNR versus bitrate for various redundant packet rate μ; CIF Foreman sequence is used; i.i.d. average packet loss rate is 10%; RS redundant packet rate μ includes $\{0.3, 0.4, 0.5, 0.6\}$.

packet rate $\mu = 0.4$ for the 10% packet loss rate. Using the same methods, it is found that for packet loss rate $p = \{5, 10, 15, 20\}\%$, the proper RS redundant packet rate is $\mu = \{0.2, 0.4, 0.55, 0.7\}$, which means that μ should increase almost linearly with p. Therefore, in later simulations, RS redundant packet rate $\mu = 4p$ will be used. The precise relationship between μ and p is left for future investigation.

Fig.6.5 and 6.6 compare the performance of the three approaches in term of PSNR versus bitrate and for i.i.d. average packet loss rate of 5% and 10%, respectively. More-over, the H.264/AVC error free case is reported with the same H.264/AVC parameters that we used in the other three approaches, and this serves to show the up-bound of the performance. Clearly, for all the video sequences, and in the whole bitrate range, the RE-RS scheme outperforms the other two approaches significantly. Specifically, for the Foreman sequence and 10% i.i.d. average packet loss rate, the proposed RE-RS scheme could provide 1.5 dB and 3.0 dB average gain over the DSGF approach and the Evenly FEC approach, respectively.

To have a better understanding of the performance of the proposed RE-RS scheme, in Fig.6.7, its performance is compared with two ULP schemes [1, 2]. We select these
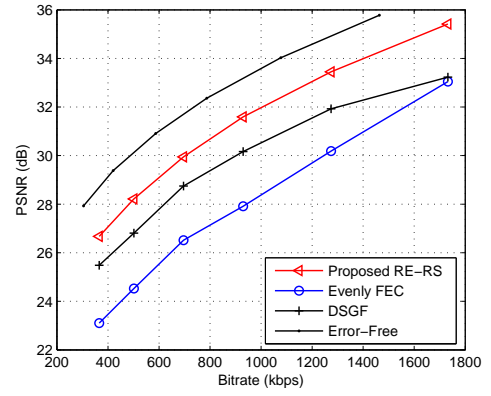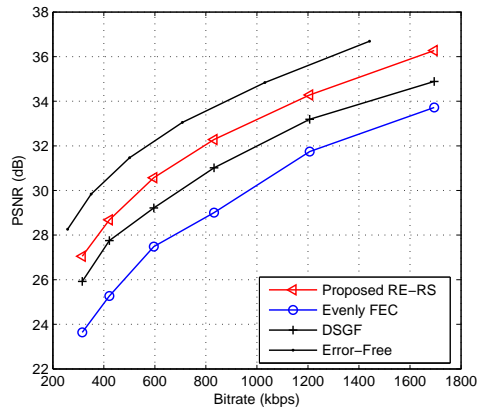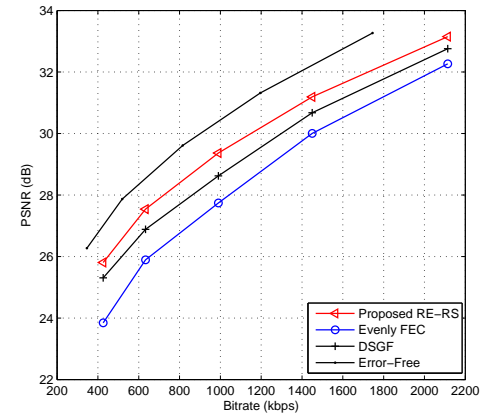
Figure 6.5: Average PSNR versus bitrate curves; i.i.d. average packet loss rate is 5% and the redundant packet rate $\mu = 0.2$; (a) Foreman sequence, (b) Bus sequence, (c) Stefan sequence, (d) Mobile sequence.

Figure 6.6: Average PSNR versus bitrate curves; i.i.d. average packet loss rate is 10% and the redundant packet rate $\mu = 0.4$; (a) Foreman sequence, (b) Bus sequence, (c) Stefan sequence, (d) Mobile sequence.
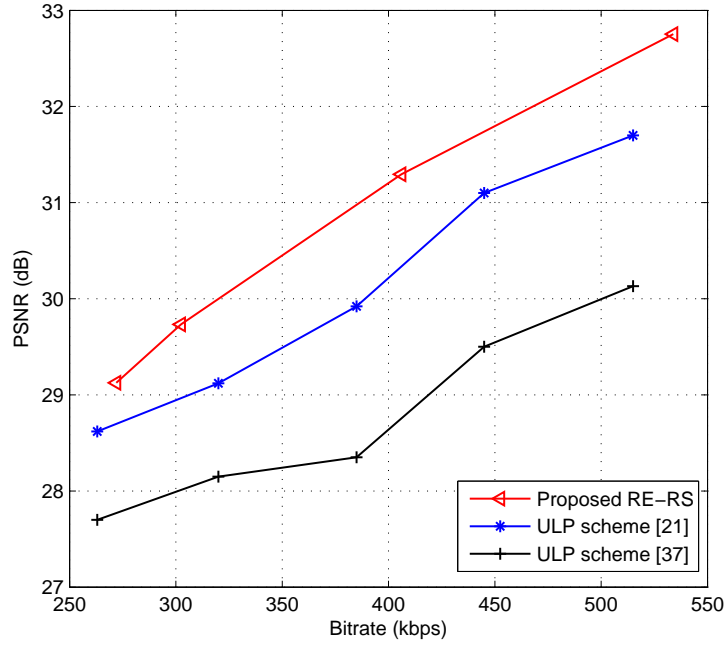
Figure 6.7: Performance comparison between the proposed RE-RS scheme and two ULP schemes [1, 2] ; CIF Paris sequence; i.i.d. 10% average packet loss rate.

two ULP schemes because both of them are implemented at frame-level, which means that they could be used for real-time applications, and this shares the same object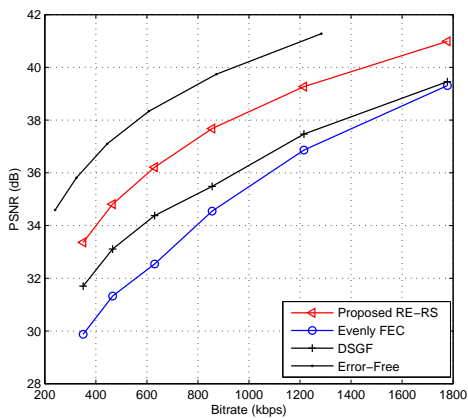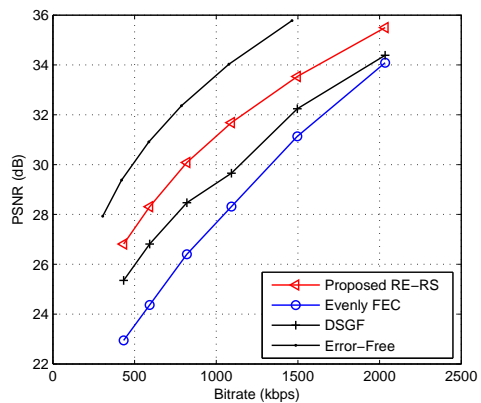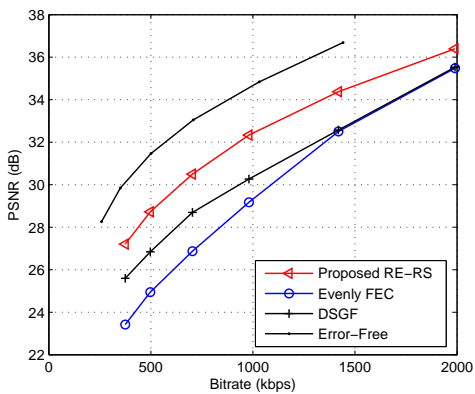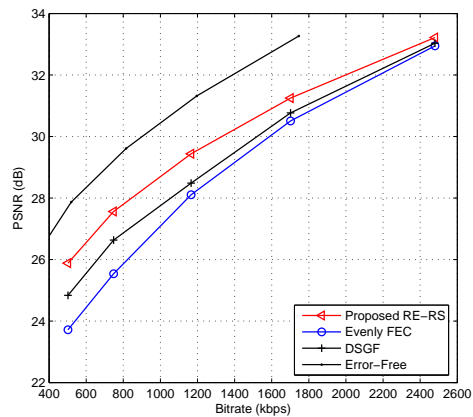ive as the proposed RE-RS scheme. Meanwhile, these two ULP schemes are based on different criteria: [1] is based on the importance of each macroblock, so more protection is allocated for important macroblocks, whereas [2] is based on the concept of data partitioning. To have fair comparison, the same simulation setting as in [1] is used, where 300 frames of the Paris sequence is used and the packet (slice) size is 200 byte. The performance curve of [2] is obtained from [1], which was used as the benchmark. As reported in Fig.6.7, the proposed RE-RS scheme outperforms both the two ULP schemes. The average gap between RE-RS and [1] is about 1 dB, whereas the performance improvement over [2] is even larger. Nevertheless, it should be mentioned that Paris sequence has low movement content, and typically, error concealment algorithm works well for this kind of video sequences. So it is expected that for the moderate and fast movement video sequences, the performance gain of RE-RS could be even larger.

In Fig.6.8, with the Foreman and Stefan sequences, the frame by frame average PSNR curves, which are obtained by averaging the frame's mse of all the 200 trials and then evaluating the per frame PSNR, are plotted for the RE-RS scheme, Evenly

126

FEC scheme and the DSGF approach. For the three approaches, the same QP is used to encode the video sequences, and the same amount of RS redundant packet rate is inserted to ensure fair comparison. It is shown that for almost all the frames, the average PSNR of the RE-RS scheme is higher than that of the Evenly FEC scheme. The gain increases with frame number, and at the last frame of the GOP, the average PSNR of the RE-RS scheme could be up to 5 dB higher than that of the Evenly FEC scheme. In the first half of the GOP, the peaks of the DSGF fluctuating PSNR could be as high as that of the RE-RS scheme; however, the PSNR of the RE-RS scheme is much less fluctuating. In fact, some PSNR bottoms of the Evenly FEC could be up to 4 dB lower than that of the RE-RS scheme. Moreover, in the second half of the GOP, even the PSNR peaks of DSGF approach fail to approach that of the RE-RS scheme. It is worth mentioning that similar results are obtained for the Bus sequence and for the other GOPs of the video sequences. In Fig.6.9, the average number of unrecovered packets among frames $[1, i]$, by the time of decoding frame i, is reported for Foreman sequence. From this figure it is observed that by the time of decoding frame i, the average number of unrecovered packets among frames $[1, i]$ for the proposed method is much smaller than the other two approaches for most of the frames. It is also noted that for all the three approaches, the RS code can recover all the lost packets in the first frame (I-frame), this is because the number of source packets in I-frame is large, and consequently the probability that the RS code fails is almost zero. This also explains why the average PSNR of the first frame in Fig.6.8 is higher than the other frames.

In all the previous experiments, i.i.d. random packet loss model is used to simulate the network packet losses. In order to validate the performance of the proposed RE-RS in different error distribution models, in Fig.6.10 the PSNR versus bitrate curves in Gilbert burst loss model is reported. Since the error resilient performance of the DSGF approach [58] is much higher than the Evenly FEC approach, which was reported in [58], we compare the proposed RE-RS results with the DSGF approach. As indicated in [65], we set the average burst length as two. As it is expected, it is found that for both the RE-RS and DSGF approaches, the PSNR curves in burst loss environment are lower than that in i.i.d. cases. It is also found that in burst loss cases, the average gain of the RE-RS scheme over the DSGF approach is 3.4 dB, being larger than that in i.i.d. case, which is 1.5 dB. This is because in burst loss case, several consecutive packets tend to be lost together. In this case, with high probability, the RS code fails
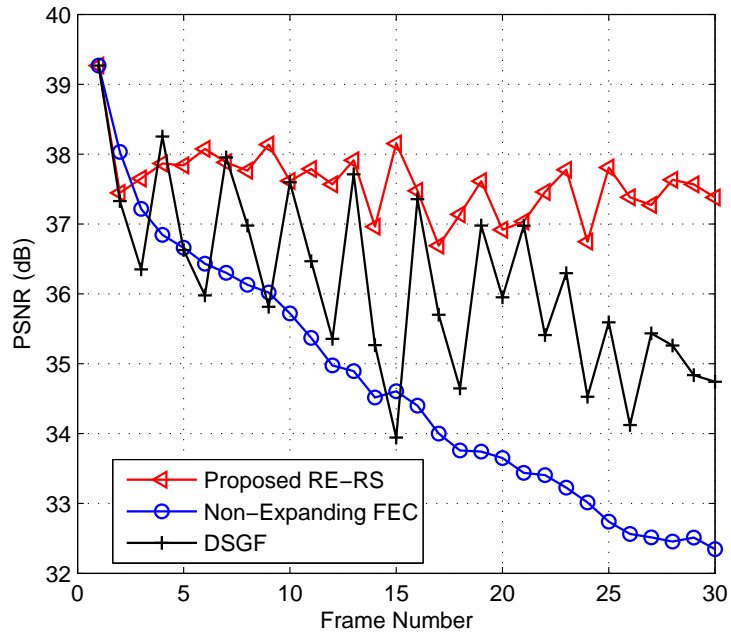
(a) Foreman

(b) Stefan

Figure 6.8: Frame by Frame video quality in one GOP; i.i.d. average packet loss rate is 5%; RS redundant packet rate $\mu = 0.2$; (a) Foreman Sequence, QP = 26, (b) Stefan Sequence, QP = 32.
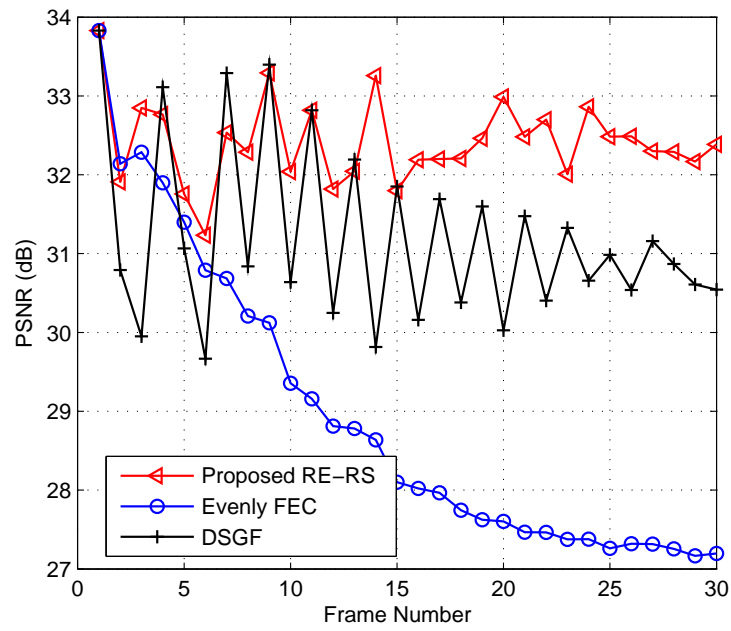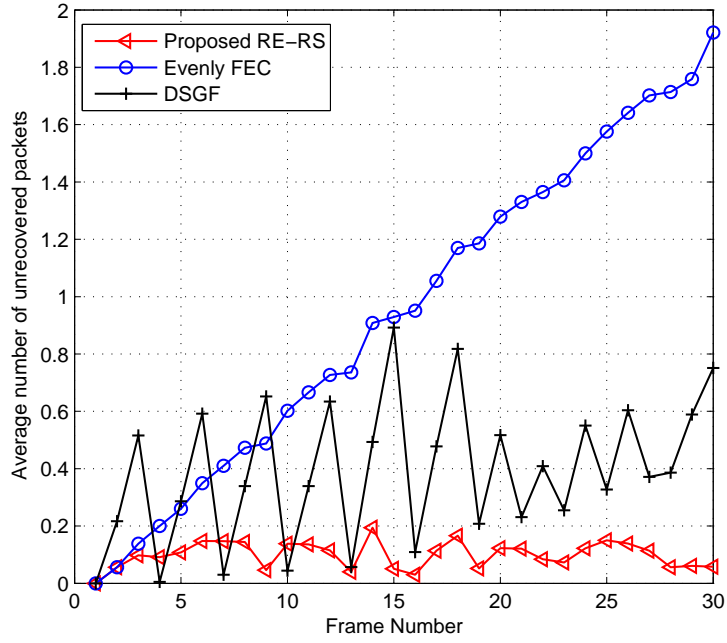
Figure 6.9: The average number of unrecovered packets among frames $[1, i]$ by the time of decoing frame i; i.i.d. average packet loss rate is 5%, RS redundant packet rate $\mu = 0.2$; Foreman Sequence; QP = 26.

to recover the consecutively lost packets. Nevertheless, the burst packet losses are less catastrophic for the RE-RS scheme, this is because for the RE-RS scheme, if the RS code fails to recover the lost packets of the current frame, with high probability, they will be recovered by the expanding RS block of the following frames. Then the reference buffer will be updated, and error propagations will be stopped.

Fig.6.11 reports the PSNR versus bitrate curves for the low complexity sliding window schemes in both i.i.d. and burst packet loss environments. It is noted that, for both i.i.d. and burst packet loss models, the simplified sliding scheme outperforms the DSGF approach in all the bitrate range. Moreover, for the i.i.d. case and in high and intermediate bitrate, the performance of sliding window scheme with window size of 4 is nearly the same as that of expanding window scheme, which suggests that for this simulation scenario RS window size of 4 frames are enough to recover most of the lost packets. The PSNR gap between the sliding window scheme and the expanding window scheme is larger in the burst loss case than in the i.i.d. case, this is because burst packet losses are more difficult to recover, then it usually needs longer sliding window size. Meanwhile, in general, this gap is smaller in high bitrate than in low
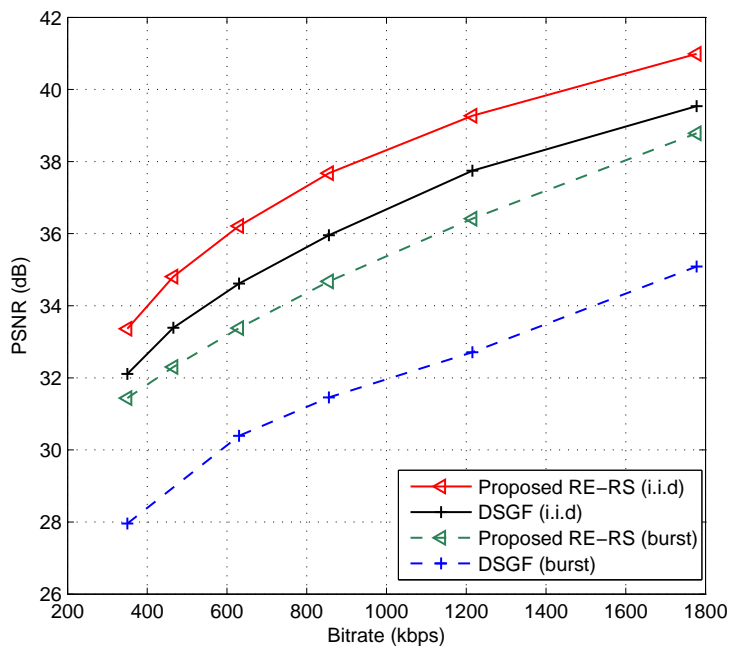
Figure 6.10: Average PSNR versus bitrate curves for the Foreman sequence; average packet loss rate is 10%, including i.i.d. packet loss model and Gilbert burst packet loss model; for burst loss the average burst length is two.

bitrate for both i.i.d. and burst cases, because in high bitrate, the video packet number in each frame is large, which makes the RS code more efficient.

6.4 Conclusions

Facing the dilemma of traditional forward error correction coding of video streams, which is either low error correction performance or long FEC decoding delay, in this section, a real-time error resilient video streaming scheme, named Randomized Expanding Reed-Solomon code, has been proposed. In this scheme, the RS coding block includes not only the video packets of the current video frame but also all the video packets of the previous frames in the current GOP. Thus, the error correction capability of the current frame could also be exploited to recover the lost packets of the previous frames. Therefore, the error propagations from the previous frames could be reduced significantly. To make the parity-check equations of the frames linearly independent, the randomly reordering technique has been proposed. Experimental results demonstrated that the proposed Expanding-RS scheme had considerable practical value for real-time video streaming applications.
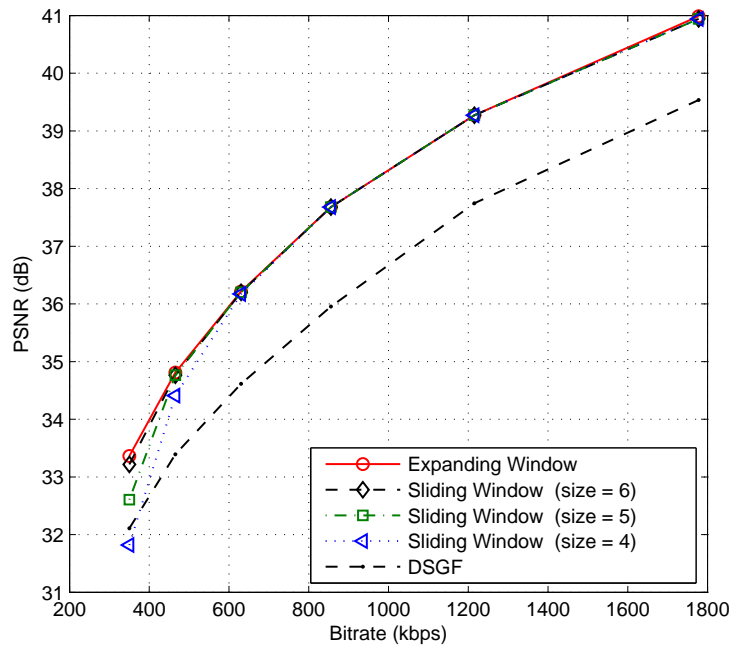
(a) 10% i.i.d. packet loss
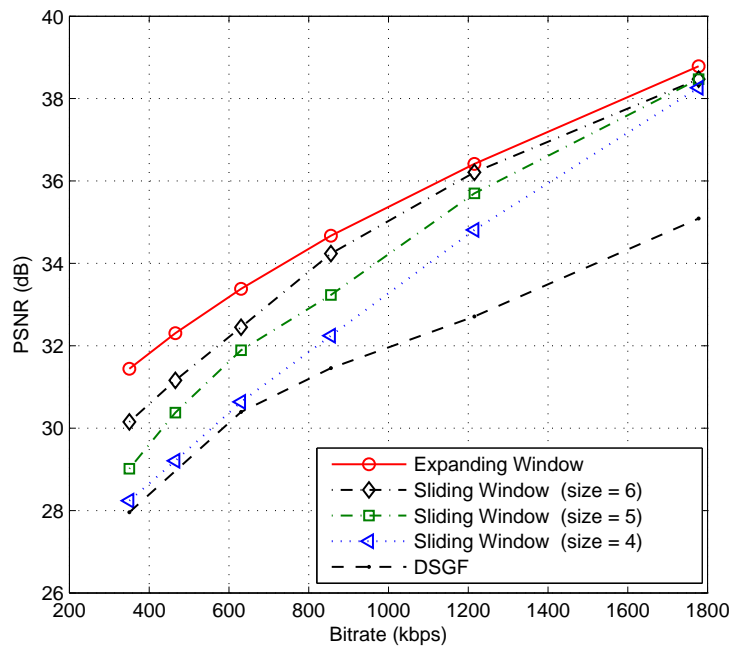
(b) 10% burst packet loss

Figure 6.11: Average PSNR versus bitrate curves for expanding window and sliding window schemes; sliding window size 4, 5 and 6; Foreman sequence; (a) 10% i.i.d. average packet loss rate, the redundant packet rate $\mu = 0.4$; (b) 10% burst packet loss, average burst length is 2, the redundant packet rate $\mu = 0.4$.

It is worth reporting that the work reported in this section has led to the following publication and patent:

1. Jimin Xiao, Tammam Tillo, and Yao Zhao, Real-Time Video Streaming Using Randomized Expanding Reed-Solomon Code, IEEE Transactions on Circuits and Systems for Video Technology, online in IEEE xplore (2013).

2. Tammam Tillo, Jimin Xiao, Real-Time Video Streaming Using Expanding Window Forward Error Correction Code, application number for China patent: 201210169952.7 (In Chinese)

Chapter 7

Conclusion

7.1 Contributions

In this thesis, we have developed and optimized several error resilience techniques for real-time interactive video streaming applications. The main contribution of this thesis is to investigate the relation between error resilient performance and end-to-end encoding/decoding delay. Meanwhile, a set of algorithms have been proposed in this context. Moreover, the required computational complexity has been thoroughly analyzed for real-time video streaming applications. In this thesis, we have found that it is possible to significantly improve the error resilient performance at the expense of either increasing the computational complexity or introducing certain level of delay in the encoding/decoding process. Since the CPU performance is raising with Moore's law, the methods proposed in this thesis, which generally need large computational complexity, should not be a bottleneck, and have practical meanings for real-time video streaming applications.

Specifically, the contributions of this thesis are listed in details as following:

1. We have developed a fine granularity coding and redundancy control scheme based on the redundant slice concept of H.264/AVC. By using the ROPE method, the end-to-end distortion is evaluated by taking into consideration both source coding distortion and channel-induced distortion. The optimal macroblock coding mode and optimal redundant coding parameters are selected. In this thesis, we proposed two different ways to insert redundant information: the first one only uses the redundant motion vector; the other approach is based on providing lower quality redundant version macroblock by using larger QP. Experimental results showed that by using macroblock granularity redundant coding, efficient

resource allocation is realized, which leads to better rate-distortion performance than conventional coarse granularity coding methods.

2. We have developed sub-GOP based FEC coding schemes to improve the error correction performance of the conventional FEC coding. By combining video slices of more than one frame into one sub-GOP for FEC coding, the FEC coding block size is enlarged, which helps to enhance the error correction performance of the FEC code. Optimal sub-GOP and FEC parity allocation solutions are given based on two scenarios. The first one is for the ideal case, where no video transmission network delay is considered. In this case, the proposed DSGF approach provides good performance without adding any FEC encoding/decoding dependency caused delay. Another one is for more practical cases, where the application's maximum end-to-end delay, the network conditions (including network packet losses and delays) and other system parameters are taken into consideration, so as to get the best system performance. The experimental results have shown the advantage of the proposed schemes over other approaches.

3. To further improve the performance of the proposed DSGF approach, and to solve the problem of video quality fluctuation in the DSGF approach, a real-time video streaming scheme using randomized expanding Reed-Solomon code has been proposed. In this scheme, the Reed-Solomon coding block includes not only the video packets of the current frame, but could also all the video packets of previous frames in the current GOP. At the decoder side, the parity-check equations of the current frame are jointly solved with all the parity-check equations of the previous frames. Since video packets of the following frames are not encompassed in the coding block, no delay will be caused for waiting for the video or parity packets of the following frames at both the encoder and decoder sides. Experimental results show that the proposed scheme outperforms other real-time error resilient video streaming approaches significantly.

7.2 Future Work

1. In our future work, we will investigate the optimal redundancy allocation scheme for the proposed sub-GOP based FEC schemes. Currently, the FEC redundancy is allocated in a heuristic way, i.e., proportional to the average network packet loss

rate. Meanwhile, investigating the FEC redundancy allocation problem, for the proposed expanding window (RE-RS) scheme, is also an interesting and important issue. Especially, the optimal redundancy allocation for burst packet losses, which still needs further research work.

In order to tackle the FEC optimal redundancy allocation problem, we are planning to use the Lagrangian Method of Constrained Optimization. In other words, we will fix the video source coding rate, and find relationship between the expected distortion and the redundancy; meanwhile, we will fix the redundancy, and find relationship between the expected distortion and the video source coding rate. Then, the redundancy point where the slope of the expected distortion to the redundancy is the same as the expected distortion to the video source coding rate will be chosen as the optimal redundancy.

2. Our current expanding window (RE-RS) scheme is based on the conventional Reed-Solomon code. The computational complexity of Reed-Solomon code is higher than the emerging fountain code, i.e., the latest RaptorQ code. How to design and benefit from the expanding window FEC coding based RaptorQ code is also worth investigation in the future.

For this issue, we are going to investigate the mechanism of the RaptorQ code, and understand the design of RaptorQ code, and the reason why RaptorQ is efficient in terms of both the error correction performance and the computational complexity. Then, by taking into consideration of both the property of the proposed expanding window scheme and the RaptorQ code, we will try to design a new expanding window scheme based on the RaptorQ code for real-time video streaming.

Appendix A

List of publications

1. Jimin Xiao, Tammam Tillo, and Yao Zhao, Real-Time Video Streaming Using Randomized Expanding Reed-Solomon Code, IEEE Transactions on Circuits and Systems for Video Technology, online in IEEE xplore (2013).

2. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Dynamic Sub-GOP Forward Error Correction Code for Real-time Video Applications, IEEE Transactions on Multimedia, Vol.14, No.4, 2012. doi:10.1109/TMM.2012.2194274

3. Jimin Xiao, Tammam Tillo, Chunyu Lin, Yungang Zhang, and Yao Zhao, A Real-time Video Streaming Scheme Exploiting the Late- and Early-arrival Packets, IEEE Transactions on Broadcasting, online in IEEE xplore (2013).

4. Jimin Xiao, Tammam Tillo, Hui Yuan and Yao Zhao, Macroblock Level Bits Allocation for Depth Maps in 3-D Video Coding, Journal of Singal Processing Systems, PCM 2012 special issue, online

5. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Error Resilient Video Coding with End-to-End Rate-Distortion Optimized at Macroblock Level, EURASIP Journal on Advances in Signal Processing, 2011:80, doi:10.1186/1687-6180-2011-80

6. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Joint Redundant Motion Vector and Intra Macroblock Refreshment for Video Transmission, EURASIP Journal on Image and Video Processing, 2011:12, doi:10.1186/1687-5281-2011-12

7. Chunyu Lin, Tammam Tillo, Jimin Xiao, and Yao Zhao, Optimizing the Deadzone Width to Improve the Polyphase-based Multiple Description Coding, online in Multimedia Tools and Applications (2013)

8. Jimin Xiao, Tammam Tillo, and Hui Yuan, Real-time Macroblock Level Bits Allocation for Depth Maps in 3-D Video Coding, 2012 Pacific-Rim Conference on Multimedia, PCM 2012

9. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Real-time Video Streaming Exploiting the Late-arrival Packets, IEEE Picture Coding Symposium, PCS 2012

10. Jimin Xiao, Tammam Tillo, Chunyu Lin and Yao Zhao, Real-Time Forward Error Correction for Video Transmission, IEEE Visual Communication and Image Processing, VCIP 2011

11. Chunyu Lin, Yao Zhao, Jimin Xiao and Tammam Tillo, Multiple description video coding based on forward error correction Within expanding windows, IEEE International Conference on Image Processing, ICIP 2013

12. Jimin Xiao, Tammam Tillo and Chunyu Lin, Is burst loss worse than random loss for video transmission, 2011 18th International Conference on Systems, Signals and Image Processing (IWSSIP), pages 213–216, Sarajevo, Bosnia and Herzegovina, Jun. 2011

13. Jimin Xiao, Tammam Tillo, Xuan Zhang, Image Database Encoding using HYBRID Video Encoder, The 4th International Congress on Image and Signal Processing (CISP 2011), Shanghai, China, Oct. 2011

14. Yina Liu, Tammam Tillo, Jimin Xiao, EngGee Lim, Zhao Wang, 2D to Cylindrical Inverse Projection of the Wireless Capsule Endoscopy Images, The 4th International Congress on Image and Signal Processing (CISP 2011), Shanghai, China, Oct. 2011

Bibliography

[1] N. Thomos, S. Argyropoulos, N.V. Boulgouris, and M.G. Strintzis. Robust transmission of h. 264/avc video using adaptive slice grouping and unequal error protection. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 593–596. IEEE, 2006.

[2] O. Harmanci and A.M. Tekalp. Stochastic frame buffers for rate distortion optimized loss resilient video communications. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 1, pages I–789. IEEE, 2005.

[3] S. Wenger. H. 264/avc over ip. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):645–656, 2003.

[4] T. Stockhammer, M.M. Hannuksela, and T. Wiegand. H. 264/avc in wireless environments. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):657–673, 2003.

[5] *Recommendation ITU-T G.114, One-Way Transmission Time*, ITU, 1996.

[6] T. Tillo, M. Grangetto, and G. Olmo. Redundant slice optimal allocation for h.264 multiple description coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(1):59 –70, Jan. 2008.

[7] Y. Xu, and C. Zhu. End-to-End Rate-Distortion Optimized Description Generation for H.264 Multiple Description Video Coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 23(9):1523 –1536, Sept. 2013.

[8] P. Baccichet, S. Rane, A. Chimienti, and B. Girod. Robust low-delay video transmission using h.264/avc redundant slices and flexible macroblock ordering. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 4, pages 93–96, 16 2007-oct. 19 2007.

[9] I. Draft. Recommendation and final draft international standard of joint video specification (itu-t rec. h. 264— iso/iec 14496-10 avc). *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVTG050*, 2003.

[10] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h.264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560 –576, 2003.

[11] K. Rijkse. H.263: Video coding for low-bit-rate communication. *Communications Magazine, IEEE*, 34(12):42–45, 1996.

[12] PN Tudor. Mpeg-2 video compression. *Electronics & communication engineering journal*, 7(6):257–264, 1995.

[13] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi. Video coding with h. 264/avc: tools, performance, and complexity. *Circuits and Systems magazine, IEEE*, 4(1):7–28, 2004.

[14] G. Côté and F. Kossentini. Optimal intra coding of blocks for robust video communication over the internet. *Signal Processing: Image Communication*, 15(1):25–34, 1999.

[15] Q.F. Zhu and L. Kerofsky. Joint source coding, transport processing, and error concealment for h. 323-based packet video. In *Proc. SPIE VCIP*, volume 3653, pages 52–62, 1999.

[16] R. Zhang, S.L. Regunathan, and K. Rose. Video coding with optimal inter/intramode switching for packet loss resilience. *Selected Areas in Communications, IEEE Journal on*, 18(6):966–976, 2000.

[17] T. Stockhammer, D. Kontopodis, and T. Wiegand. Rate-distortion optimization for jvt/h. 26l video coding in packet loss environment. In *Int. Packet Video Workshop*, 2002.

[18] Y. Zhang, W. Gao, Y. Lu, Q. Huang, and D. Zhao. Joint source-channel rate-distortion optimization for h. 264 video coding over error-prone networks. *Multimedia, IEEE Transactions on*, 9(3):445–454, 2007.

[19] H. Yang. Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in h. 264/avc. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(7):845–856, 2007.

[20] Y. Liao and J.D. Gibson. Enhanced error resilience of video communications for burst losses using an extended rope algorithm. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1853–1856. IEEE, 2009.

[21] S. Wan and E. Izquierdo. Rate-distortion optimized motion-compensated prediction for packet loss resilient video coding. *Image Processing, IEEE Transactions on*, 16(5):1327–1338, 2007.

[22] H. Yang and K. Rose. Optimizing motion compensated prediction for error resilient video coding. *Image Processing, IEEE Transactions on*, 19(1):108–118, 2010.

[23] L. Zhang, Q. Peng, and X. Wu. SSIM-Based error resilient video coding over packet-switched networks. *Advances in Multimedia Information Processing–PCM 2012*, 263–272, 2012, Springer.

[24] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, Apr.2004.

[25] V.A. Vaishampayan. Design of multiple description scalar quantizers. *Information Theory, IEEE Transactions on*, 39(3):821–834, 1993.

[26] V.A. Vaishampayan and S. John. Balanced interframe multiple description video compression. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 3, pages 812–816. IEEE, 1999.

[27] Y. Wang, M.T. Orchard, V. Vaishampayan, and A.R. Reibman. Multiple description coding using pairwise correlating transforms. *Image Processing, IEEE Transactions on*, 10(3):351–366, 2001.

[28] A.R. Reibman, H. Jafarkhani, Y. Wang, M.T. Orchard, and R. Puri. Multiple description coding for video using motion compensated prediction. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 3, pages 837–841. IEEE, 1999.

[29] T. Tillo and G. Olmo. Data-dependent pre-and postprocessing multiple description coding of images. *Image Processing, IEEE Transactions on*, 16(5):1269–1280, 2007.

[30] S. Shirani, M. Gallant, and F. Kossentini. Multiple description image coding using pre-and post-processing. In *Information Technology: Coding and Computing, 2001. Proceedings. International Conference on*, pages 35–39. IEEE, 2001.

[31] M. Gallant, S. Shirani, and F. Kossentini. Standard-compliant multiple description video coding. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 946–949. IEEE, 2001.

[32] N. Franchi, M. Fumagalli, and R. Lancini. Flexible redundancy insertion in a polyphase down sampling multiple description image coding. In *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, volume 2, pages 605–608. IEEE, 2002.

[33] J.G. Apostolopoulos. Reliable video communication over lossy packet networks using multiple state encoding and path diversity. In *Photonics West 2001-Electronic Imaging*, pages 392–409. International Society for Optics and Photonics, 2000.

[34] I. Radulovic, P. Frossard, Y.K. Wang, M.M. Hannuksela, and A. Hallapuro. Multiple description video coding with h. 264/avc redundant pictures. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(1):144–148, 2010.

[35] Chunbo Zhu, Ye-Kui Wang, M.M. Hannuksela, and Houqiang Li. Error resilient video coding using redundant pictures. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(1):3 –14, jan. 2009.

[36] C. Lin, T. Tillo, Y. Zhao, and B. Jeon. Multiple description coding for h. 264/avc with redundancy allocation at macro block level. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(5):589–600, 2011.

[37] J. Xiao, T. Tillo, C. Lin, and Y. Zhao. Error-resilient video coding with end-to-end rate-distortion optimized at macroblock level. *EURASIP Journal on Advances in Signal Processing*, 2011(1):1–10, 2011.

[38] M. Ghanbari. Postprocessing of late cells for packet video. *Circuits and Systems for Video Technology, IEEE Transactions on*, 6(6):669–678, 1996.

[39] I. Rhee and S.R. Joshi. Error recovery for interactive video transmission over the internet. *Selected Areas in Communications, IEEE Journal on*, 18(6):1033–1049, 2000.

[40] S. Fukunaga, T. Nakai, and H. Inoue. Error resilient video coding by dynamic replacing of reference pictures. In *Global Telecommunications Conference, 1996. GLOBECOM'96.'Communications: The Key to Global Prosperity*, volume 3, pages 1503–1508. IEEE, 1996.

[41] W. Tu and E. Steinbach. Proxy-based reference picture selection for real-time video transmission over mobile networks. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4–pp. IEEE, 2005.

[42] T. Sikora. Trends and perspectives in image and video coding. *Proceedings of the IEEE*, 93(1):6–17, 2005.

[43] D.S. Taubman, M.W. Marcellin, and M. Rabbani. Jpeg2000: Image compression fundamentals, standards and practice. *Journal of Electronic Imaging*, 11(2):286–287, 2002.

[44] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the h. 264/avc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(9):1103–1120, 2007.

[45] U. Horn, K. Stuhlmüller, M. Link, and B. Girod. Robust internet video transmission based on scalable coding and unequal error protection. *Signal Processing: Image Communication*, 15(1):77–94, 1999.

[46] A.E. Mohr, E.A. Riskin, and R.E. Ladner. Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction. *Selected Areas in Communications, IEEE Journal on*, 18(6):819–828, 2000.

[47] M. van der Schaar and H. Radha. Unequal packet loss resilience for fine-granular-scalability video. *Multimedia, IEEE Transactions on*, 3(4):381–394, 2001.

[48] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo. Sliding-window raptor codes for efficient scalable wireless video broadcasting with unequal loss protection. *Image Processing, IEEE Transactions on*, 19(6):1491–1503, 2010.

[49] D. Vukobratovic, V. Stankovic, D. Sejdinovic, L. Stankovic, and Z. Xiong. Scalable video multicast using expanding window fountain codes. *Multimedia, IEEE Transactions on*, 11(6):1094–1104, 2009.

[50] C. Hellge, D. Gómez-Barquero, T. Schierl, and T. Wiegand. Layer-aware forward error correction for mobile broadcast of layered media. *Multimedia, IEEE Transactions on*, 13(3):551–562, 2011.

[51] T. Fang and L.P. Chau. A novel unequal error protection approach for error resilient video transmission. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 4022–4025. IEEE, 2005.

[52] C. Zhang, H. Yang, S. Yu, and X. Yang. Gop-level transmission distortion modeling for mobile streaming video. *Signal Processing: Image Communication*, 23(2):116–126, 2008.

[53] Xingjun Zhang, Xiaohong Peng, S. Fowler, and Dajun Wu. Robust h.264/avc video transmission using data partitioning and unequal loss protection. In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 2471 –2477, 29 2010-july 1 2010.

[54] Xing-Jun Zhang, Xiao-Hong Peng, R. Haywood, and T. Porter. Robust video transmission over lossy network by exploiting h.264/avc data partitioning. In *Broadband Communications, Networks and Systems, 2008. BROADNETS 2008. 5th International Conference on*, pages 307 –314, sept. 2008.

[55] X. Yang, C. Zhu, Z.G. Li, X. Lin, and N. Ling. An unequal packet loss resilience scheme for video over the internet. *Multimedia, IEEE Transactions on*, 7(4):753–765, 2005.

[56] T. Tillo, E. Baccaglini, and G. Olmo. Unequal protection of video data according to slice relevance. *Image Processing, IEEE Transactions on*, 20(6):1572–1582, 2011.

[57] Jimin Xiao, Tammam Tillo, Chunyu Lin, and Yao Zhao. Joint redundant motion vector and intra macroblock refreshment for video transmission. *EURASIP Journal on Image and Video Processing*, 2011(1):1–11, 2011.

[58] J. Xiao, T. Tillo, C. Lin, and Y. Zhao. Dynamic sub-gop forward error correction code for real-time video applications. *Multimedia, IEEE Transactions on*, 14(4):1298–1308, 2012.

[59] Jimin Xiao, Tammam Tillo, Chunyu Lin, Yungang Zhang, and Yao Zhao. A real-time error resilient video streaming scheme exploiting the late- and early-arrival packets, *IEEE Transactions on Broadcasting*, online in IEEE xplore (2013) .

[60] Jimin Xiao, Tammam Tillo, and Yao Zhao. Real-time video streaming using randomized expanding reed-solomon code. *Circuits and Systems for Video Technology, IEEE Transactions on*, accpeted.

[61] MB Dissanayake, CTE Hewage, ST Worrall, WAC Fernando, and AM Kondoz. Redundant motion vectors for improved error resilience in h. 264/avc coded video. In *Multimedia and Expo, 2008 IEEE International Conference on*, pages 25–28. IEEE, 2008.

[62] J.R. Chen, C.S. Lu, and K.C. Fan. A significant motion vector protection-based error-resilient scheme in h. 264. In *Multimedia Signal Processing, 2004 IEEE 6th Workshop on*, pages 287–290. IEEE, 2004.

[63] G.J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *Signal Processing Magazine, IEEE*, 15(6):74–90, 1998.

[64] `http://iphome.hhi.de/suehring/tml/download/`.

[65] D. Loguinov and H. Radha. End-to-end internet video traffic dynamics: Statistical study and analysis. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 723–732. IEEE, 2002.

[66] V. Jacobson, R. Frederick, S. Casner, and H. Schulzrinne. Rtp: A transport protocol for real-time applications. 2003.

[67] Y.J. Liang, J.G. Apostolopoulos, and B. Girod. Analysis of packet loss for compressed video: Effect of burst losses and correlation between error frames. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(7):861–874, 2008.

[68] Z. Li, J. Chakareski, X. Niu, Y. Zhang, and W. Gu. Modeling and analysis of distortion caused by markov-model burst packet losses in video transmission. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(7):917–931, 2009.

[69] Y. Wang, S. Wenger, J. Wen, and A.K. Katsaggelos. Error resilient video coding techniques. *Signal Processing Magazine, IEEE*, 17(4):61–82, 2000.

[70] S. Soltani, K. Misra, and H. Radha. Delay constraint error control protocol for real-time video communication. *Multimedia, IEEE Transactions on*, 11(4):742–751, 2009.

[71] P.A. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. *Multimedia, IEEE Transactions on*, 8(2):390–404, 2006.

[72] S. Lin, S. Mao, Y. Wang, and S. Panwar. A reference picture selection scheme for video transmission over ad-hoc networks using multiple paths. In *Proc. of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 96–9, 2001.

[73] P. Frossard. Fec performance in multimedia streaming. *Communications Letters, IEEE*, 5(3):122–124, 2001.

[74] N. Farber, K. Stuhlmuller, and B. Girod. Analysis of error propagation in hybrid video coding with application to error resilience. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 2, pages 550–554. IEEE, 1999.

[75] B. Girod and N. Farber. Feedback-based error control for mobile video transmission. *Proceedings of the IEEE*, 87(10):1707–1723, 1999.

[76] Y.K. Wang, S. Wenger, and M.M. Hannuksela. Common conditions for svc error resilience testing. *ISO/IEC JTC*, 1, 2005.

[77] *Recommendation ITU-T G.114, One-Way Transmission Time*, ITU, 1996.

[78] Y.-K. Wang, S. Wenger, and M. M. Hannuksela. Common conditions for svc error resilience testing. *JVT document P206*, Aug 2005.

[79] X. Zhang, Y. Xu, H. Hu, Y. Liu, Z. Guo, and Y. Wang. Modeling and analysis of skype video calls: Rate control and video quality. *Multimedia, IEEE Transactions on*, 15(6):1446 - 1457, Oct. 2013.

[80] Wen-Jiin Tsai and Jian-Yu Chen. Joint temporal and spatial error concealment for multiple description video coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 20(12):1822 –1833, dec. 2010.

[81] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer. Raptorq forward error correction scheme for object delivery, internet engineering task force. Technical report, Internet Draft draft-ietf-rmt-bb-fec-raptorq-00, Work in progress (January 2010), 2007.

[82] J.P. Brennan and J. Wolfskill. Remarks on the probability the determinant of an $n \times n$-matrix over a finite field vanishes. *Discrete mathematics*, 67(3):311–313, 1987.

[83] P. Baccichet, R. Shantanu, and G. Bernd. Systematic lossy error protection based on h. 264/avc redundant slices and flexible macroblock ordering. *Journal of Zhejiang University-Science A*, 7(5):900–909, 2006.